Learning the Micro Deformations by Max-pooling for Offline Signature Verification

Yuchen Zheng[a,*], Brian Kenji Iwana[b], Muhammad Imran Malik[c], Sheraz Ahmed[d], Wataru Ohyama[e], Seiichi Uchida[b]

[a] *College of Information Science and Technology, Shihezi University, Shihezi, China*
[b] *Department of Advanced Information Technology, Kyushu University, Fukuoka, Japan*
[c] *Department of Computing, National University of Sciences and Technology, Islamabad, Pakistan*
[d] *German Research Center for Artificial Intelligence, Kaiserslautern, Germany*
[e] *Graduate School of Engineering, Saitama Institute of Technology, Saitama, Japan*

Abstract

For signature verification systems, micro deformations can be defined as the small differences in the same strokes of signatures or special writing habits of different signers. These micro deformations can reveal the core distinction between the genuine signatures and skilled forgeries. In this paper, we prove that Convolutional Neural Networks (CNNs) have the potential to extract those micro deformations by max-pooling. More specifically, the micro deformations can be determined by watching the location coordinates of the maximum values in pooling windows of max-pooling. Extensive analysis and experiments demonstrate that it is possible to achieve state-of-the-art performance by using this location information as a new feature for capturing micro deformations, along with convolutional features. The proposed method outperforms the state-of-the-art systems on four publicly available datasets of different languages, i.e., English (GPDSsynthetic, CEDAR), Persian (UTSig), and Hindi (BHSig260).

Keywords: Offline Signature Verification, Micro Deformations, Max-pooling

[*] Corresponding author

Email addresses: `ouczyc@outlook.com` (Yuchen Zheng), `brian@human.ait.kyushu-u.ac.jp` (Brian Kenji Iwana), `malik.imran@seecs.edu.pk` (Muhammad Imran Malik), `sahmed@dfki.uni-kl.de` (Sheraz Ahmed), `ohyama@sit.ac.jp` (Wataru Ohyama), `uchida@kyushu-u.ac.jp` (Seiichi Uchida)

1. Introduction

Handwritten signatures are some of the most widely used biometric systems. They are used for legal purposes on many administrative and financial documents such as bank cheques, credit cards, passports, identity certificates, and other applications in our daily life [1]. Establishing an effective and efficient handwritten signature verification system to automatically handle a huge volume of signatures plays an important role in security domains [2]. Generally, handwritten signature verification systems are divided into two categories: online and offline. For online systems, data is collected as temporal sequences such as positions of the pen, pressure coordinate sequence, pen elevation coordinate sequence, velocity, stroke order, etc. [3] For offline systems, data is collected from static digital images. Whether online or offline, signature verification is often defined as a two-class classification problem where an automated system has to decide if a query signature is genuine (signature belongs to the referenced authentic writer) or forged (falsely replicated signature of another person). Furthermore, forgeries are commonly categorized as [4],

- Random Forgery: genuine signature of any writer other than the authentic writer.
- Simple/Casual Forgery: a forger only knows the name of the authentic author.
- Simulated/Skilled Forgery: the forger has access to authentic signatures and forges after practicing an unrestricted number of times.

The verification task becomes particularly challenging with skilled forgeries produced by experienced forgers after deliberate practice. To establish a robust signature verification system that could successfully cater to all of the abovementioned forgeries and genuine signatures, the feature extraction procedure is one of the most important steps.

To design a suitable feature extractor for verification systems, one idea is to extract "micro deformations" as potential features from the signatures and use

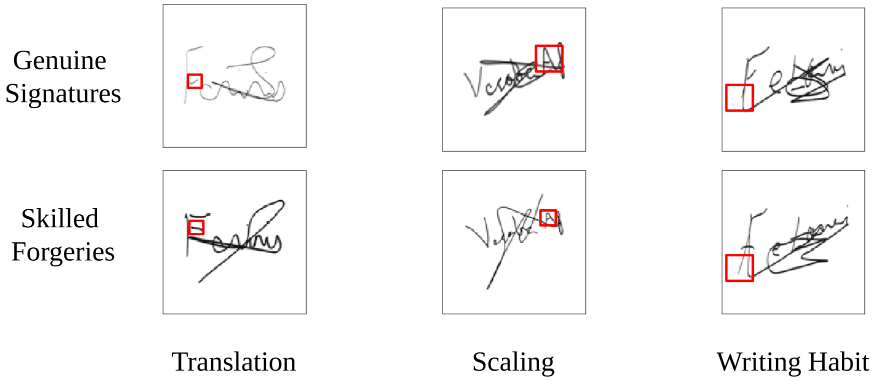


Figure 1: Examples of some specific “micro deformations” occurred between genuine signatures and skilled forgeries. The red blocks represent the part with micro deformations between genuine signatures and skilled forgeries

them to discriminate the genuine signatures and corresponding skilled forgeries. Specifically, the micro deformations between the genuine signatures and skilled forgeries can be described as small translations, transformations, or distortions of strokes, scaling in local regions of signatures, and special writing habits of different signers, etc. Fig. 1 shows examples of different micro deformations that exists between genuine signatures and skilled forgeries. The left sample in Fig. 1 shows a vertical translation in the part of ‘F’ between genuine signature and skilled forgery. The middle sample shows a scaling problem in the part of ‘A’. The right sample shows a specific writing habit of different signers. A tail has occurred at the bottom of ‘E’ in skilled forgery, but it does not occur in a genuine signature. Those micro deformations can be very hard to be discriminated against by traditional verification systems. Without a suitable feature extractor, the micro deformations between genuine signatures and corresponding skilled forgeries may easily be ignored due to only major differences, or “macro deformations”.

In this paper, we prove that Convolutional Neural Networks (CNNs) have the potential to capture the micro deformations in max-pooling layers. Max-pooling is a well-known operation to select the maximum value in fixed pooling windows from the convolutional features. Max-pooling is employed in CNNs for two roles. The first role is to aggregate the information within a pooling window to downsample the convolutional feature maps. The second role of max-pooling is its ability to provide a form of translation invariance, i.e. absorb the micro deformations. Even if convolutional features undergo local spatial translations by the deformation in the input images, by only preserving the maximal response from a pooling window, the reduced feature maps will be invariant to the translations. In other words, it means that the max-pooling operation is not sensitive to such micro deformations by only preserving the maximums in pooling windows.

The motivation of this study is to find how can we detect and penalize the cases when the max-pooling operation is going to remove the micro deformations between the genuine signatures and corresponding skilled forgeries. In the traditional max-pooling operation, spatial information is lost, which may represent the crucial cues of these important micro deformations. We want to recover lost information and micro deformations by watching the behaviors of the max-pooling operation. We note that this collective information (the micro deformations between genuine signatures and skilled forgeries) is the key information for offline signature verification systems. For example, practiced forgers cannot capture all of the writing habits from a specific signer and the skilled forgeries also contain some writing habits from the skilled forgers, which creates some micro deformations between genuine signatures and skilled forgeries.

The main idea of this paper is preserving the micro deformations that would normally be lost due to the ordinary usage of max-pooling. We call the location information (position coordinates) as "displacement features". The idea of using the max location, i.e., displacement feature, is first introduced in [5] for an isolated character recognition task. Although the paper shows the usefulness of the displacement features, the performance improvement is limited to the recognition task. In contrast, we will show the displacement feature can achieve drastic improvement in the offline signature verification task. We can have this result because the micro deformations are very crucial for the signature verification tasks and the displacement features are very essential for representing the micro deformations. Thorough experimentation and analysis have been conducted on a large scale GPDS synthetic offline signature database which validates the use of micro deformations as the proposed system clearly outperforms all the systems reported to date (to the best of authors' knowledge) for offline signature verification on these data. In addition, the proposed method also achieves state-of-the-art results on publicly available benchmark datasets of different languages, i.e., English (CEDAR dataset), Persian (UTSig dataset), and Hindi (BHSig260 dataset).

