DEEP AUTO-ENCODING AND BIOHASHING FOR SECURE FINGER VEIN RECOGNITION

Hatef Otroshi Shahreza and Sébastien Marcel

Idiap Research Institute, Switzerland

ABSTRACT

Biometric recognition systems relying on finger vein have gained a lot of attention in recent years. Besides security, the privacy of finger vein recognition systems is always a crucial concern. To address the privacy concerns, several biometric template protection (BTP) schemes are introduced in the literature. However, despite providing privacy, BTP algorithms often affect the recognition performance. In this paper, we propose a deep-learning-based approach for secure finger vein recognition. We use a convolutional auto-encoder neural network with a multi-term loss function. In addition to the auto-encoder loss function, we deploy triplet loss for the embedding features. Next, we apply Biohashing to our deep features to generate protected templates. The experimental results indicate that the proposed method achieves superior performance to previous finger vein recognition methods protected with Biohashing. Besides, our proposed method has less execution time and requires less memory.[1]

Index Terms— Auto-encoder, Biohashing, convolutional neural network, deep learning, finger vein recognition, template protection.

1. INTRODUCTION

Biometric recognition systems are growing and widely used in different applications for authentication purposes. In contrast to the conventional authentication tools such as PIN or password, which are always in danger of being forgotten or stolen, biometric authentication offers excellent convenience for the user. Meanwhile, a very crucial concern in biometric systems is privacy. It is mainly because if a biometric template is compromised, it can not be revoked or changed [1]. To address this challenge, many biometric template protection (BTP) schemes are introduced in the literature. Although BTP algorithms provide privacy for the biometric system, they affect the recognition performance.

In finger vein biometric systems, the recognition relies on vascular patterns which are formed by blood vessels of a human finger. Since these patterns have adequate traits, they are used for the purpose of automatic individual recognition. Several methods have been proposed in the literature to extract such features from finger vein images. Most of these methods [2, 3, 4, 5, 6, 7] extract the binary vessel structure and then compare the extracted templates using the Miura algorithm [4]. Figure 1 shows two sample finger vein images and their corresponding Wide Line Detector (WLD) [2], Repeated Line Tracking (RLT) [3], and Maximum Curvature (MC) [4] features. Besides these methods, some other methods are recently proposed which use deep neural networks [8, 9, 10]. Nevertheless, these methods are vulnerable to direct and indirect attacks[11].

Additional works try to increase protection of finger vein recognition (FVR) systems. In [13], authors explore the effect of Biohashing template protection algorithm [14] on the performance of a FVR system using WLD, RLT, and MC feature extractors. In [15], authors use Index-of-Maximum (IoM) [16] hashing to propose an alignment-free template protection scheme. In [17], authors use a user-specific random projection on the extracted biometric features to reduce the features dimension and generate protected templates. Then, they train

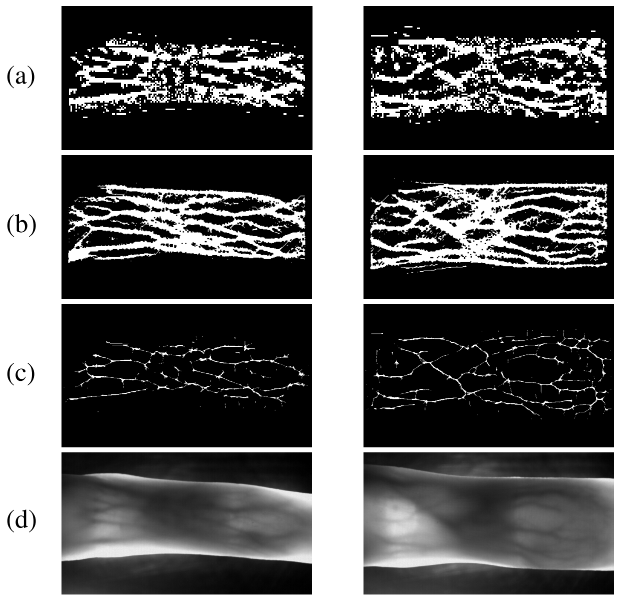


Fig. 1: Two sample finger vein images from UTFVP dataset [12] and their corresponding WLD, RLT, and MC features for two individuals in the UTFVP dataset: (a) WLD, (b) RLT, (c) MC, (d) Finger vein image.

