UDC 004.056.55

MODIFIED BIOMETRIC TEMPLATE PROTECTION METHOD WITH NONLINEAR TRANSFORMATIONS

Onai M. V. – PhD, Associate Professor of the Department of Computer Systems Software, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine. ROR: https://ror.org/00syn5v21. ORCID: https://orcid.org/0000-0002-4938-8355.

Kosenko O. V. – Bachelor student of the Department of Computer Systems Software, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine. ROR: https://ror.org/00syn5v21. ORCID: https://orcid.org/0009-0001-2115-7918.

ABSTRACT

Context. Biometric data is a common option for authentication or identification. However, it is vulnerable and not replaceable in case of stealing. Several methods for constructing protected biometric templates are proposed in literature, one of them is biohashing. However, linearity of biohashing may be a vulnerability. MLP-hash is similar, but adds nonlinearity. It is modified in this work.

Objective. The goal of this work is to develop a modification of MLP-hash which is faster and allows better separation of users by their templates.

Method. This work focuses on modifying MLP-hash, a biohashing variation with nonlinear transformations. One of the proposed changes is the usage of normalization before applying nonlinear transformation in each layer of MLP-hash. Different normalization methods are investigated and compared. The other proposed change is the simplification of the pseudorandom matrices used in each layer of MLP-hash. Each such matrix is replaced by a block matrix in which blocks that are laying on the diagonal are orthonormal matrices and all other blocks are filled with zeros. Each nonzero block is generated from the user's secret token. In order to make the effect of each nonzero block less localized, a pseudorandom permutation is added before each matrix multiplication and also after all layers. Pseudorandom permutations are also generated with the user's secret token as seed. The proposed method can be used in a similar way to how original MLP-hash and biohashing methods are used: it takes the user's secret token and biometric vector of fixed length and outputs a binary vector of fixed length with the same or smaller dimensionality. MLP-hash with block matrices is compared to the original while applying different normalization techniques and different nonlinear transformations.

Results. The proposed modifications, original MLP-hash and biohashing have been implemented in code. Speed and accuracy of user separation with the usage of these methods have been compared on feature vectors extracted from fingerprints with the usage of Gabor filters.

Conclusions. The conducted experiments have shown an increase of speed and ability to separate user templates from the substitution of proposed block matrices and an increase of ability to separate user templates from the usage of normalization. Comparison of different normalizations and nonlinear transformation has also been conducted. The practical usefulness of the developed method is that it is faster and can be used in applications when users expect no delays while still being difficult to invert. The prospects for further research include testing this method with other biometric modalities, other nonlinear transformations and normalization techniques and an analysis of inversion difficulty of the developed method in comparison to MLP-hash and biohashing.

KEYWORDS: biometrics, biometric template, biometric template protection, one-way transformation, biohashing, elliptic curve cryptography, finite fields.

ABBREVIATIONS

AAD is an average absolute deviation;
EER is an equal error rate;
FAR is a false acceptance rate;
FRR is a false rejection rate;
MLP is a multilayer perceptron;
RNG is a random number generator;
ROI is a region of interest.

NOMENCLATURE

B is a pseudorandom orthonormal matrix used in biohashing;

$\mathbf{B}_{\text{block } i}$ is an i-th diagonal block of block matrix **B**;

$\mathbf{B}_i$ is a pseudorandom orthonormal matrix of i-th layer of MLP-hash or its modifications;

b is a number of nonzero blocks;

b is a binary vector resulting from the usage of biohashing, MLP-hash or its modifications;

b_i is an i-th element of **b**;

$b_{\text{layer } i}$ is a number of nonzero blocks in i-th layer;

$F_{i,\theta}(x, y)$ is a value of pixel at position (x, y) of i-th region of ROI tiling after applying Gabor filter with angle θ;

$\mathbf{F}_x$ is a Sobel filter for estimating gradient along x-axis;

$\mathbf{F}_y$ is a Sobel filter for estimating gradient along x-axis;

f() is a nonlinear transformation;

$f_{\text{custom}}()$ is a custom nonlinear transformation defined in this work;

G is a Gabor filter;

$\mathbf{G}_x$ is a gradient estimation along x-axis;

$\mathbf{G}_y$ is a gradient estimation gradient along y-axis;

H(**a**, **b**) is a hamming distance between binary vectors **a**, **b**;

l is an amount of layers of MLP-hash or its modification;

$M(i, j)$ is a gradient magnitude at position (i, j);

m is a dimensionality of an output of biohashing or its modifications;

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16

m_i is a dimensionality of an output of i-th layer of MLP-hash or its modifications;

N_{fa} is a number of false acceptances;

N_{fr} is a number of false rejections;

N_{ta} is a number of correct authentications;

N_{tr} is a number of correct rejections;

n is a dimensionality of biometrics feature vector;

n_c is a number of circles breaking the ROI;

n_d is a number of directions breaking the ROI;

n_g is a number of Gabor filters used;

n_i is a number of pixels in the i-th region of ROI tiling;

n_p is a number of pseudorandom projections in one layer;

n_T is a number of threshold values for FRR and FAR calculations;

O() is a worst-case algorithm complexity;

$P(i, j)$ is a Poincare index at position (i, j);

$\mathbf{P}_i$ is a pseudorandom permutation used in i-th layer;

$\mathbf{P}_{i,1}$ is a pseudorandom permutation used before projection of i-th layer;

$\mathbf{P}_{i,2}$ is a pseudorandom permutation used after projection of i-th layer;

$P_{i,\theta}$ is an average of pixel values in the i-th region of ROI tiling after applying Gabor filter with angle θ;

p_i is a number of pseudorandom projections used in parallel in i-th layer;

$\mathbb{R}$ is the set of all real numbers;

r_{inner} is a radius of an innermost circle in ROI;

r_{ROI} is a radius of ROI;

$S(\mathbf{a}, \mathbf{b})$ is a cosine similarity of vectors $\mathbf{a}$, $\mathbf{b}$;

T is a threshold of a classifier;

t is a secret token;

$V_{i,\theta}$ is an AAD in the i-th region of ROI tiling after applying Gabor filter with angle θ;

$v_x(i, j), v_y(i, j)$ – components of an orientation field;

w is a window size of an orientation field;

$\mathbf{x}$ is a biometric feature vector;

$\mathbf{y}$ is a result of applying pseudorandom projection to feature vector;

$\mathbf{y}_i$ is an output of i-th layer of MLP-hash or its modification;

$\mathbf{y}_{i,\text{block } j}$ is a part of vector $\mathbf{y}_i$ corresponding to the j-th block;

γ is an eccentricity of a Gabor filter;

$\Delta_k(i, j)$ is a k-th orientation difference at position (i, j) restricted by range $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right]$;

$\delta_k(i, j)$ is a k-th orientation difference at position (i, j);

θ is an angle of a Gabor filter;

$\theta(i, j)$ is an orientation at position (i, j);

$\theta_k(i, j)$ is an orientation at k-th neighbour of position (i, j);

λ is a wavelength of a Gabor filter;

$\mu_{\mathbf{x}}$ is a standard deviation of elements within $\mathbf{x}$;

σ is a standard deviation of a Gabor filter;

$\sigma_{\mathbf{x}}$ is a standard deviation of elements within $\mathbf{x}$;

τ is a binarization threshold;

ψ is a phase shift of a Gabor filter.