The main contributions of this paper are summarized as follows.

- We prove that the micro deformations can be extracted from ordinary CNNs. Micro deformations are very crucial for discriminating genuine signatures and skilled forgeries in offline signature verification tasks.
- A two-phase CNN based feature extraction model that not only considers the information between different users but also between the genuine signatures and their corresponding skilled forgeries is applied to the offline signature verification system. For the first phase, a CNN is pre-trained to discriminate the different users. For the second phase, another CNN is applied to discriminate the genuine signatures and skilled forgeries.
- A large scale database GPDSsynthetic is used in experiments to train the feature extractor. The first 500, 1,000, and 2,000 users are used for training the CNN model. The final 5,000 users are used for testing the proposed method and establishing the verification system. The state-of-the-art results are achieved on GPDSsynthetic, CEDAR, UTSig, and BHSig260 datasets from different languages.

This work is an extension of [6]. In the previous work, the feature extractor is only trained between the genuine signatures and the skilled forgeries, and ignored the information between different users. Compared to the previous work, first, this paper considers the user differences in training the feature extractor. Then, more experiments are conducted on three additional signature datasets and compared with state-of-the-art verification systems. Next, more analysis and discussions are organized in this paper, including evaluating different numbers of users to train the feature extractor, demonstrating the similarity matrix of the proposed features in the PCA subspaces, and different kernel functions used in the SVMs as the writer-dependent classifiers to build the verification system. In addition, we also demonstrate how the displacement features capture the micro deformations compared to the pooling features.

The rest of the paper is organized as follows: Section 2 discusses the merits and drawbacks in some existed offline signature verification systems and feature extractors. Section 3 describes how to training a CNN to capture the micro deformations by the displacement features in detail. Section 4 presents the experimental results and discussion. Finally, Section 5 concludes this paper with remarks and future work.

2. Related work

The modern offline signature verification systems often include three steps: preprocessing, feature extraction, and model training. The preprocessing mainly includes noise removal [7, 8], signature normalization and centering [9, 10]. For the feature extraction step, features are often extracted from original/preprocessed signature images by using handcrafted feature extractors [11, 12] and/or deep learning based feature extractors [13, 14]. For the model training step, writer-dependent (one model is trained for each user) [12, 15] and writer-independent (a single model is used to classify signatures from any user) [1, 16] classifiers are normally applied in verification systems.

In order to build a robust offline signature verification system, the feature extraction step plays a crucial role. Considering the recent researches, feature extractors for offline signature verification can be divided into two groups: handcrafted feature extractors and deep learning based feature extractors.

2.1. Handcrafted feature extractors

The Handcrafted feature extractors are widely used in many computer vision applications [17]. In the field of offline signature verification, various handcrafted features are used to train writer-dependent or writer-independent classifiers, such as geometrical features [11], Local Binary Pattern (LBP) features [12], and Scale Invariant Feature Transform (SIFT) features [18]. More recently, many researchers focused on designing robust handcrafted features according to global or local information. In [19], Zois et al. proposed post-oriented grid features that encode the geometric structure of the signatures by grid templates. In [20], Okawa proposed a feature extraction method based on a Fisher vector (FV) with fused "KAZE" features from both foreground and background signature images. The "KAZE" features consider the structures between strokes and stroke contour information more effectively. However, only using the handcrafted features is hard to discriminate genuine signatures and the corresponding skilled forgeries since the handcrafted features often extract local information from the signature images. To learn a good representation from different signatures especially discriminating the genuine signatures and corresponding skilled forgeries by novel approaches is needed.

2.2. Deep learning based feature extractors

In recent years, many deep learning based frameworks are proposed for image classification and detection [21, 22], natural language processing [23, 24], and signature verification [9, 25]. In offline signature verification systems, various deep learning based features are proposed to capture the behaviors of different writers [9, 13]. Zhang et al. [14] proposed an unsupervised feature for offline signature verification, based on Deep Convolutional Generative Adversarial Networks (DCGANs), which has a robust generalization ability compared to handcrafted features. Lai and Jin [26] proposed a CNN based model that learned discriminative features from global and local levels. Furthermore, a Position-Dependent Siamese Network (PDSN) is designed to model the local feature structure which helps to learn a discriminative feature space. Hafemann et al.

[9, 13] proposed a CNN based feature extraction approach, named "Signet", to obtain discriminative features between genuine signatures and skilled forgeries for different users.

Compared to the existing methods, the proposed method has the following advantages. First, the proposed method proves that the micro deformations are captured by using displacement features from a max-pooling layer, which is the key information to discriminate genuine signatures and corresponding skilled forgeries. Second, the proposed method trains the network with a huge number of users, which can be easily used in large-scale verification systems. Third, the proposed method can also discriminate the different users well by a pre-trained CNN.

3. Learning micro deformations by max-pooling

In this section, we introduce how to fuse the displacement features and pooling features as the final discriminative features for capturing the micro deformations not only between different users but also between genuine signatures and skilled forgeries. First, we introduce a CNN based architecture trained between genuine signatures from different users so that to capture the differences between different users. Then, based on the pre-trained CNN, we introduce how to extract the displacement features and fuse them with pooling features to capture the micro deformations between genuine signatures and skilled forgeries in a combined architecture. Finally, we introduce how to train writer-dependent classifiers based on the fused features to build a complete verification system.

3.1. Pre-training a CNN to classify the signatures from different users

To discriminate different users, we design a CNN based architecture with 3 convolutional and pooling layers, 2 fully-connected layers and a softmax layer. In the convolutional layers, the kernel size is 3×3 with stride 1, and the number of the filters is 32, 64 and 128, respectively. In the pooling layers, the max-pooling size is 2×2 with stride 2. In the fully-connected layers, the first fully-connected layer has $4,096$ nodes and reduces to $2,048$ in the second fully-connected layer. Rectified Linear Unit (ReLU) is used as the activation function for the network, and batch normalization is used to speed the training process. In the softmax layer, the number of the neurons is according to the number of users (for example, 500 users correspond to 500 neurons). Cross-entropy is used as the loss function to train the network.

3.2. Capturing the micro deformations between genuine signatures and skilled forgeries

After the CNN training process, the difference information between the different users is obtained. Since the target of the signature verification systems is mainly to distinguish genuine signatures from corresponding skilled forgeries, the proposed method should also learn the difference (micro deformations) between the genuine signatures and skilled forgeries. In this part, we first introduce the displacement features extracted from the max-pooling layers. Then, we introduce how to fuse the displacement features with pooling features to capture the micro deformations between genuine signatures and skilled forgeries.