This research is based upon work supported by the H2020 TReSPAsS-ETN Marie Skodowska-Curie early training network (grant agreement 860813). Authors are with the Biometrics Security and Privacy Group of Idiap Research Institute, Martigny, Switzerland. Hatef Otroshi Shahreza (hatef.otroshi@epfl.ch) is also affiliated with École Polytechnique Fédérale de Lausanne (EPFL).

[1] Source code: https://gitlab.idiap.ch/bob/bob.paper.icassp2021.deepae_biohashing_securefvr

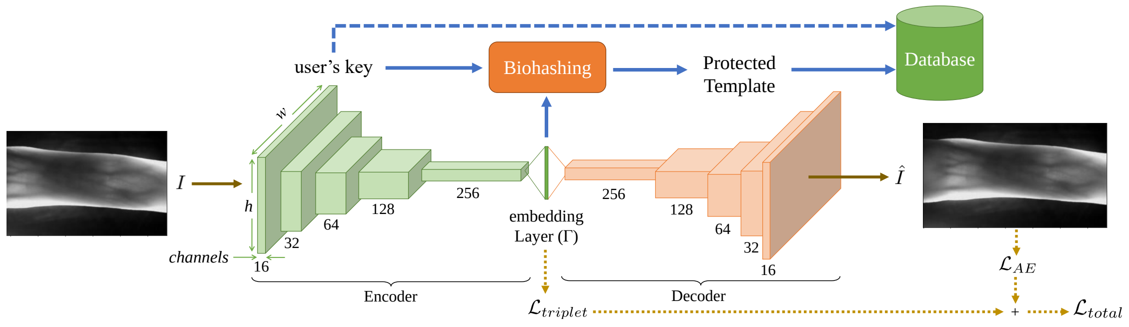


Fig. 2: Block-diagram of the proposed method

a deep belief networks to match the protected templates. In [18], authors use Biohashing for the extracted features, and then apply a binary transformation on the Biohashing output. Finally, the results are given to a multilayer extreme learning machine (ML-ELM) for training and classification.

In this paper, we propose a novel deep-learning-based approach for secure FVR. As depicted in figure 2, we use a convolutional neural network (CNN) with an auto-encoder (AE) structure. We train the proposed CNN with a multi-term loss function, including an auto-encoder reconstruction loss and a triplet loss for the bottleneck (embedding) layer. Then, we use the features at the embedding layer and apply Biohashing [14] to generate protected templates. To our knowledge this approach is original and it was never proposed to reduce the dimensionality of finger vein images with a deep AE prior standard Biohashing, hence avoiding the enormous dimensionality reduction gap achieved by applying directly Biohashing to pre-processed images. It is worth mentioning that Biohashing relies on a random projection matrix, which is often referred to as *key*. In practice, the seed of the matrix generation is used as the key. Hence, if the key is stolen, the performance of the system can be affected. Therefore, we define two scenarios to evaluate the performance of the protected templates: the *normal* scenario and the *stolen* scenario. In the normal scenario, the protected biometric templates are used without performing any attack (templates are not compromised). However, in the stolen scenario, we consider the situation when a genuine user's key is stolen, and it is used by an impostor to perform an attack. Experimental results indicate that in comparison to the previous FVR methods, which are secured with Biohashing, our method is superior both in the normal scenario and the stolen scenario. Furthermore, our model requires less memory to store the projection matrix.

The rest of the paper is organized as follows. Section 2 introduces the proposed method and the experimental results are provided in Section 3. Finally, the paper is concluded in Section 4.

2. PROPOSED METHOD

2.1. Overview

As explained in Section 1, we propose a deep-learning-based approach for FVR. Figure 2 illustrates the block diagram of the proposed method. As shown in this figure, we use a deep convolutional auto-encoder to extract features from finger vein images, which is explained in detail in section 2.2. Next, we use the Biohashing algorithm to generate protected templates. This algorithm is also explained in section 2.3. Finally, for the recognition stage, the Biohash templates should be compared, which is described in 2.4.

It is noteworthy that the proposed FVR method is cancellable as well, because Biohashing is cancellable. Indeed, a new protected template can be generated any time using Biohashing with a new key.