INTRODUCTION

Authentication and identification based on biometric data has widespread usage, including device access control, biometric ID-cards, area access control etc. During the process of identification or authentication two biometric templates formed from features of users' biometric data are compared to decide if they belong to the same person. Because biometric samples of the same person slightly vary this comparison is not a full match but rather similarity calculation.

Biometric data is private and its usage and protection is heavily regulated by laws. Biometric templates are usually constructed from the most prominent features, which means that its privacy is also required. There are different methods of biometric template protection, which transform biometric data in a way that prevents inversion but turns similar biometric templates into similar protected templates.

The object of study is the process of biometric template protection.

The process of biometric template protection transforms biometric features to a representation that conceals them while allowing user identification or authentication by measuring similarity between templates. This process should be constructed with regard to variations in biometric samples.

The subject of study is biohashing, a biometric template protection method based on pseudorandom projections and modifications of this method.

This work focuses on a modification of biohashing called MLP-hash, which adds nonlinear transformations to make it harder to invert.

The purpose of the work is to increase the speed of MLP-hash and to increase accuracy of user separation by classifiers based on protected templates generated by it.

1 PROBLEM STATEMENT

Required properties of biometric template protection methods include [1]:

1) non-reversibility: it should not be possible to get original biometric data from protected template;

2) accuracy: biometric system should not lose accuracy from the transformation used for protection;

3) diversity: users should be able to create different templates that are not linkable;

4) revocability: ability to replace template in case it gets stolen.

This work focuses on modifying a biometric template protection method based on biohashing [2]. The resulting method must have all properties listed above, and should be usable in the same way as biohashing: taking as input biometric feature vector $\mathbf{x} \in \mathbb{R}^n$ and secret token t and producing $\mathbf{b} \in \{0, 1\}^m$, $m \leq n$. Besides that the developed modification must be faster than the original MLP-hash.

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16

OPEN ACCESS

2 REVIEW OF THE LITERATURE

There are several biometric template protection methods described in literature. These include projection-based methods like biohashing [2], bloom filters [3], index-of-max hashing [4] etc. These methods apply one-way transformation to biometric features to conceal them.

Biohashing is based on pseudorandom projection. This projection depends on the user's secret token t.

Algorithm 1 – Biohashing

Input: $\mathbf{x} \in \mathbb{R}^n$, t
Output: $\mathbf{b} \in \{0, 1\}^m$
1. Generate pseudorandom matrix $\mathbf{B}$ using t as seed
2. Apply Gram-Schmidt process to $\mathbf{B}$
3. $\mathbf{y} \leftarrow \mathbf{B} \cdot \mathbf{x}$
4. for $i = 1$ to m do
  4.1. if $y_i < \tau$ then $b_i \leftarrow 0$
  4.2. else $b_i \leftarrow 1$
  4.3. end if
5. end for
6. return $\mathbf{b}$

Several improvements to base biohashing are proposed in [5]. These include:

1) normalization of feature vectors before applying biohashing;

2) usage of several projection spaces to increase result dimensionality;

3) usage of several feature permutations to increase result dimensionality.

Binarization and dimension reduction of the feature vector after projection make this process a one-way transformation. However, some authors raise concerns about the linear nature of this transformation saying that this transformation may be partially reversed. In [6] there is a demonstration of reversal for projection-based methods. Although results are demonstrated for lower dimensions than those usually used in practice, this demonstration shows that linear nature is in fact a vulnerability of projection-based methods.

Although biohashing creates a protected template that does not fully reveal original biometric in case of being stolen, the created template should still be stored and transmitted securely and should be renewed immediately upon a suspicion of being compromised, else an attacker may use it for impersonation and unauthorized access may be gained. While being sent through an unsafe communication channel this template may be additionally encrypted with the use of a symmetric cipher such as AES or an asymmetric one, for example with the use of elliptic curve cryptography.

MLP-hash [7] is a modification of biohashing which adds nonlinear transformations to it. It contains l pseudorandom projections using matrices $\mathbf{B}_i$ of size $m_i \times m_{i-1}$, $m_0 = n$, $m_l = m$.

Algorithm 2 – MLP-hash

Input: $\mathbf{x} \in \mathbb{R}^n$, t
Output: $\mathbf{b} \in \{0, 1\}^m$
1. $\mathbf{y}_0 \leftarrow \mathbf{x}$
2. for $i = 1$ to l do
  2.1. Generate pseudorandom matrix $\mathbf{B}_i$ using t as seed
  2.2. Apply Gram-Schmidt process to $\mathbf{B}_i$
  2.3. $\mathbf{y}_i \leftarrow f(\mathbf{B}_i \cdot \mathbf{y}_{i-1})$
3. end for
4. for $i = 1$ to m do
  4.1. if $y_i < \tau$ then $b_i \leftarrow 0$
  4.2. else $b_i \leftarrow 1$
  4.3. end if
5. end for
6. return $\mathbf{b}$

Nonlinear transformation makes it more difficult to invert a protected template, especially when this transformation is a many-to-one function, which is in general irreversible. In [7], a ReLU (1) is used as a nonlinear transformation, which is a many-to-one function.

$$\mathrm{ReLU}(x) = \begin{cases} 0 \text{ if } x < 0; \\ x \text{ if } x \geq 0. \end{cases} \quad (1)$$

3 MATERIALS AND METHODS

One of the proposed modifications of MLP-hash is the generalization of improvement proposed in [5], namely, the usage of normalization, for the multilayer structure of MLP-hash by applying normalization before each nonlinear transformation. Experimental results from [5] show that usage of normalization before biohashing improves EER in comparison to base biohashing, therefore it is expected that the usage of normalization in MLP-hash will also improve EER in comparison to base MLP-hash, however, the conclusion can only be driven from experimental evidence. Different normalization methods are tried:

1) normalization by range of values (so that all elements of a normalized vector are within the range [− 1, 1]):

$$2 \cdot \frac{\mathbf{x} - \min\limits_{i=1,\ldots,n} x_i}{\max\limits_{i=1,\ldots,n} x_i - \min\limits_{i=1,\ldots,n} x_i} - 1;$$

2) l_2-normalization (so that the length of a normalized vector equals 1):

$$\frac{\mathbf{x}}{|\mathbf{x}|};$$

3) statistical normalization (so that mean of elements in vector is 0 and standard deviation is 1):

$$\frac{\mathbf{x} - \mu_{\mathbf{x}}}{\sigma_{\mathbf{x}}},$$

where $\mu_{\mathbf{x}}$ is the mean of elements of $\mathbf{x}$ and $\sigma_{\mathbf{x}}$ is the standard deviation of elements in $\mathbf{x}$. It should be noted

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16
OPEN ACCESS

that μ and σ are calculated for elements within one vector, not across vectors, so this normalization should be viewed as a soft normalization by range that fits most but not all elements into range [− 1, 1] rather than a statistical tool.