3.2.1. Extracting displacement features from pooling layers

Considering the spatial information lost in max-pooling operation, we extract displacement features [5, 27] from the first pooling layer of the pre-trained CNN. The displacement features represent the location and direction of the maximums in the pooling windows, which might be the crucial information between the genuine signatures and corresponding skilled forgeries. Fig. 2 shows the examples of translation, scaling, and nonlinear cases that occurred in the max-pooling operation. In the translation case, a vertical translation in the maximums shifts the elements up and down. In the scaling case, the area zooms out from the top figure to the bottom figure. In the nonlinear case, the shape of the area is also different between the two figures. However, after the max-pooling operation, both cases achieve the same pooling results in a pooling window. If we only retain the maximums from the max-pooling operation, spa-

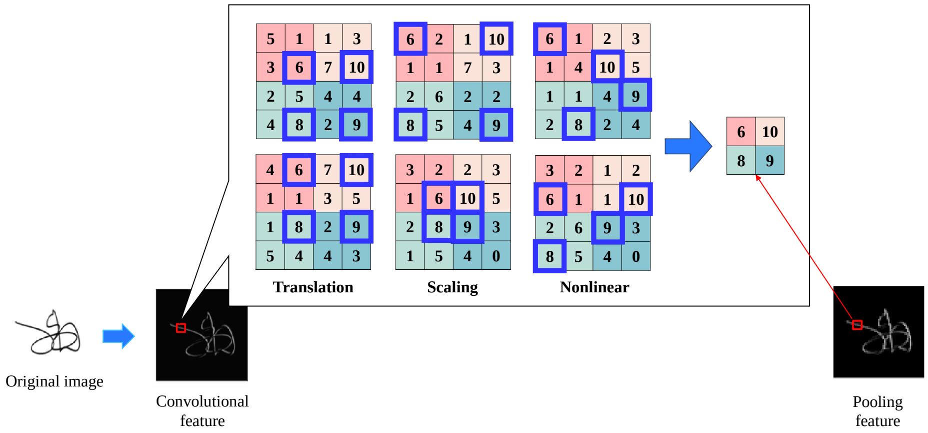


Figure 2: Relationship between the micro deformations to the location of maximums in pooling windows. Different colors represent the different pooling windows. The pooling size is 2 × 2 with stride 2. The blue blocks represent the maximums of each pooling window. Note that those micro deformations do not affect the final max-pooling results. This means that conventional CNNs ignore the micro deformations.

tial information will be lost. The displacement features can record these crucial deformations lost in max-pooling operation.

Fig. 3 shows the procedure of extracting the pooling features and displacement features simultaneously. Here, the pooling size is 2×2 with stride 2, the value of the displacement features both in horizontal and vertical directions bound by $[-1, 1]$. The displacement features record the location and direction of the max response in the receptive field and are intended to extract information which might be the major difference between the genuine signatures and corresponding skilled forgeries. Fig. 4 presents the pooling features and displacement features of samples from the GPDSsynthetic dataset based on a Hue-Saturation-Value (HSV) color model whose color and intensity denote the direction and average length of the displacement features. Here, the left part is the samples of genuine signature and the right part is the samples of skilled forgeries. In the same channel, the pooling features between the genuine signatures and corresponding skilled forgeries are very similar. But, the displacement features describe the location information of maximums in max-pooling opera-

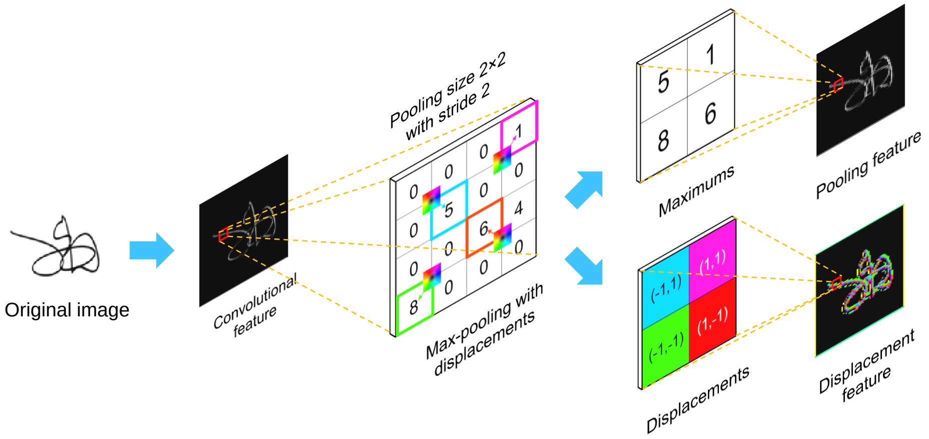


Figure 3: The procedure of the feature extraction in pooling windows. The pooling size is 2×2 with stride 2. The pooling features are the maximums in the pooling windows, the displacement features are the position coordinates of the corresponding maximums. The displacement vector $(-1, 1)$ means that the vertical displacement from the center to the maximum value is -1 and the horizontal displacement is 1.

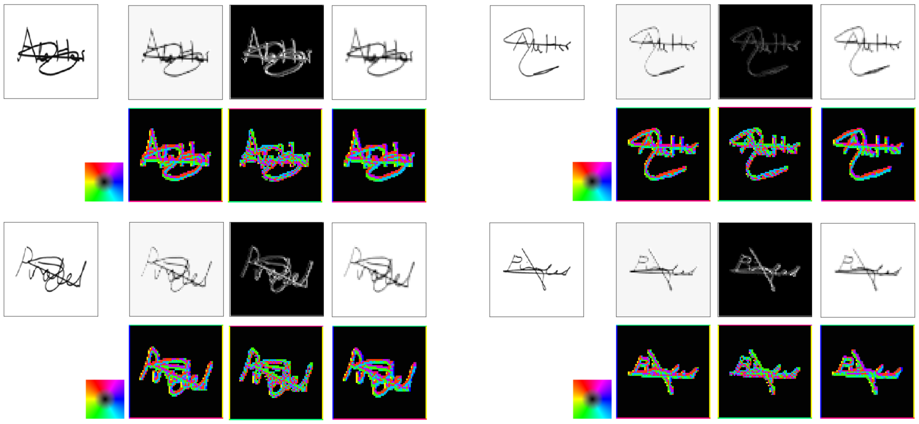


Figure 4: Visualization of the pooling features and the displacement features of two samples on GPDSsynthetic dataset. The samples on the left are genuine samples and the samples on the right are skilled forgeries. For each sub-figure, the upper left image is the original signature, the first row shows the corresponding pooling features, and the second row shows the displacement features. Each column represents one convolutional filter. The visualization of displacement features is based on an HSV color model whose color and intensity denote the direction and average length of the displacement features.

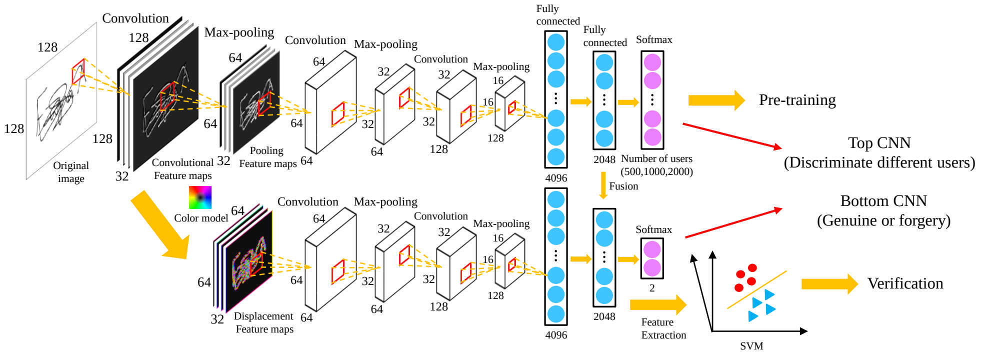


Figure 5: The feature extraction and verification procedures. The top CNN is designed to classify the difference between different users. The bottom CNN is designed to obtaining the difference between the genuine signatures and skilled forgeries by using displacement features.

tion, which might capture some micro deformations of forgery signatures when some skilled writers imitated the genuine signatures.