2.2. Auto-encoder

2.2.1. Network Structure

We use a convolutional auto-encoder that reduces the size of the input image by the encoder to generate the embedding layer, and then reconstruct the image by the decoder. The encoder network consists of five convolutional layers with 16, 32, 64, 128, 256 filters, respectively. We use 3×3 kernel with stride 2 in each layer, which divides the spatial size by factor 2. Additionally, we use Batch normalization [19] after each convolution operation. Finally, we use a fully connected layer to get the embedding layer. In this paper, we consider the embedding layer with length 100. For the decoder network, we use the transpose convolution layers. Except for the final layer, which has `sigmoid` function, we use the rectified linear unit (ReLU) for the other layers. In our experiments, we use the UTFVP dataset[12] which contains images with 672×380 resolution, and then get 100-length features from the embedding layer.

2.2.2. Multi-term Loss Function

To train the proposed network, we use a multi-term loss function. Let's consider $I, \hat{I}, \Gamma$ as the input image, the reconstructed image, and the embedding layer, respectively. The total loss is

$$\mathcal{L}_{total} = (1-\alpha)\mathcal{L}_{AE} + \alpha\mathcal{L}_{triplet}, \tag{1}$$

where α is a hyper-parameter (in $[0,1]$ interval) to control the contribution of $\mathcal{L}_{AE}$ and $\mathcal{L}_{triplet}$, where $\mathcal{L}_{AE}$ and $\mathcal{L}_{triplet}$ are the auto-encoder loss, and the embedding triplet loss, respectively. For the auto-encoder loss, we use the weighted sum of

l_1 and l_2 norm of the auto-encoder error which we empirically observed to produce a better accuracy than l_2 norm alone:

$$\mathcal{L}_{AE} = \|I - \hat{I}\|_2 + 0.5 \times \|I - \hat{I}\|_1 \tag{2}$$

Furthermore, the embedding triplet loss is defined as below:

$$\mathcal{L}_{triplet} = \texttt{sigmoid}\Big(\|\Gamma_a - \Gamma_p\|_2^2 - \|\Gamma_a - \Gamma_n\|_2^2\Big), \tag{3}$$

where Γ_a, Γ_p, and Γ_n are the values of embedding layer for anchor, positive, and negative images, respectively [20].

2.2.3. *Training Process*

To train the proposed auto-encoder with our multi-term loss function, we use Adam [21] optimizer. We use the initial learning rate of 10^{-3}, and decrease the learning rate every 10 epochs. We use the Pytorch framework for the experiments.

For our experiments, we use the UTFVP finger vein dataset [12] which contains 1440 finger vein images with 672×380 resolution that have been collected from 60 individuals. We apply data augmentation technique to the training set by randomly adjusting each finger vein image with a combination of the following transformations:

- rotation [range: < 7 degree]
- width shift [range: $< 0.025\times$ image width]
- height shift [range: $< 0.025\times$ image height]
- channel shift (i.e., offset) [range: < 0.075]
- zoom [range: (0.95,1.05)]

2.3. Biohashing algorithm for Template Protection

As mentioned earlier, we use the Biohashing algorithm [14] to generate the protected templates. Let's consider Γ, indicating an unprotected biometric template calculated at the embedding layer of our auto-encoder. Then, the protected template, B, can be generated by algorithm 1 using Γ and user's key, k. The Biohash templates, B, and the user's key should be eventually stored at the system database during enrollment.

2.4. Scoring and Comparing Biohash Templates

In the subsequent experiments, we will consider that FVR operated in verification mode only. In the enrolling stage, the protected templates for every individual are stored at the system database. For the verification stage, either verification or identification, the probe templates should be compared with the templates in the database. To find the score between the probe template and the model template, we use the Hamming distance between the Biohash templates.

3. EXPERIMENTAL RESULTS

3.1. Experiment Setup

As mentioned earlier, we used the publicly available finger vein UTFVP dataset [12] in our experiments. We used the training, development, and evaluation subsets of this dataset as described in [13]. The training subset is used for training our neural network, the development subset is used for threshold estimation, and the evaluation subset is used for reporting the final results in table 1 and table 2. We also implemented Wide Line Detector (WLD) [2], Repeated Line Tracking (RLT) [3], and Maximum Curvature (MC) [4] algorithms to extract biometric features from finger vein images, and then generate protected templates using Biohashing algorithm. We consider the Biohash-protected of WLD, RLT, and MC as the baselines in our experiments. Figure 1 shows two sample finger vein images and the corresponding WLD, RLT, and MC features for two individuals in the UTFVP dataset.