Another proposed modification is the simplification of pseudorandom matrices used in MLP-hash layers by replacing them with block matrices of the following form:

$$\mathbf{B}_i = \begin{pmatrix} \mathbf{B}_{i,\text{ block }1} & 0 & \cdots & 0 \\ 0 & \mathbf{B}_{i,\text{ block }2} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{B}_{i,\text{ block }b} \end{pmatrix},$$

where $\mathbf{B}_{i,\text{ block }j}$ is a pseudorandom orthonormal matrix of size $\frac{m_i}{b_{\text{layer }i}} \times \frac{m_{i-1}}{b_{\text{layer }i}}$, $m_0 = n$, $m_l = m$. It should be noted that both m_i and m_{i-1} must be divisible by $b_{\text{layer }i}$. If this modification is intended to be used with layer sizes not divisible by block number, it should be extended to work with blocks of different sizes or with intersecting blocks. Whole matrix $\mathbf{B}_i$ is also orthonormal: scalar product of rows intersecting the same block is 0 because extending block rows by zeros does not change it, scalar product of rows intersecting different blocks is 0 because there is no position at which both vectors contain nonzero element, and the norm of each row is still 1, since it is only affected by nonzero elements which are contained in diagonal blocks and these blocks are orthonormal. Thus simplified matrix still defines a pseudorandom projection. It can be thought of as breaking a feature vector into smaller vectors and projecting each using nonzero blocks.

The values in a biometric feature vector may be correlated, often this correlation is stronger in nearby positions. This means that even projected fragments of vector may form patterns similar to those in the original. In order to break locality of nonzero blocks a pseudorandom permutation is introduced. Although such permutations can be inserted before and after each projection in a multilayer structure, forming layers of the following structure:

$$\mathbf{y}_i = \mathrm{f}(\mathbf{P}_{i,2} \cdot \mathbf{B}_i \cdot \mathbf{P}_{i,1} \cdot \mathbf{y}_{i-1}),$$

some of them are redundant if nonlinear transformation can be defined as applying a function to each element of vector that depends only on selected element and an unordered collection of other elements (examples include just applying any nonlinear function element-wise, normalization by methods listed above or a combination of the two), since permutation that happens before such transformation can be brought outside and can be combined with next layers first permutation:

$$\begin{aligned} &\mathbf{P}_{i+1,1} \cdot \mathrm{f}(\mathbf{P}_{i,2} \cdot \mathbf{B}_i \cdot \mathbf{P}_{i,1} \cdot \mathbf{y}_{i-1}) = \\ &= \mathbf{P}_{i+1,1} \cdot \mathbf{P}_{i,2} \cdot \mathrm{f}(\mathbf{B}_i \cdot \mathbf{P}_{i,1} \cdot \mathbf{y}_{i-1}) = \\ &= \mathbf{P}_{i+1,\text{ combined}} \cdot \mathrm{f}(\mathbf{B}_i \cdot \mathbf{P}_{i,1} \cdot \mathbf{y}_{i-1}), \end{aligned}$$

therefore only one of the two permutations is necessary for each layer except the last, after which a pseudorandom permutation is used. Layer structure is then defined as follows:

$$\mathbf{y}_i = \mathrm{f}(\mathbf{B}_i \cdot \mathbf{P}_i \cdot \mathbf{y}_{i-1}).$$

Note that although a pseudorandom permutation is represented here as matrix multiplication for shorter notation, it does not need to be a matrix multiplication in an actual implementation.

Multiplying matrix of size $m \times n$ by vector of size n consists of $m \times n$ products and $m \times (n - 1)$ sums, so it has time complexity $O(mn)$. In comparison, multiplication of the simplified matrix of size $m \times n$ with b blocks (where m and n are divisible by b) by vector of size n can be decomposed into b multiplications of matrix of size $\frac{m}{b} \times \frac{n}{b}$ by vector of size $\frac{n}{b}$, which means that it has time complexity $\mathrm{O}\left(\frac{mn}{b}\right)$. Random permutation in a collection of size n requires at most n copying operations and has a time complexity of $O(n)$. Therefore, a modified projection has a time complexity of $\mathrm{O}\left(\frac{mn}{b} + n\right)$. To generate an orthonormal matrix of size $m \times n$, $m \times n$ random numbers need to be generated and then Gram-Schmidt process needs to be applied. Gram-Schmidt process for matrix of size $m \times n$ consists of subtracting projections of rows on rows above them. For a k-th row, $k - 1$ projections must be calculated, each consisting of a dot product of vectors of size n (consisting of n multiplications and $n - 1$ additions) and of product of resulting scalar and vector of size n (n multiplications). Subtraction of a projection consists of n operations. To make the resulting matrix orthonormal, each row is normalized after subtracting projections (n multiplications and $n - 1$ additions and one square root computation). Therefore, applying Gram-Schmidt process to matrix of size $m \times n$ requires

$$\begin{aligned} &\sum_{k=1}^{m}((k-1)(4n-1)+2n) = \\ &= \frac{m(m-1)}{2}(4n-1)+2mn \end{aligned}$$

arithmetic operations and therefore has a time complexity $O(m^2 n)$, ignoring the difference between addition, subtraction and multiplication. For a simplified matrix with b blocks only $\frac{mn}{b}$ numbers must be generated, and

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16
OPEN ACCESS

Gram-Schmidt process on one matrix of size $\frac{m}{b}\times\frac{n}{b}$ has $O\left(\frac{m^2 n}{b^3}\right)$ time complexity. Generation of a random permutation has $O(n)$ time complexity. In summary, generation and usage of the original pseudorandom projection has time complexity $O(m^2 n)$, while generation and usage of simplified projection has time complexity $O\left(\frac{m^2 n}{b^2}+\frac{mn}{b}+n\right)$.

The simplification of matrices reduces not only time complexity of the method, but also space complexity. The projection matrix of size $m \times n$ requires storage for mn floating-point numbers. Gram-Schmidt process (if it is done in-place) requires additional storage for n numbers for projections it creates (at most one projection needs to be stored at the same time) plus a constant amount of numerical variables. Multiplying matrix of size $m \times n$ by vector of size n produces vector of size n. For a modified method, only $\frac{mn}{b}$ numbers are required to represent a projection matrix and Gram-Schmidt process needs to store only $\frac{n}{b}$ plus a constant amount of number variables at once. A pseudorandom permutation of a vector of size n used in modified layers requires $O(n\ln n)$ storage space (considering that each number stored must be an index of an array of size n). In summary, while generation and usage of pseudorandom projection with projection matrix of size $m \times n$ has space complexity $O(mn)$, generation and usage of simplified projection has space complexity $O\left(\frac{mn}{b}+m+n\ln n\right)$.