3.2.2. Training a CNN to capture the micro deformations

To capture the micro deformations between the genuine signatures and skilled forgeries, we used a second CNN to classify the genuine signatures and skilled forgeries based on the extracted displacement features. The architecture that we used for fusing the pooling features and displacement features is shown in Fig. 5. We can see that the architecture for processing the displacement features is the same as the pre-trained CNN without the first convolutional and pooling layers. The other difference between the two CNNs is the final softmax layers. The top CNN is designed to classify the original signature images from different users. The displacement features are extracted from the top CNN and used as the inputs of the bottom CNN. Here, we divide the displacement into the horizontal and vertical directions and apply the same architecture. We then fuse the pooling features and displacement features in the last fully-connected layer as the final discriminative features.

3.3. Training the writer-dependent classifiers

After the feature extraction processing, we obtain the fused features from the previous CNN based feature extractor. The next step is to build the verification system. Here, we train the writer-dependent classifiers to build the system. For each user (not included in CNN training procedure), we use the genuine signatures as the positive samples and genuine signatures from other users as the negative samples to build the training set (no forgery signature is used). Then, we choose linear and RBF kernel SVMs as the writer-dependent classifiers to train the models.

When using writer-dependent classifiers, there is a large imbalance in positive and negative samples due to the target user's genuine signatures being used as positive samples, and other users' signatures used as random forgeries. To overcome the imbalanced problem that the negative samples are much more than the positive samples, we use different weights settings for the positive and negative classes [9] in the objective function. Then, the SVM objective function becomes,

$$\begin{aligned} &\min \ \frac{1}{2}\mathbf{w}^T\mathbf{w} + C_p \sum_{\substack{i=1 \\ y_i=+1}}^{M} \xi_i + C_n \sum_{\substack{i=1 \\ y_i=-1}}^{N} \xi_i \\ &s.t. \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \ \text{ and } \ \xi_i \geq 0, \end{aligned} \tag{1}$$

where $\mathbf{x}_i$ is a fused feature of the training sample with target value y_i, $\mathbf{w}$ is the weight parameter in SVM, ξ_i is the slack variables, M and N are the numbers of the positive and negative samples, C_p and C_n are the weights for the positive and negative class,

$$C_p = \frac{N}{M} C_n. \tag{2}$$

For the testing procedure, we build two test sets. The first one uses the remaining genuine signatures of the target user as the positive samples and skilled forgeries of the target user as the negative samples to build the test set for evaluating the skilled forgeries experiment. The second one uses the remaining genuine signatures of the target user as the positive samples and the remaining genuine signatures of the other users as the negative samples for evaluating the random impostor experiment.

4. Experiment

In this section, we first introduce the datasets, preprocessing and experimental protocol. Then, we use two CNN based architectures to extract features from original signatures and evaluate the proposed methods. The first one uses the original signature to train a CNN between the different users and the displacement features to train another CNN for capturing the micro deformations between the genuine signatures and skilled forgeries. The second one uses the original signature to train a CNN between the genuine signatures and skilled forgeries and the displacement features to train another CNN between the different users. Next, we evaluate the proposed method by using different numbers of users (500, 1,000, and 2,000 users). Finally, we calculate the class-wise similarity matrix between the genuine signatures, skilled forgeries, and random impostors in Principal Component Analysis (PCA) subspaces and discuss the performance of the proposed models in detail.

4.1. Experimental protocol

4.1.1. Datasets

We conduct the experiments on GPDSsynthetic [28], CEDAR [29], UT-Sig [30], and BHSig260 [31] datasets to evaluate the proposed method, which belong to different language scripts. The GPDSsynthetic is a large scale dataset that contains 24 genuine signatures and 30 skilled forgeries for each user. The number of users is 10,000, so the GPDSsynthetic dataset contains 240,000 genuine signatures and 300,000 skilled forgeries and it is very suitable for deep learning based methods. The CEDAR dataset is an English signature dataset that consists of 55 users with 24 genuine signatures and 24 forgeries for each user. Hence, the dataset is comprised of 2,640 signatures. The UTSig is a Persian offline signature dataset which consists of 8,280 signatures from 115 users. Each user has 27 genuine signatures of an authentic person, 3 opposite-hand signed, and 42 skilled forgeries. The BHSig260 dataset contains two subsets: BHSig260 (Bengali) dataset and BHSig260 (Hindi) dataset. The BHSig260 dataset is a Bengali signature dataset that consists of 24 genuine signatures and 30 forged signatures from 100 users. The BHSig260 dataset is a Hindi signature dataset that consists of 24 genuine signatures and 30 forged signatures from 160 users.

4.1.2. Preprocessing

Since the signature images are different sizes in the datasets and the proposed method expects the inputs of a fixed size, we apply several preprocessing steps in our experiment. First, we center the signatures by using the signatures' center of mass. Then, we binarize the images using OTSU's method [8]. Finally, to determine the input size, we conduct an experiment with different input sizes (64×64, 128×128, and 256×256) on the GPDSsynthetic dataset. If the input size is small (64×64), the performance is not better than bigger input sizes (128×128 and 256×256). And the performances are similar when the input sizes are 128×128 and 256×256. Therefore, we choose 128×128 as the input size in our experiments.

4.1.3. Experimental settings

For the procedure of training the CNN based feature extractor, we use the part of signatures from the GPDSsynthetic dataset. For building the training set, we use the signatures from the user of No. 5001 to No. 5500 (500 users), No. 5001 to No. 6000 (1,000 users) and No. 5001 to No. 7000 (2,000 users). Then, we use the final 50 users (No. 9951 to No. 10000) as the validation set.

To build the signature verification systems, we train linear and RBF kernel SVMs as the writer-dependent classifiers for each user. To evaluate the proposed method, we use 5 sub-datasets, GPDS-150, GPDS-300, GPDS-1000, GPDS-2000, GPDS-5000 (the first 100, 150, 1,000, 2,000, 5,000 users of GPDS-10000 dataset) for final evaluation. For a specific user, we randomly select 5 genuine signatures as the positive samples and 5 genuine signatures from other users as the negative samples to build the training set for writer-dependent classifiers. For the writer-dependent classifiers training process, the weights C_n are found by grid search on the validation set, and the C_p is calculated by Eq. (2).

For the evaluation of the test set, the remaining genuine signatures from the target user are used for calculating the False Rejection Rate (FRR). The False Acceptance Rate for the skilled forgeries (FAR$_{skilled}$) experiment has been obtained with forgery samples of the target user. The False Acceptance Rate for the random impostor (FAR$_{random}$) experiment has been obtained with the genuine signatures from all the remaining users. The Equal Error Rate for skilled forgeries experiment (EER$_{skilled}$) is calculated by FAR$_{skilled}$ = FRR, and the EER for the random impostor experiment (EER$_{random}$) is calculated by FAR$_{random}$ = FRR.

4.2. Training two CNN based architectures

To evaluate the performance of the displacement features in the feature extraction process, we design two CNN based architectures trained between 500 users. The first one (Proposed-1) uses the original signature to train a CNN for capturing the micro deformations between the different users and the displacement features to train another CNN between the genuine signatures and skilled forgeries. The second one (Proposed-2) uses the original signature to train a CNN between the genuine signatures and skilled forgeries and the displacement features to train another CNN between the different users. The purpose of this design is to evaluate the kind of information that the displacement can extract from the signatures (the difference between different users or the "micro deformations" between genuine signatures and skilled forgeries). The architectures of the 'Proposed-1' and 'Proposed-2' are shown in Fig. 6. Since this study mainly focuses on capturing the micro deformations between the genuine signatures and corresponding skilled forgeries, the architecture selection is not the key point. Other state-of-the-art CNN-based architectures can also be selected for building verification systems.