As mentioned in Section 1, we consider two scenarios in our experiments: the *normal* scenario and the *stolen* scenario. In the normal scenario, which is the expected scenario for most cases, each user's key is considered to be secret and not been disclosed. However, in the stolen scenario, the impostor has access to the genuine user's secret key and use it with the impostor's own finger vein template. While such a scenario is expected to happen rarely in practice, the system's vulnerability relies on the leakage of the user's secret key.

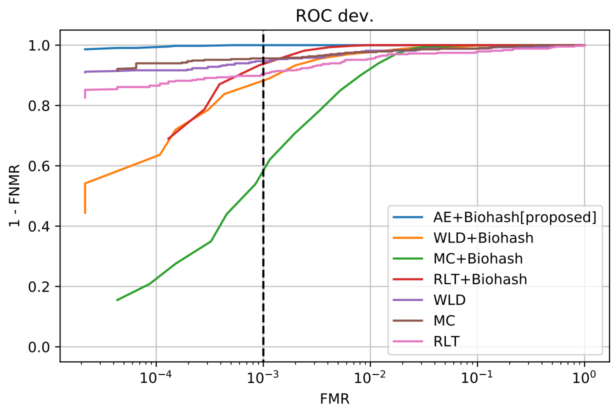
We should note that we use Bob[2] toolbox [22, 23] and the open-source implementation[3] of Biohash-protected finger vein verification systems in [13] for our experiments. In the following experiments, we have considered Biohashing with the length 100 for all models. In addition, we empirically fixed the hyper-parameter α in equation 2 equal with 0.1. It is worth mentioning that the models were evaluated on a system with an Intel i7-7700K 4.2 GHz CPU and an NVIDIA 1080 Ti GPU.

Algorithm 1 Biohashing algorithm for template protection

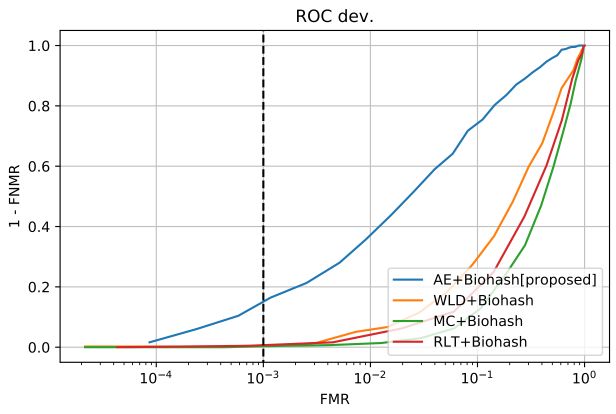
```
1:  Inputs:
2:      $\Gamma$ : unprotected biometric template
3:      $M$ : length of the unprotected template ($\Gamma$)
4:      $m$ : length of the protected template
5:      $k$ : user's seed
6:  Output: $B = \{b_i | i = 1, 2, ..., m\}$ binary BioHash protected template
7:  Procedure:
8:  Generate a set of pseudo-random vectors, $\{r_i \in \mathbb{R}^M | i = 1,2,...,m\}$, based on the user's seed, $k$.
9:  Apply the Gram-Schmidt process to transform the generated pseudo-random vectors $\{r_i \in \mathbb{R}^M | i = 1, ..., m\}$ into an orthonormal set of matrices $\{r_{\perp i} \in | i = 1,...,m\}$
10: Compute $\{\langle \Gamma, r_{\perp i}\rangle \in \mathbb{R} | i = 1,...,m\}$ where $\langle .,. \rangle$ indicates inner product operation.
11: Compute $m$ bits BioHash $\{b_i | i = 1,2,...,m\}$ from
        $b_i = \begin{cases} 0 & \text{if } \langle \Gamma, r_{\perp i}\rangle \le \tau \\ 1 & \text{if } \langle \Gamma, r_{\perp i}\rangle > \tau \end{cases}, i = 1,...,m,$
    where $\tau$ is a preset threshold.
12: End Procedure
```

[2] https://www.idiap.ch/software/bob/

[3] https://gitlab.idiap.ch/bob/bob.chapter.fingerveins_biohashing



(a)



(b)

Fig. 3: Comparison of ROC curves for our method and previous methods: (a) normal scenario, (b) stolen scenario.