It should be taken into consideration that while proposed modification of projection matrices significantly reduces computational complexity, it also makes the process easier to partially invert, which should be taken into consideration while choosing the value of parameter b.

A method combining both of the proposed modifications is defined by the following algorithm.

Algorithm 3 – Modified MLP-hash

Input: $\mathbf{x} \in \mathbb{R}^n$, t
Output: $\mathbf{b} \in \{0, 1\}^m$
1. $\mathbf{y}_0 \leftarrow \mathbf{x}$
2. for $i = 1$ to l do
  2.1. Initialize $\mathbf{y}_i$
  2.2. Apply a pseudorandom permutation to $\mathbf{y}_{i-1}$ with seed t
  2.3. for $j = 1$ to $b_{\text{layer } i}$ do
    2.3.1. Generate pseudorandom matrix $\mathbf{B}_{i,\text{ block } j}$ using t as seed
    2.3.2. Apply Gram-Schmidt process to $\mathbf{B}_{i,\text{ block } j}$
    2.3.3. $\mathbf{y}_{i,\text{ block } j} \leftarrow \mathbf{B}_{i,\text{ block } j} \cdot \mathbf{y}_{i-1,\text{ block } j}$
  2.4. end for
  2.5. normalize $\mathbf{y}_i$
  2.6. $\mathbf{y}_i = f(\mathbf{y}_i)$
3. end for
4. Apply a pseudorandom permutation to $\mathbf{y}_l$ with RNG seed t
5. for $i = 1$ to m do
  5.1. if $y_i < \tau$ then $b_i \leftarrow 0$
  5.2. else $b_i \leftarrow 1$
  5.3. end if
6. end for
7. return $\mathbf{b}$

The proposed modification is tested with several different nonlinear transformations, including ReLU, Leaky ReLU, tanh and a custom function with sine component (2), which is denoted as f_{custom} in this work.

$$\text{LeakyReLU}(x)=\begin{cases}ax & \text{if} \quad x<0;\\ x & \text{if} \quad x\geq 0,\end{cases}$$
$$\tanh(x)=\frac{e^x-e^{-x}}{e^x+e^{-x}},$$
$$f_{\text{custom}}(x)=ax+b\,|\,x\,|\sin x. \qquad (2)$$

An additional modification is added to MLP-hash as a generalization of spaces augmentation from [5], which increases dimensionality of biohashing results by using several projection spaces for one input vector and concatenating projected vectors. In this work this principle is used for each layer by replacing layer matrices of size $m_i \times m_{i-1}$ with p_i orthonormal matrices of size $m_i \times m_{i-1}p_{i-1}$ (note that dimensionality of $\mathbf{y}_{i-1}$ increases as well), multiplying $\mathbf{y}_{i-1}$ by each of them and concatenating results into vector of size $m_i \times p_i$. Note that with this modification being implemented spaces augmentation can still be removed from some layers if unnecessary by just setting the corresponding p_i to 1.This modification can also be extended to simplified projections proposed in this work by generating $b_i p_i$ blocks of size $\frac{m_i}{b_i}\times\frac{m_{i-1}p_i}{b_i}$ per each layer matrix instead of b_i, multiplying each $\mathbf{y}_{i-1,\text{ block } j}$ by p_i of these blocks and concatenating resulting vectors. In this work this modification is used only to increase layer sizes for computational speed testing.

4 EXPERIMENTS

The developed method is tested on feature vectors extracted from fingerprint images from the FVC2000 dataset [8]. This dataset contains fingerprints of 10 people, 8 images per fingerprint. These images are grayscale with brightness in range [0,255] and have 300×300 pixels.

Feature vectors are extracted from fingerprint images using a bank of Gabor filters by method similar to the one proposed in [9].

First of all, a pivot point is chosen. In this work it is a singularity point. Singularity point detection is based on

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16
OPEN ACCESS

an orientation field [10], which indicates ridge directions. Before the orientation field computation image is blurred using a Gaussian filter to reduce the effect of noise. Then, image gradient is estimated using Sobel filters [11] of the following form:

$$\mathbf{F}_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, \quad \mathbf{F}_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix},$$

producing gradient estimations $\mathbf{G}_x$ and $\mathbf{G}_y$ along the x and y axes respectively. These are used for computing components of orientation field within non-overlapping windows of size $w \times w$:

$$\mathrm{v}_x(i,j) = 2\sum_{u=iw}^{iw+w-1}\sum_{v=jw}^{jw+w-1}\mathbf{G}_x(u,v)\cdot\mathbf{G}_y(u,v),$$

$$\mathrm{v}_y(i,j) = \sum_{u=iw}^{iw+w-1}\sum_{v=jw}^{jw+w-1}(\mathbf{G}_x(u,v)^2 - \mathbf{G}_y(u,v)^2).$$

These components are then blurred using a Gaussian filter and then orientation is computed as:

$$\theta(i,j) = \frac{1}{2}\arctan\left(\frac{\mathrm{v}_x(i,j)}{\mathrm{v}_y(i,j)}\right).$$

Another value computed from image gradient is the gradient magnitude:

$$\mathrm{M}(i,j) = \sum_{u=iw}^{iw+w-1}\sum_{v=jw}^{jw+w-1}(\mathbf{G}_x(u,v)^2 + \mathbf{G}_y(u,v)^2).$$

It can be used as an estimation of image quality.

Singular points are detected with the use of Poincare index calculated on orientation field. To get its value at some position, 8 positions around it are numbered counterclockwise as $\theta_k(i,j)$, $k = 1, 2, ..., 8$. Differences between them are calculated as $\delta_k(i,j) = \theta_{k+1}(i,j) - \theta_k(i,j)$ for $k = 1, ..., 7$, $\delta_8(i,j) = \theta_1(i,j) - \theta_8(i,j)$, and changed to choose the smaller angle between directions (since directions of fingerprint ridges are ambiguous and may be changed by $\pm\pi$ without changing their meaning):

$$\Delta_k(i,j) = \begin{cases} \delta_k(i,j) & \text{if } |\delta_k(i,j)| < \frac{\pi}{2}; \\ \pi + \delta_k(i,j) & \text{if } \delta_k(i,j) \le -\frac{\pi}{2}; \\ \pi - \delta_k(i,j) & \text{otherwise.} \end{cases}$$

Poincare index is computed from these differences as:

$$\mathrm{P}(i,j) = \frac{1}{2}\sum_{k=1}^{8}\Delta_k(i,j).$$

For closed curves (and closed 8-connected neighbourhood) Poincare index takes one of these values: $-\frac{1}{2}, 0, \frac{1}{2}, 1$, with values different from 0 corresponding to singularity points.