For the training process, the hyperparameters are selected from the valida-

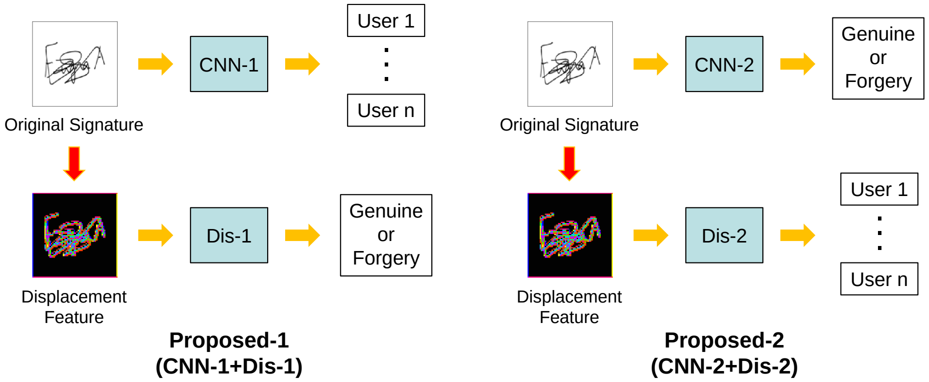


Figure 6: The architectures of the proposed models. The left part is the architecture of 'Proposed-1' model and the right part is the architecture of 'Proposed-2'. model.

tion set. We use Adam as the optimizer to minimize the loss function with mini-batch size 64. The model is trained for 40 epochs. The initial learning rate is set to 10^{-4} and reduced by a factor of 0.95 after each epoch. After the CNN training process, we take the trained model as a feature extractor to extract the fused features and train the RBF kernel SVMs as the writer-dependent classifiers to build the verification system. Here, we note that the inference time of feature extraction and verification procedure is very efficient and similar to traditional CNN-based architectures and feature extraction methods.

To evaluate the performance of the proposed method, we test the proposed method on the GPDS-150, GPDS-300, GPDS-1000, GPDS-2000, and GPDS-5000 datasets and compared it with the traditional CNNs (CNN-1 and CNN-2) based features, and only using the displacement features based architectures (Dis-1 and Dis-2). For the traditional CNN models, CNN-1 is a feature extractor trained to discriminate different users and CNN-2 is trained between the genuine signatures and skilled forgeries. For the displacement features based architectures, Dis-1 is trained between the genuine signatures and skilled forgeries and Dis-2 is trained between different users. The experimental results are the averages of all users with 10 trials. We conducted the pairwise t-test with confidence value 0.05 between the proposed methods and traditional CNN mod-

Table 1: Performance of different models for the skilled forgeries experiment (EER$_{skilled}$ in %). The CNN based architectures are trained by using data from 500 users. RBF kernel SVMs are used as the writer-dependent classifiers.

Dataset	CNN-1+SVM	CNN-2+SVM	Dis-1+SVM	Dis-2+SVM	Proposed-1	Proposed-2
GPDS-150	12.34±0.42	9.85±0.37	10.86±0.35	14.59±0.39	**8.23±0.41**	9.54±0.44
GPDS-300	12.45±0.44	9.94±0.41	10.94±0.52	14.36±0.32	**8.18±0.47**	9.71±0.36
GPDS-1000	12.38±0.47	9.72±0.39	10.84±0.48	14.88±0.56	**8.37±0.51**	9.52±0.39
GPDS-2000	12.52±0.39	9.68±0.38	11.01±0.42	14.75±0.47	**8.44±0.49**	9.72±0.42
GPDS-5000	12.43±0.51	9.75±0.44	10.88±0.45	14.62±0.51	**8.33±0.47**	9.65±0.35

Table 2: Performance of different models for the random impostors experiment (EER$_{random}$ in %).

Dataset	CNN-1+SVM	CNN-2+SVM	Dis-1+SVM	Dis-2+SVM	Proposed-1	Proposed-2
GPDS-150	1.88±0.32	4.85±0.42	5.63±0.44	2.04±0.41	2.01±0.45	**1.89±0.31**
GPDS-300	1.75±0.33	4.92±0.34	5.72±0.41	2.14±0.38	1.89±0.42	**1.56±0.28**
GPDS-1000	1.82±0.41	4.98±0.42	5.84±0.45	1.98±0.45	1.92±0.53	**1.50±0.28**
GPDS-2000	1.85±0.35	5.01±0.53	5.82±0.37	1.91±0.37	1.87±0.47	**1.50±0.40**
GPDS-5000	1.78±0.41	4.88±0.39	5.91±0.47	1.94±0.34	1.83±0.49	**1.23±0.31**

els (Proposed-1 corresponds to CNN-1 and Proposed-2 corresponds to CNN-2).

Table 1 shows the results of the skilled forgeries experiment. This experiment is to verify whether the query samples are genuine signatures or skilled forgeries. We can see that only using the SVMs with features extracted from a traditional CNN (CNN-1 or CNN-2) can achieve reasonable results. Furthermore, the result of CNN-2 is better than CNN-1. This is because CNN-2 is trained between the genuine signatures and skilled forgeries, but the CNN-1 is only trained between the different users. Only using the displacement features (Dis-1 and Dis-2) can obtain similar results but slightly worse. This is because the displacement features only contain the location information of the maximums that are not enough to fully represent the original signatures for the verification system. In addition, as the number of users included in the datasets increase, the EERs do not change significantly. This means that the proposed feature extractor is stable for larger-scale verification systems.

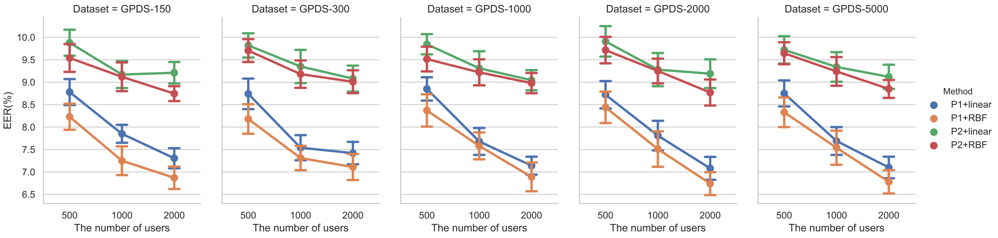


Figure 7: Performance of the skilled forgeries experiment (EER$_{skilled}$ in %) when using 5 samples of each users to train the writer-dependent classifiers. Here, 'P1' and 'P2' are the 'Proposed-1' and 'Proposed-2' based feature extractors, respectively. 'Linear' and 'RBF' represent the linear SVM and RBF kernel SVM that are used in verification system.

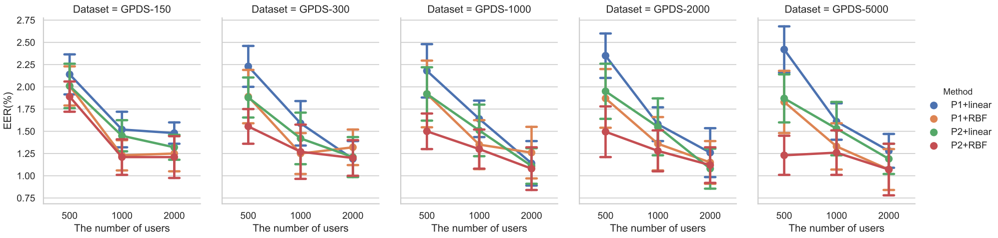


Figure 8: Performance of the random impostors experiment (EER$_{random}$ in %) when using 5 samples of each users to train the writer-dependent classifiers.