3.2. Recognition Evaluation

Figure 3 compares the receiver operating characteristic (ROC) curves of our method (AE+Biohash) against the protected templates of WLD, RLT, and MC methods, namely WLD+Biohash, RLT+Biohash, and MC+Biohash, respectively, in normal and stolen scenarios on the development subset of the UTFVP dataset. Besides, in the normal scenario, we compare the ROC curve of our method to WLD, RLT, and MC methods alone without template protection. As shown in this figure, in the normal scenario, our method achieves superior performance than unprotected versions of WLD, RLT, and MC methods and also than their Biohash-protected versions. Similarly, in the stolen scenario, our method has far better performance than the Biohash-protected templates of WLD, RLT, and MC methods.

In addition to the ROC curve, we compare the performance of our method in terms of False Match Rate (FMR), False Non-Match Rate (FNMR), and Half Total Error Rate (HTER) for the evaluation subset of the UTFVP dataset, which is reported in the table 1. Please note that for the values in this table, the threshold for each method is selected individually in the way that we achieve minimum Equal Error Rate (EER) on the development subset. This table also confirms that our method achieves the best performance in both the normal and the stolen scenario. In the normal scenario, the error on test data is surprisingly zero for our method. In addition, in the stolen scenario, our proposed approach achieves far better performance than the compared methods.

Table 1: Comparing the performance of the proposed method with Biohashed-protected templates of previous methods in terms of FMR, FNMR, and HTER on the evaluation subset of UTFVP dataset. (Note that the best performance is **embolden**)

	Normal Scenario			Stolen Scenario		
method	FMR	FNMR	HTER	FMR	FNMR	HTER
[Proposed]	**0.0%**	**0.0%**	**0.0%**	**18.7%**	**16.7%**	**17.7%**
WLD + Biohash	1.4%	1.7%	1.5%	34.3%	44.0%	39.1%
RLT + Biohash	0.5%	0.7%	0.6%	46.8%	34.4%	40.6%
MC + Biohash	2.9%	2.3%	2.6%	42.1%	53.0%	47.5%

Table 2: The average execution time (second) and the required memory to store the projection matrix for generating the Biohash-protected templates

	WLD	**MC**	**RLT**	**AE [Proposed]**
Exe. Time	0.17	3.25	22.6	**0.06** (**0.004**)[*]
Memory	6.2 MB	106.4 MB	38.3 MB	**40.0 KB**

[*]The values are for the CPU (GPU) implementation.

3.3. Complexity and the Required Memory

In the table 2, we compare the complexity of the mentioned methods in terms of execution time and the required memory to store the projection matrix for generating Biohash-protected templates. In terms of execution time for extracting biometric features, our method achieves the best performance. In particular, in the GPU implementation, the proposed approach is quite fast. Besides, in terms of the required memory, our method needs less memory to store the projection matrix for the Biohashing algorithm. Less memory is mainly because the number of biometric features is less in our method; therefore, the projection matrix is smaller. However, we should note that for the feature calculation, our method requires to store the encoder part of our network which needs 27.5 MB RAM.

4. CONCLUSION

In this paper, we proposed a deep-learning-based approach for secure finger vein recognition. We trained a deep convolutional auto-encoder with a multi-term loss function. In addition to the auto-encoder loss function, we used a triplet loss for the embedding features. Next, we used the extracted features at the embedding layer and generated protected templates using Biohashing. The experimental results indicated that in the normal scenario and in the stolen scenario, our method achieves superior performance than regular finger vein recognition methods protected directly by Biohashing. Besides, our proposed method has less execution time and requires less memory.

5. REFERENCES

[1] MengHui Lim, Andrew Beng Jin Teoh, and Jaihie Kim, "Biometric feature-type transformation: Making templates compatible for secret protection," *IEEE Signal Processing Magazine*, vol. 32, no. 5, pp. 77–87, 2015.

[2] Beining Huang, Yanggang Dai, Rongfeng Li, Darun Tang, and Wenxin Li, "Finger-vein authentication based on wide line detector and pattern normalization," in *Proceedings of the 20th International Conference on Pattern Recognition (ICPR)*, Istanbul, Turkey, Aug. 2010, IEEE, pp. 1269–1272.