If several singularity points are detected, the pivot point is determined by their average, weighted by gradient magnitude, so that singular points detected in regions with worse quality (which have a higher chance of being wrong) contribute less to the resulting point. If no singular points are detected, the image is discarded.

After the pivot point is chosen, a region of interest is selected. In this work ROI is similar to that from [9] and is bounded by circle of radius r_{ROI} centered at the pivot point and separated into sectors by n_c concentric circles having radiuses $r_{\mathrm{inner}}, r_{\mathrm{inner}} + \frac{r_{\mathrm{ROI}} - r_{\mathrm{inner}}}{n_c - 1}, ..., r_{\mathrm{ROI}}$ (region bound by r_{inner} is not used because it is highly sensitive to changes in pivot position) and by n_d directions defined by angles $0, \frac{2\pi}{n_d}, ..., \frac{2\pi(n_d-1)}{n_d}$, partitioning ROI into $(n_c - 1)n_d$ ring sectors in total.

Gabor filters are matrix filters and they are used to extract texture features from images. Filters used in this work are the real parts of Gabor filters and have the form:

$$\mathbf{G}(x,y;\lambda,\theta,\sigma,\gamma,\psi) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)\cos\left(\frac{2\pi x'}{\lambda} + \psi\right),$$

where $x' = x\cos\theta + y\sin\theta$, $y' = -x\sin\theta + y\cos\theta$. In this work, $\gamma = 1$ (so that gaussian component is radially symmetric), $\psi = 0$ (so that cosine wave peaks at line crossing the filter center), $\sigma = 5$ (chosen empirically), $\lambda = 10$ (chosen empirically to roughly match waves formed by ridges) and θ takes n_g different values to form a Gabor filter bank: $0, \frac{\pi}{n_g}, ..., \frac{\pi(n_g-1)}{n_g}$.

Before Gabor filters are applied an image is normalized so that the mean pixel value is 0 and standard deviation is 1. Then, filters are applied one at a time. For each filter direction θ and each ring sector of ROI an average absolute deviation from mean is calculated as:

$$V_{i,\theta} = \frac{1}{n_i}\left(\sum_{n_i}\left|\mathrm{F}_{i,\theta}(x,y) - P_{i,\theta}\right|\right).$$

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16
OPEN ACCESS

AAD is considered 0 if the sector is located outside of image. The feature vector of a fingerprint consists of these AAD values and have length $(n_c - 1)n_d n_g$.

In this work, $n_c = 5$, $n_d = 8$ and $n_g = 8$, therefore the feature vector has length 256.

MLP-hash with normalizations and MLP-hash with both normalizations and simplified projection matrices are implemented and tested with different normalization methods (min-max, l_2, statistical and without normalization) and different nonlinear transformations (ReLU, leaky ReLU, tanh, f_{custom}). Original MLP-hash is implicitly included as MLP-hash without normalization, without matrix simplification and with ReLU. For comparison, original biohashing is also implemented.

All biohashing variations are tested with a classifier based on Hamming distance. Hamming distance H(**a**, **b**) is the amount of differing elements. Classifier is built such that if H(**a**, **b**) $\le T$, where T is a threshold, **a** and **b** are considered as those from the same user, else they are considered coming from different users. To see how much distinction between templates is lost from the data transformation, another classifier based on cosine similarity S is implemented.

$$\mathrm{S}(\mathbf{a},\mathbf{b}) = \frac{\mathbf{a}\cdot\mathbf{b}}{|\mathbf{a}|\cdot|\mathbf{b}|}.$$

If S(**a**, **b**) $\ge T$, where T is a threshold, vectors **a**, **b** are considered belonging to the same user, else – to different. This classifier is used with unprotected feature vectors.

The metrics being used in this work are FRR, FAR and EER. FRR, or false rejection rate, is the fraction of genuine user authentication attempts that are rejected:

$$\mathrm{FRR} = \frac{N_{fr}}{N_{fr} + N_{ta}}.$$

FAR, or false acceptance rate, is the fraction of imposter authentication attempts that are considered genuine by classifier:

$$\mathrm{FAR} = \frac{N_{fa}}{N_{fa} + N_{tr}}.$$

EER, or equal error rate, is considered equal to FAR when FAR is equal to FRR.

Because there may be no point at which FRR precisely equals FAR for a finite dataset, EER is approximated based on the closest FRR and FAR values. Suppose that n_T parametrisations of the classifier are used, for example, different threshold values T. For these values FRR and FAR pairs are calculated and sorted by descending FRR, forming a sorted list of pairs $(\mathrm{FRR}_i, \mathrm{FAR}_i)$, $i = 1, ..., n_T$. Let i_c be the smallest index at which FRR < FAR. Then EER is approximated as:

$$\mathrm{EER} = \frac{\mathrm{FRR}_{i_c-1} + \mathrm{FAR}_{i_c-1} + \mathrm{FRR}_{i_c} + \mathrm{FAR}_{i_c}}{4}.$$

Biohashing and its modifications are probed in two scenarios: base scenario and stolen token scenario. Base scenario implies that every user has a different secret token and keeps its secrecy, while stolen token scenario implies that users token has lost its secrecy and is used by everyone.

5 RESULTS

FAR, FRR and EER are calculated for the following parameters of MLP-hash modifications: $l = 3$, $m_i = 128$ for all layers. For the modification with matrix simplification, $b = 8$.

Features extracted with the use of Gabor filters are used in the classifier based on cosine similarity. The plot of FRR against FAR for this classifier is presented on Fig. 1. The EER of this classifier is 0.1685.

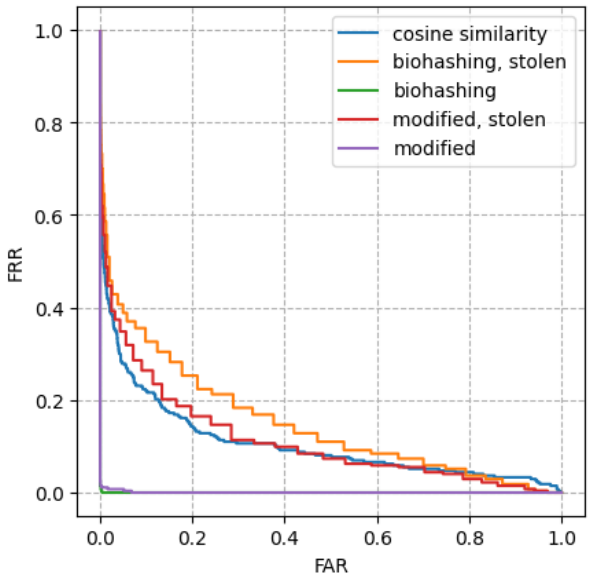

Figure 1 – Plot of FRR versus FAR of classifiers based on unprotected biometric templates (by cosine similarity) and of classifiers based on templates protected by biohashing and its modification proposed in this work, where "stolen" refers to stolen token scenario

The results of applying base biohashing are used in a classifier based on Hamming distance. Plots of FRR against FAR for this classifier are presented on Fig. 1 for comparison with these for unprotected templates. The EER of this classifier is 0.0037 in base scenario and 0.2234 in stolen token scenario.