Table 2 shows the results of the random impostors experiment. This experiment is mainly to discriminate the signatures from the target user with the signatures from other users. We can see that the proposed methods also achieve good performance.

4.3. Training the models with different number of users

The previous experiments only use 500 signatures to train the proposed feature extractors. In this section, we discuss the influence of using different numbers of users. We consider using 1,000 and 2,000 users to train the C-NN based architectures. We also use the previous architectures (Proposed-1 and Proposed-2) in our experiments. For the process of training the writer-dependent classifiers, we also choose the linear and RBF kernel SVMs to build the verification system. Here, we also select 5 genuine signatures of each user to build the training set.

For the 1,000 users' training set, we select the signatures from the user of No. 5001 to No. 6000. the validation and test sets are as same as the previous experiments. For the CNN training process, since it is hard to train between the huge number of users, we chose an initial learning rate of 1e-5 for 400 epochs. Other hyper-parameters are as same as before. For 2,000 users' training set, we select the signatures from the user of No. 5001 to No. 7000. For the CNN training process, the initial learning rate is also set to 1e-5, and the number of the epoch is set to 600. Other hyper-parameters are also as same as before. The experimental results are shown in Figs. 7 and 8.

From Fig. 7 and 8, we can see that as the number of users is increased, the EERs of the proposed models decrease on all datasets. For the skilled forgeries experiment, the first architecture with RBF kernel SVMs (P1+RBF) achieves the best results when the training samples include both 1,000 and 2,000 users. For the random imposters experiment, when the training samples include 1,000 users, the first architecture with RBF kernel SVMs (P1+RBF) achieves the best result on the GPDS-2000 dataset. When the training samples include 2,000 users, the first architecture obtains the best result on the GPDS-300 dataset. In other cases, the second architecture with RBF kernel SVM (P2+RBF) obtains the best results. Through the experiments, we can see that the fused features that are trained not only between different users, but also between the genuine signatures and skilled forgeries can capture the micro deformations and work well to discriminate the genuine signatures, skilled forgeries, and random impostors.

Fig. 9 presents two improved examples by using the proposed method. We visualized the displacement features to observe the micro different behaviors between the genuine signatures and skilled forgeries. From Fig. 9, we can see that the genuine signatures have similar behaviors on the displacement features and the skilled forgeries are different from the genuine samples in some places. For the first sample, in the left part of the first filter that is shown in the figure, the genuine samples have some features in red and blue directions, but it is rare in the skilled forgery samples. In the second filter, the displacement features

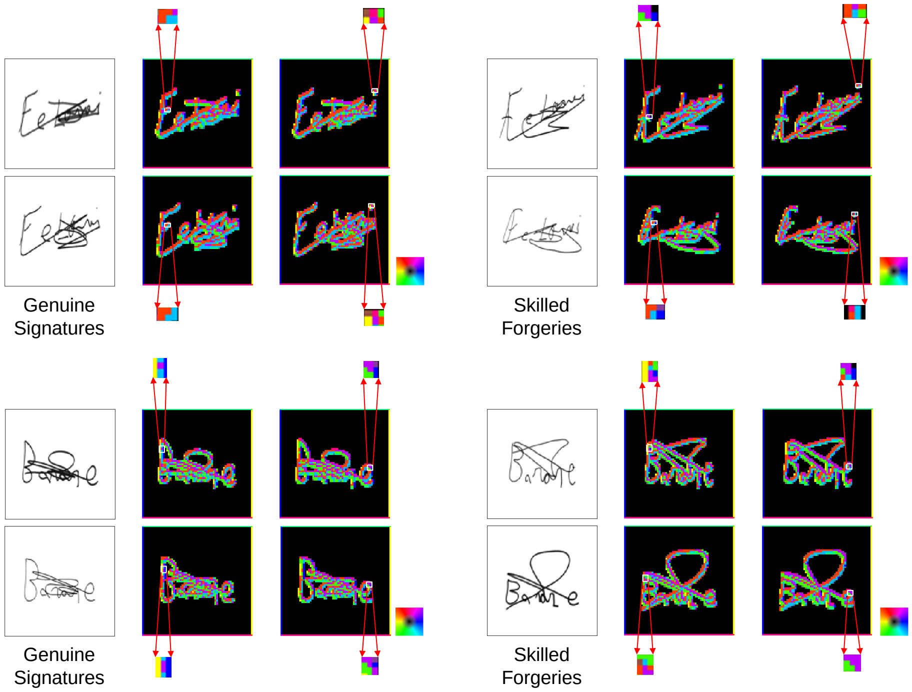


Figure 9: Examples improvement of the displacement features by capturing the micro deformation between the genuine signatures and skilled forgeries. The first column is the original signature images, the second and the third columns are the displacement features extracted from two convolutional filters.

can capture the same direction on the 'dot' part in the genuine signatures but in the skilled forgeries, the corresponding position is different compared to the genuine signatures. The second sample has a similar behavior with the first sample. The improved samples demonstrate that the proposed method can capture some micro deformations or distortions between the genuine signatures and skilled forgeries, which is very important for the verification systems.

4.4. Similarity of the proposed features in the PCA subspaces

To further observe the behaviors of the proposed feature extractors, we measure the similarities of the fused features not only between the different users but also between the genuine signatures and skilled forgeries in the PCA subspaces. At first, we train the Proposed-1 and Proposed-2 models on the training set. Then, we extract the features of the genuine signatures and skilled forgeries from the target user, and the genuine signatures from the other users. Finally, we train different PCA models on the samples that are in the same classes and preserve 10% of eigenvalues for each PCA model to build the PCA subspaces. The similarity can be defined by using the canonical angles,

$$\cos\delta_i = \sup_{\substack{\boldsymbol{\alpha}_i \perp \boldsymbol{\alpha}_j, \boldsymbol{\beta}_i \perp \boldsymbol{\beta}_j \\ 1 \leq i, j \leq p}} \frac{\boldsymbol{\alpha}_i^T \boldsymbol{\beta}_i}{\|\boldsymbol{\alpha}_i\| \|\boldsymbol{\beta}_i\|}. \tag{3}$$

Here, $\boldsymbol{\alpha} \in \mathbf{P}, \boldsymbol{\beta} \in \mathbf{Q}$, $\mathbf{P}$ and $\mathbf{Q}$ are two PCA subspaces, $\mathbf{P}, \mathbf{Q} \in \mathbb{R}^n$, $\dim \mathbf{P} = p \leq \dim \mathbf{Q} = q$. Then the similarity can be defined as,

$$S = \frac{1}{p} \sum_{i=1}^{p} \cos^2 \delta_i. \tag{4}$$

If two PCA subspaces completely coincide with each other, all canonical angles will be 0 and S equals to 1. The similarity gets smaller as the two spaces separate. Finally, the similarity is zero when the two subspaces are orthogonal to each other [32]. The similarity matrices are shown in Fig. 10.

From Fig. 10, we can see that, as the number of users is increased, the similarity between the genuine signatures and skilled forgeries decreases. In the model of Proposed-2, this similarity between the genuine signatures and skilled forgeries is higher than the Proposed-1 model, which means that the features learned from the Proposed-1 have better discrimination than the Proposed-2. For the random impostor samples, the similarities with the genuine signatures are all very low, and it also decreases as the number of the user increases. Through this, we can observe that the proposed method can not only discriminate the different users well but also discriminate the genuine signatures and their corresponding skilled forgeries well.