[3] Naoto Miura, Akio Nagasaka, and Takafumi Miyatake, "Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification," *Machine Vision and Applications*, vol. 15, no. 4, pp. 194–203, 2004.

[4] Naoto Miura, Akio Nagasaka, and Takafumi Miyatake, "Extraction of finger-vein patterns using maximum curvature points in image profiles," *IEICE TRANSACTIONS on Information and Systems*, vol. 90, no. 8, pp. 1185–1194, 2007.

[5] Joon Hwan Choi, Wonseok Song, Taejeong Kim, Seung-Rae Lee, and Hee Chan Kim, "Finger vein extraction using gradient normalization and principal curvature," in *Image Processing: Machine Vision Applications II*. International Society for Optics and Photonics, 2009, vol. 7251, p. 725111.

[6] Ajay Kumar and Yingbo Zhou, "Human identification using finger images," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 2228–2244, 2011.

[7] Jean-Luc Starck, Jalal Fadili, and Fionn Murtagh, "The undecimated wavelet decomposition and its reconstruction," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 297–309, 2007.

[8] Cihui Xie and Ajay Kumar, "Finger vein identification using convolutional neural network and supervised discrete hashing," in *Deep Learning for Biometrics*, pp. 109–132. Springer, 2017.

[9] Rig Das, Emanuela Piciucco, Emanuele Maiorana, and Patrizio Campisi, "Convolutional neural network for finger-vein-based biometric identification," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 2, pp. 360–373, 2018.

[10] Borui Hou and Ruqiang Yan, "Convolutional autoencoder model for finger-vein verification," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 5, pp. 2067–2074, 2019.

[11] Sébastien Marcel, Mark S Nixon, and Stan Z Li, *Handbook of biometric anti-spoofing*, Springer, 1 edition, 2014.

[12] B. T. Ton and R. N. J. Veldhuis, "A high quality finger vascular pattern dataset collected using a custom designed capturing device," in *Proceedings of the 2013 International Conference on Biometrics (ICB)*, Madrid, Spain, Jun. 2013, pp. 1–5.

[13] Vedrana Krivokuća and Sébastien Marcel, "On the recognition performance of biohash-protected finger vein templates," in *Handbook of Vascular Biometrics*, pp. 465–480. Springer, Cham, 2020.

[14] Andrew Teoh Beng Jin, David Ngo Chek Ling, and Alwyn Goh, "Biohashing: two factor authentication featuring fingerprint data and tokenised random number," *Pattern Recognition*, vol. 37, no. 11, pp. 2245–2255, 2004.

[15] Simon Kirchgasser, Christof Kauba, Yen-Lung Lai, Jin Zhe, and Andreas Uhl, "Finger vein template protection based on alignment-robust feature description and index-of-maximum hashing," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 2020.

[16] Zhe Jin, Jung Yeon Hwang, Yen-Lung Lai, Soohyung Kim, and Andrew Beng Jin Teoh, "Ranking-based locality sensitive hashing-enabled cancelable biometrics: Index-of-max hashing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 2, pp. 393–407, 2017.

[17] Yi Liu, Jie Ling, Zhusong Liu, Jian Shen, and Chongzhi Gao, "Finger vein secure biometric template generation based on deep learning," *Soft Computing*, vol. 22, no. 7, pp. 2257–2265, 2018.

[18] Wencheng Yang, Song Wang, Jiankun Hu, Guanglou Zheng, Jucheng Yang, and Craig Valli, "Securing deep learning based edge finger vein biometrics with binary decision diagram," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4244–4253, 2019.

[19] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the International Conference on Machine Learning (ICML)*, Lille, France, Jul. 2015, pp. 448–456.

[20] Florian Schroff, Dmitry Kalenichenko, and James Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2015, pp. 815–823.

[21] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, California., USA, May 2015.

[22] A. Anjos, L. El Shafey, R. Wallace, M. Günther, C. McCool, and S. Marcel, "Bob: a free signal processing and machine learning toolbox for researchers," in *Proceedings of the 20th ACM Conference on Multimedia Systems (ACMMM)*, Oct. 2012.

[23] A. Anjos, M. Günther, T. de Freitas Pereira, P. Korshunov, A. Mohammadi, and S. Marcel, "Continuously reproducing toolchains in pattern recognition and machine learning experiments," in *Proceedings of the International Conference on Machine Learning (ICML)*, Aug. 2017.