The results of applying MLP-hash with inclusion of normalization are used in a classifier based on Hamming distance. EER values for base scenario are listed in Table 1, for stolen token scenario – in Table 2.

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16

Table 1 – EER values of classifier based on Hamming distance with MLP-hash with normalizations in base scenario

	ReLU	leaky ReLU	tanh	f_{custom}
None	0.001450	0.001272	0.010925	0.007313
min-max	0.000089	0.001272	0.001450	0.004884
l_2	0.001450	0.001272	0.001450	0.001272
statistical	0.128294	0.078907	0.089744	0.073222

Table 2 – EER values of classifier based on Hamming distance with MLP-hash with normalizations in stolen token scenario

	ReLU	leaky ReLU	tanh	f_{custom}
None	0.260111	0.208829	0.275921	0.223456
min-max	0.305886	0.295050	0.283323	0.310211
l_2	0.260111	0.208829	0.209834	0.204212
statistical	0.223456	0.213624	0.219844	0.196276

The results of applying MLP-hash with inclusion of normalization and with simplified projections are used in a classifier based on Hamming distance. EER values for base scenario are listed in Table 3, for stolen token scenario – in Table 4. Plots of FRR and FAR for a Hamming distance classifier with the use of the modified method with f_{custom} and statistical normalization are presented on Fig. 1 for comparison with unprotected templates and base biohashing.

Table 3 – EER values of classifier based on Hamming distance with MLP-hash with simplified projections in stolen token scenario

	ReLU	leaky ReLU	tanh	f_{custom}
None	0.000089	0.015898	0.014716	0.014716
min-max	0.054983	0.029342	0.021940	0.016433
l_2	0.000089	0.015898	0.004706	0.010925
statistical	0.001628	0.012999	0.018328	0.010925

Table 4 – EER values of classifier based on Hamming distance with MLP-hash with simplified projections in stolen token scenario

	ReLU	leaky ReLU	tanh	f_{custom}
None	0.276455	0.208829	0.204034	0.211259
min-max	0.378307	0.296767	0.271126	0.241695
l_2	0.276455	0.208829	0.190413	0.190413
statistical	0.254960	0.212441	0.183188	0.186800

For a modified method with the usage of f_{custom} as a nonlinear transformation and statistical normalization, distances between templates of genuine users each having his own token, of different users each having his own token, of genuine users but each time with new token and of different users who use a new token each time are calculated. Histogram of these distances is presented on Fig. 2.

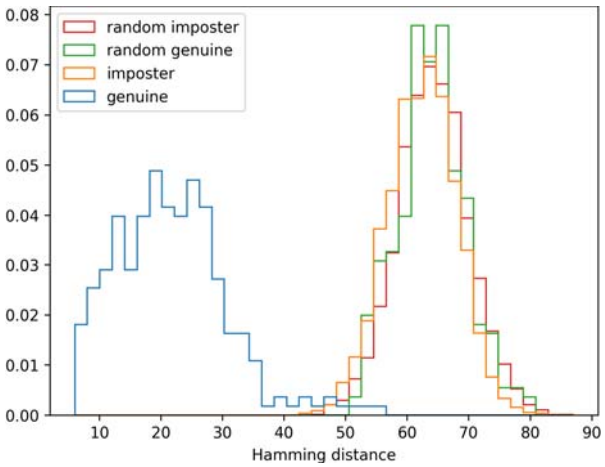

Figure 2 – Histograms of distances between protected templates, where "random" corresponds to choosing a new token each time

In order to see the speed improvement for different layer and block sizes, the amount of projections for the first layer is set as $p_1 = n_p$, which multiplies its output size by n_p as well, and other layers output size is multiplied by n_p. Computation durations of a modified MLP-hash for different parametrisations are listed in Table 5.

Table 5 – Computation time of modified MLP-hash in seconds. Each column corresponds to certain value of n_p, each row – to certain value of b

	1	2	4	8	16
1	0.009750	0.017471	0.107659	1.512096	7.540924
2	0.004071	0.009447	0.037941	0.198579	1.952194
4	0.003696	0.007272	0.019800	0.056675	0.395279
8	0.003670	0.006958	0.014189	0.043949	0.118840
16	0.003275	0.006494	0.013236	0.027612	0.066130

6 DISCUSSION

As can be seen from Table 2 and Table 4, applying l_2-normalization or statistical normalization reduces EER in most cases. From the same tables it can be seen that using leaky ReLU, tanh or f_{custom} leads to smaller EER in most cases. While using tanh or f_{custom}, using simplified projections leads to smaller EER. However, this improvement is not present while using ReLU or Leaky ReLU. In the stolen token scenario, the most optimal parametrisation without simplifying matrices (statistical normalization, f_{custom}) has 1.33 times smaller EER than with original parametrisation (no normalization, ReLU) and 1.14 times smaller EER than with base biohashing. In this same scenario, the most optimal parametrisation with simplified matrices (statistical normalization, f_{custom}) has 1.05 times smaller EER than with same parametrisation without matrix simplification, or 1.19 times smaller than with base biohashing.

From the comparisons above it can be concluded that applying normalization improves EER. Simplification of projection matrices also leads to improvement of EER in some cases, although it is less noticeable and the main advantage of it is increasing computation speed.

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16
OPEN ACCESS
CC BY SA

The proposed modification complies with requirements for biometric template protection methods listed in problem statement:

1) non-reversibility: without matrix simplification this method is at least as difficult to partially invert as base biohashing and MLP-hash;

2) accuracy: the modified method does not lead to increased EER in comparison to base biohashing, furthermore, it leads to improvement and it is closer to EER of classifier based on unprotected feature vectors (1.11 times bigger) than biohashing is (1.33 times bigger);

3) diversity: as it can be seen from Fig. 2, the distributions of distances between templates of different users is similar to that of same user but with different tokens, which means that if user creates several templates with different tokens, it will be difficult to decide if these templates are from one user or from several;

4) revocability: a protected biometric template can be replaced by replacing users' secret token t, in the same way as it can be replaced while using biohashing or MLP-hash.

As it can be seen from Table 5, the modified method is faster than non-modified (for n_p = 1 it is 2.39 times faster with just b = 2), and the difference in speed is increasing with increasing block number (for n_p = 1 and b = 16 modified method is 2.98 times faster) and it is more significant for bigger layer sizes (for n_p = 16, which corresponds to layers of size 4096, and b = 16 modified method is 114.03 times faster). It should be kept in mind that this modification may make inversion easier and that an analysis of inversion of MLP-hash (and of the modification presented in this work) was not performed, therefore setting a block number to high values may make the biometric system less safe.