4.5. Comparing the proposed method with state-of-the-art models on the GPDS dataset

We also compare our proposed method with many state-of-the-art systems on the GPDS dataset. Since the GPDS dataset has many versions, we choose

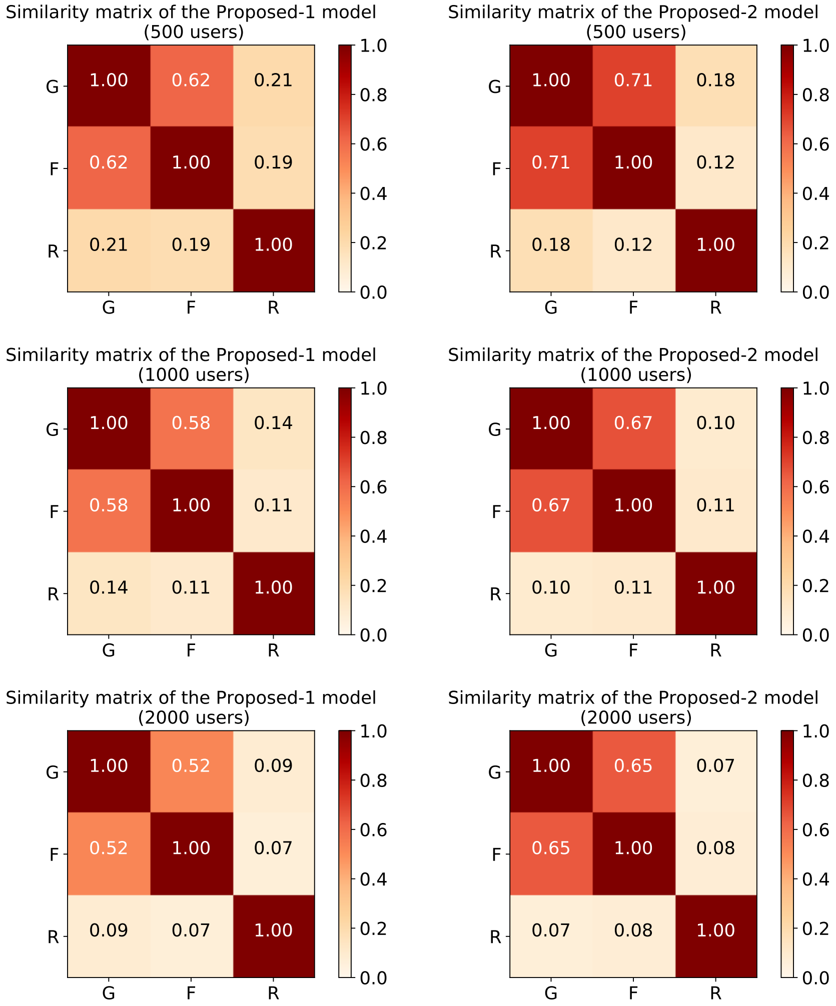


Figure 10: The similarity matrix of Proposed-1 and Proposed-2 models in PCA subspaces when the number of the training samples include 500, 1,000, and 2,000 users. 'G' is the class of the genuine signatures, 'F' is the class of the skilled forgeries, 'R' is the class of the random impostor samples.

Table 3: Comparison with the state-of-the-art systems on the GPDS dataset (EER$_{skilled}$ (standard deviation) in %).

Systems	Source	#Refs	Number of user							
			75	150	160	300	1000	2000	4000	5000
[33]	GPDS960Gray	5	-	-	4.01(0.39)	4.40(0.34)	-	-	-	-
[33]	GPDS960Gray	12	-	-	2.86(0.24)	3.34(0.22)	-	-	-	-
[34]	GPDS960Gray	5	-	-	3.83(0.33)	4.53(0.14)	-	-	-	-
[34]	GPDS960Gray	14	-	-	2.74(0.18)	3.47(0.16)	-	-	-	-
[25]	GPDS960Gray	10	15.08	-	-	20.94	-	-	-	-
[13]	GPDS960Gray	12	-	-	-	**0.41(0.05)**	-	-	-	-
[35]	GPDS960Gray	12	-	-	-	9.25	-	-	-	-
[9]	GPDS960Gray	5	-	-	2.41(0.12)	2.42(0.24)	-	-	-	-
[9]	GPDS960Gray	12	-	-	**1.72(0.15)**	1.69(0.18)	-	-	-	-
[19]	GPDS960Gray	12	-	-	-	3.24	-	-	-	-
[36]	GPDS960Gray	5	-	-	-	9.04	-	-	-	-
[36]	GPDS960Gray	12	-	-	-	5.53	-	-	-	-
[37]	GPDS960Gray	10	-	-	-	9.94	-	-	-	-
[38]	GPDS960Gray	4	-	-	-	16.92	-	-	-	-
[38]	GPDS960Gray	8	-	-	-	15.95	-	-	-	-
[38]	GPDS960Gray	12	-	-	-	15.07	-	-	-	-
[39]	GPDS960Gray	10	-	-	10.70	12.83	-	-	-	-
[16]	GPDS960Gray	12	-	-	-	3.47(0.15)	-	-	-	-
[40]	GPDSsynthetic	pairs	-	-	-	-	-	10.37	-	-
[41]	GPDSsynthetic	5	-	10.64(0.32)	-	11.01(0.28)	-	-	-	-
[42]	GPDSsynthetic	5	-	10.89(0.27)	-	11.56(0.33)	-	10.37	-	-
[43]	GPDSsynthetic	5	-	-	-	-	-	-	7.99	-
[43]	GPDSsynthetic	7	-	-	-	-	-	-	7.34	-
[43]	GPDSsynthetic	10	-	-	-	-	-	-	6.81	-
[44]	GPDSsynthetic	5	-	-	-	22.13(0.42)	-	-	-	-
[44]	GPDSsynthetic	12	-	-	-	14.93(0.18)	-	-	-	-
[45]	GPDSsynthetic	10	-	-	-	-	-	-	6.13(0.29)	-
[34]	GPDSsynthetic	10	-	-	-	-	-	-	8.70(0.35)	-
[46]	GPDSsynthetic	10	6.84	-	-	-	-	-	-	-
[47]	GPDSsynthetic	10	7.24	-	-	-	-	-	-	-
[25]	GPDSsynthetic	10	12.83	12.67	-	-	12.43	12.80	13.3	-
[35]	GPDSsynthetic	10	-	-	-	-	-	-	18.32	-
[14]	GPDSsynthetic	14	-	-	-	-	-	-	14.79	-
[48]	GPDSsynthetic	10	6.62	-	-	-	-	-	-	-
[11]	GPDSsynthetic	5	-	11.48	-	12.11	11.07	11.34	-	11.10
[12]	GPDSsynthetic	5	-	16.45	-	16.50	17.01	16.63	-	16.93
Ours	GPDSsynthetic	5	6.52(0.42)	6.87(0.35)	-	7.11(0.41)	6.89(0.45)	6.74(0.36)	6.69(0.38)	6.78(0.37)
Ours	GPDSsynthetic	10	5.31(0.37)	5.45(0.42)	-	5.38(0.36)	5.54(0.41)	5.47(0.32)	5.31(0.41)	5.29(0.47)
Ours	GPDSsynthetic	12	**4.74(0.34)**	**4.82(0.38)**	-	4.52(0.42)	**4.71(0.43)**	**4.62(0.38)**	**4.64(0.40)**	**4.59(0.41)**

the GPDS960Gray and GPDSsynthetic as the source datasets. Then, according to the number of users, we build the subsets of the source datasets and compare them in Table. 3. The proposed feature extractor is trained between 2,000 users on GPDSsynthetic dataset.