CONCLUSIONS

This work is focused on modifying a nonlinear biohashing-based biometric template protection method, further generalizing and improving it.

The scientific novelty of this work is a modification of the MLP-hash method including normalization between layers of MLP-hash and simplification of pseudorandom projection matrices. Results of experiments conducted with feature vectors extracted from fingerprints show that applying l_2 or statistical normalization improves EER of the classifier using protected templates (for example, with statistical normalization and f_{custom} EER is 1.14 times smaller than without normalization and with f_{custom} and 1.33 times smaller than without normalization and with ReLU). Because the classifier based on templates protected by the modified method is more accurate than a classifier based on templates protected by base MLP-hash and biohashing, it can be said that the modified method preserves more distinction between templates. The method is shown to have all properties required from biometric template protection methods. The main advantage of simplifying projection matrices is the decrease of time complexity of the method from $O(m^2n)$ to $O\left(\frac{m^2n}{b^2}+\frac{mn}{b}+n\right)$ and space complexity from $O(mn)$ to $O\left(\frac{mn}{b}+m+n\ln n\right)$. Time complexity difference is also shown by experimental results, with the modified method being 2.39 times faster for n_p = 1, b = 2 and 114.03 times faster for n_p = 16, b = 16. Projection simplification also causes a small decrease in EER of a corresponding classifier: 1.05 times smaller while both simplified and non-simplified methods use statistical normalization and f_{custom} or 1.39 times smaller in comparison to base MLP-hash.

The practical significance of obtained results is that the modified method is faster and causes less classifier performance loss than the original. It can be used in authentication systems with high-dimensional feature vectors when users expect an absence of delays, or it can be used with hardware having low computational power and memory. The developed method produces protected templates that are similar to unprotected in overall structure, especially if unprotected templates are binarized before usage, so they can be used in the same applications in similar ways. These applications include authentication, identification, cryptographic key generation or binding (this key may be further transformed to match the system it is being used, for example by being extended or shortened to a bit string of a needed length, integer in some specific range for a cryptographic key exchange based on elliptic curves over finite fields, basis for lattice based cryptography etc.).

Prospects for further research include investigating even more nonlinear transformations and normalization methods, usage of the modified method with other biometric modalities and feature extraction techniques. Invertibility of the original MLP-hash and an effect nonlinear function choice has on it should also be studied, along with the decreasing of inversion difficulty coming from the simplification of projection matrices, in order to determine which parametrisations allow saving computation time without noticeable drop in safety.

ACKNOWLEDGEMENTS

The study was performed without financial support.

DECLARATIONS

Conflict of interest: The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship, or otherwise, that could affect the research and its results presented in this paper.

Authors' contributions: Mykola Onai: conceptualization, methodology, formulation of tasks, software, analysis of research results, formulation of conclusions, review, and editing; Oleksandr Kosenko: conceptualization, methodology, formulation of tasks, software, analysis of research results, formulation of conclusions, review, and editing.

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16

Data availability: The manuscript has no associated data.

Software availability: The manuscript has associated software in a public repository: https://github.com/KosenkoAlexander/modified-MLP-hash.

Use of artificial intelligence tools: The authors confirm that they did not use artificial intelligence technologies in creating the submitted work.

REFERENCES

1. Maltoni D., Maio D., Jain A. K., Prabhakar S. Handbook of Fingerprint Recognition (2nd. ed.). London, Springer, 2009, 494 p. DOI: 10.1007/978-1-84882-254-2
2. Teoh A., Ngo D., Goh A. Biohashing: two factor authentication featuring fingerprint data and tokenised random number, *Pattern Recognition*, 2004, Vol. 37, Issue 11, pp. 2245–2255. DOI: 10.1016/j.patcog.2004.04.011
3. Baier H., Breitinger F., Busch C., Rathgeb C. On application of bloom filters to iris biometrics, *IET Biometrics,* 2014, Vol. 3, Issue 4, pp. 207–218. DOI: 10.1049/iet-bmt.2013.0049
4. Jin Z., Hwang J. Y., Lai Y., Kim S., Teoh A. Ranking-Based Locality Sensitive Hashing-Enabled Cancelable Biometrics: Index-of-Max Hashing, *IEEE Transactions on Information Forensics and Security,* 2018, Vol. 13, Issue 2, pp. 393–407. DOI: 10.1109/TIFS.2017.2753172
5. Lumini A. and Nanni L. An improved BioHashing for human authentication, *Pattern Recognition*, 2007, Vol. 40, Issue 3, pp. 1057–1065. DOI: 10.1016/j.patcog.2006.05.030
6. Durbet A., Grollemund P., Lafourcade P., Migdal D., Thiry-Atighehchi K. Authentication Attacks on Projection-based Cancelable Biometric Schemes, *International Conference on Security and Cryptography (SECRYPT) : 19th International Conference, Lisbon, 11–13 July, 2022 : proceedings*. SCITEPRESS Digital Library, 2022, pp. 568–573. DOI: 10.5220/0011277100003283
7. Otroshi H. S. and Krivokuća V. H. and Marcel S. MLP-Hash: Protecting Face Templates via Hashing of Randomized Multi-Layer Perceptron, *European Signal Processing Conference (EUSIPCO) : 31st European Conference, Helsinki, 4–8 September, 2023 : proceedings.* – Helsinki, IEEE, 2023, pp. 605–609. DOI: 10.23919/EUSIPCO58844.2023.10289780
8. Maio D., Maltoni D., Cappelli R., Wayman J., Jain A. K. FVC2000: Fingerprint verification competition, *Pattern Analysis and Machine Intelligence,* 2002, Vol. 24, Issue 3, pp. 402–412. DOI: 10.1109/34.990140
9. Jain A. K., Prabhakar S., Hong L., Pankanti S. Filterbank-based fingerprint matching, *IEEE transactions on image processing,* 2000, Vol. 9, Issue 5, pp. 846–859. DOI: 10.1109/83.841531
10. Kawagoe M., Tojo A.Fingerprint pattern classification, *Pattern Recognition*, 1984, Vol. 17, Issue 3, pp. 295–303. DOI: 10.1016/0031-3203(84)90079-7
11. Duda R. O., Hart P. E. Pattern classification and scene analysis. New York, Wiley, 1978, 512 p.

Received 31.07.2025.
Accepted 08.01.2026.
Published 27.03.2026.

УДК 004.056.55

МОДИФІКОВАНИЙ МЕТОД ЗАХИСТУ БІОМЕТРИЧНИХ ШАБЛОНІВ З НЕЛІНІЙНИМИ ПЕРЕТВОРЕННЯМИ

Онай М. В. – канд. техн. наук, доцент кафедри програмного забезпечення комп'ютерних систем Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна. ROR: https://ror.org/00syn5v21. ORCID: https://orcid.org/0000-0002-4938-8355.

Косенко О. В. – бакалавр кафедри програмного забезпечення комп'ютерних систем Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна. ROR: https://ror.org/00syn5v21. ORCID: https://orcid.org/0009-0001-2115-7918.

АНОТАЦІЯ

Актуальність. Біометричні дані нерідко використовуються для аутентифікації або ідентифікації. Однак такі дані є вразливими і не є замінюваними у разі викрадення. У літературних джерелах запропоновано кілька методів створення захищених біометричних шаблонів, одним з яких є біогешинг. Однак лінійність біогешинга може бути його вразливістю. MLP-hash є схожим методом, що додає нелінійні перетворення. Цей метод модифікується у даній роботі.

Мета роботи. Метою даної роботи є розроблення модифікації MLP-hash, що є швидшою за оригінал та дозволяє більш чітке розділення користувачів за їх шаблонами.

Метод. Дана робота зосереджена на модифікуванні MLP-hash, варіації біогешингу з нелінійними перетвореннями. однією з запропонованих змін є застосування нормалізації перед застосуванням нелінійного перетворення у кожному шарі MLP-hash. Іншою запропонованою зміною є спрощення псевдовипадкових матриць, що використовуються у кожному шарі MLP-hash. Кожна така матриця замінюється на блочну матрицю, у якій блоки, що лежать на діагоналі, є ортонормальними матрицями, а решта блоків заповнюються нулями. Кожен ненульовий блок генерується з використанням користувацького секретного токена. Для того щоб зробити вплив кожного ненульового блоку менш локалізованим, перед кожним множенням на матрицю та після всіх шарів додаються псевдовипадкові перестановки. Псевдовипадкові перестановки також генеруються з використанням користувацького секретного токена у якості сіда. Застосування запропонованого методу близьке до застосування оригінального MLP-hash та біогешинга: метод приймає користувацький секретний токерн та біометричний вектор фіксованої довжини та повертає бінарний вектор

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16

фіксованої довжини з такою ж або меншою розмірністю. MLP-hash з блоковими матрицями порівняно з оригіналом при застосуванні різноманітних способів нормалізації та різноманітних нелінійних перетворень.

Результати. Програмно реалізовано запропоновану модифікацію, оригінальний MLP-hash та біогешинг. Порівняно швидкодію та точність розділення користувачів з використанням цих методів на векторах характеристик, виділених з відбитків пальців з використанням фільтрів Габора.

Висновки. Проведені експерименти показали збільшення швидкодії та здатності розділення користувацьких шаблонів у результаті підстановки запропонованих блочних матриць та підвищення здатності розділення користувацьких шаблонів у результаті застосування нормалізації. Крім того, проведено порівняння застосування різних методів нормалізації та різних нелінійних перетворень. Практична цінність розробленого методу полягає в тому, що він є швидшим та може бути використаний у складі програмного забезпечення, користувачі якого очкують на відсутність затримок, зберігаючи при цьому складність інвертування. Перспективи подальших досліджень включають тестування розробленого методу з іншими біометричними модальностями, іншими нелінійними перетвореннями та техніками нормалізації та аналіз складності інвертування розробленого методу у порівнянні з MLP-hash та біогешингом.

КЛЮЧОВІ СЛОВА: біометрія, біометричний шаблон, захист біометричних шаблонів, одностороннє перетворення, біогешинг, еліптична криптографія, скінченні поля.

ЛІТЕРАТУРА

1. Handbook of Fingerprint Recognition (2nd. ed.) / [D. Maltoni, D. Maio, A. K. Jain, S. Prabhakar]. – London : Springer, 2009. – 494 p. DOI: 10.1007/978-1-84882-254-2
2. Teoh A. Biohashing: two factor authentication featuring fingerprint data and tokenised random number / A. Teoh, D. Ngo, A. Goh // Pattern Recognition. – 2004. – Vol. 37, Issue 11. – P. 2245–2255. DOI: 10.1016/j.patcog.2004.04.011
3. On application of bloom filters to iris biometrics / [H. Baier, F. Breitinger, C. Busch, C. Rathgeb] // IET Biometrics. – 2014. – Vol. 3, Issue 4. – P. 207–218. DOI: 10.1049/iet-bmt.2013.0049
4. Ranking-Based Locality Sensitive Hashing-Enabled Cancelable Biometrics: Index-of-Max Hashing / [Z. Jin, J. Y. Hwang, Y. Lai et al.] // IEEE Transactions on Information Forensics and Security. – 2018. – Vol. 13, Issue 2. – P. 393–407. DOI: 10.1109/TIFS.2017.2753172
5. Lumini A. An improved BioHashing for human authentication / A. Lumini and L. Nanni // Pattern Recognition. – 2007. – Vol. 40, Issue 3. – P. 1057–1065. DOI: 10.1016/j.patcog.2006.05.030
6. Authentication Attacks on Projection-based Cancelable Biometric Schemes / [A. Durbet, P. Grollemund, P. Lafourcade et al.] // International Conference on Security and Cryptography (SECRYPT) : 19th International Conference, Lisbon, 11–13 July, 2022 : proceedings. – SCITEPRESS Digital Library, 2022. – P. 568–573. DOI: 10.5220/0011277100003283
7. Otroshi H. S. MLP-Hash: Protecting Face Templates via Hashing of Randomized Multi-Layer Perceptron / H. S. Otroshi and V. H. Krivokuća and S. Marcel // European Signal Processing Conference (EUSIPCO) : 31st European Conference, Helsinki, 4–8 September, 2023 : proceedings. – Helsinki, IEEE, 2023. – P. 605–609. DOI: 10.23919/EUSIPCO58844.2023.10289780
8. FVC2000: Fingerprint verification competition / [D. Maio, D. Maltoni, R. Cappelli et al.] // Pattern Analysis and Machine Intelligence. – 2002. – Vol. 24, Issue 3. – P. 402–412. DOI: 10.1109/34.990140
9. Filterbank-based fingerprint matching / [A. K. Jain, S. Prabhakar, L. Hong, S. Pankanti] // IEEE transactions on image processing. – 2000. – Vol. 9, Issue 5. – P. 846–859. DOI: 10.1109/83.841531
10. Kawagoe M. Fingerprint pattern classification / M. Kawagoe, A. Tojo // Pattern Recognition. – 1984. – Vol. 17, Issue 3. – P. 295–303. DOI: 10.1016/0031-3203(84)90079-7
11. Duda R. O. Pattern classification and scene analysis / R. O. Duda, P. E. Hart. – New York : Wiley, 1978. – 512 p.

© Onai M. V., Kosenko O. V., 2026
DOI 10.15588/1607-3274-2026-1-16