From Table. 3, we can see that the proposed model works well on the G-PDSsynthetic dataset. As the number of users increases, the performance of the proposed model is stable. Even though some comparison methods work well on GPDS-160 and GPDS-300 datasets, they are not necessarily applicable to large amounts of users. In addition, the proposed method performs better than systems[11, 12, 35, 36, 37, 41, 42, 45] which apply the traditional hand-crafted or computer vision based features to feature extraction procedure.

4.6. Comparing the proposed method with state-of-the-art models on other datasets

To evaluate the generalization performance of the proposed method, we choose three real signature datasets, CEDAR, UTSig, and BHSig260, of different language scripts. We first train the feature extractor between 2,000 users on GPDSsynthetic dataset. Then, we extract the features and train writer-dependent classifiers on these 3 datasets respectively. For CEDAR and UTSig dataset, we randomly select 5, 10, and 12 genuine signatures of each user to train the writer-dependent classifiers. For BHSig260 dataset, we randomly select 2, 5, and 8 genuine signatures of each user to train the writer-dependent classifiers. Table. 4, 5, and 6 present the comparison with state-of-the-art performance on CEDAR, UTSig, and BHSig260 datasets, respectively.

For the CEDAR dataset, we obtain competitive results using the proposed method. The proposed method obtains 2.76% EER$_{skilled}$ when the number of reference samples is 12. System [51] achieves 0.79% EER$_{skilled}$, which is better than our system. However, the proposed method outperforms System [51] on the UTSig dataset. System [13] achieves 2.33% EER$_{skilled}$, which is a little better than our system. Compared to this system, the proposed method does not use samples from the CEDAR dataset to fine-tune the feature extractor, which is

Table 4: Comparison with the state-of-the-art systems on the CEDAR dataset.

Systems	#Refs	EER$_{random}$ (%)	EER$_{skilled}$ (%)
[48]	10	0.61	5.76
[9]	4	-	5.92(0.48)
[9]	8	-	4.77(0.76)
[9]	12	-	4.53(0.42)
[40]	pairs	-	8.50
[49]	1	-	11.59
[19]	5	-	4.12
[19]	10	-	3.02
[13]	10	0.37	3.60(1.26)
[13]	10	-	2.33(0.88)*
[16]	12	-	3.32(0.22)
[50]	5	-	2.30
[51]	10	-	**0.79**
[52]	5	-	2.90
[53]	16	-	1.00
Ours	5	0.65(0.17)	3.89(0.45)
Ours	10	0.32(0.12)	2.95(0.38)
Ours	12	**0.21(0.07)**	2.76(0.43)

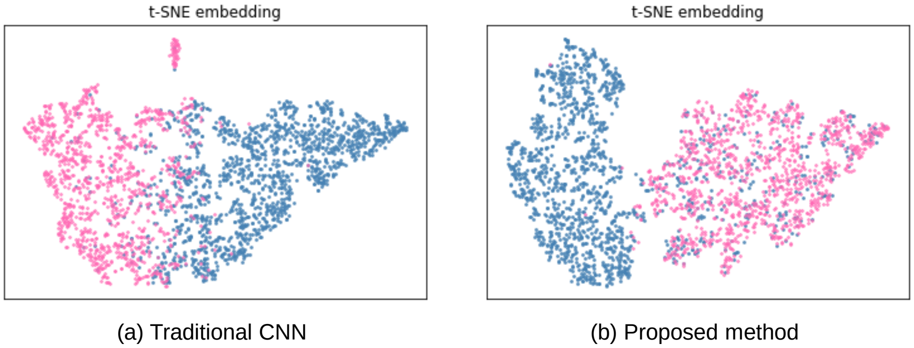
* Fine-tuned model.

Table 5: Comparison with the state-of-the-art systems on the UTSig dataset.

Systems	#Refs	EER$_{random}$ (%)	EER$_{skilled}$ (%)
[25]	12	-	17.45
[48]	12	1.11	11.75
[43]	5	-	11.16
[43]	10	-	9.80
[51]	12	-	6.22
[30]	12	-	29.71
[54]	12	-	16.00
Ours	5	1.33(0.11)	7.86(0.47)
Ours	10	0.96(0.15)	6.62(0.58)
Ours	12	**0.82(0.13)**	**6.14(0.32)**

Table 6: Comparison with the state-of-the-art systems on the BHSig260 dataset.

Dataset	Systems	#Refs	EER$_{random}$ (%)	EER$_{skilled}$ (%)
Bengali	[55]	2	1.78	10.67
Bengali	[31]	8	-	33.82
Bengali	Ours	2	1.65(0.21)	9.87(0.34)
Bengali	Ours	5	1.03(0.16)	8.92(0.41)
Bengali	Ours	8	**0.88(0.13)**	**8.21(0.38)**
Hindi	[55]	2	1.34	11.88
Hindi	[31]	8	-	24.47
Hindi	Ours	2	1.42(0.18)	10.53(0.45)
Hindi	Ours	5	1.11(0.21)	9.84(0.42)
Hindi	Ours	8	**0.95(0.17)**	**9.01(0.39)**



(a) Traditional CNN

(b) Proposed method

Figure 11: Visualization of features of random 50 users on CEDAR dataset by t-SNE. (a) Traditional CNN-based feature extractor. (b) Proposed feature extractor. Here, the feature extractors are trained on 2,000 users. The pink (light) and blue (dark) dots represent the genuine signatures and skilled forgeries.

the reason that the performance of the proposed method is a little worse than that system. In addition, the lower EER$_{random}$ achieved in random forgery scenario also demonstrates the general effectiveness of the proposed method.

Fig. 11 compares the features learned from the traditional CNN-based feature extractor with the proposed feature extractor by t-SNE [56]. We can see that the genuine signatures are mixed and easily confused with the skilled forgeries around the center of Fig. 11 (a). However, the two are more easily separated in Fig. 11 (b). This means that the proposed feature extractor discriminates the genuine signatures and skilled forgeries well than the traditional CNN-based feature extractor. In other words, it proves that the displacement features are better than pooling features for capturing the micro deformations between the genuine signatures and skilled forgeries.

For the UTSig dataset, the proposed method obtains state-of-the-art results compared with other systems. The proposed method achieves 7.86%, 6.62%, and 6.14% EER$_{skilled}$ when the number of reference samples is 5, 10, and 12, respectively. Even if we only use 5 reference samples of each user, the proposed method also obtains the best performance compared with other systems.

For the BHSig260 dataset, to compare the proposed method with state-of-the-art systems in a fair way, we choose 2, 5, and 8 instead of 5, 10, and 12 as the number of reference samples to train the writer-dependent classifiers. Due to this dataset contains two sub-datasets, Bengali and Hindi, we report the results separately. We can see that the proposed method also achieves the best performance compared with other state-of-the-art systems. For the Bengali dataset, the proposed method obtains 9.87% EER$_{skilled}$ when the number of reference samples is 2, which is still better than 10.67% EER$_{skilled}$ obtained by [55]. For the Hindi dataset, we can obtain a consistent conclusion as the proposed method also performs well on this dataset.

According to the experiments, we can see that the proposed feature extractor has a strong generalization ability on different datasets. The limitation of the proposed feature extractor is that we do not train the feature extractor on different target datasets. However, it achieves state-of-the-art performances on different datasets. It means that the proposed feature extractor which is only trained on the GPDSsynthetic dataset could capture the micro deformations between the genuine signatures and skilled forgeries.

5. Conclusion

In this paper, we proved that ordinary CNNs have the potential to extract micro deformations in the max-pooling operation for offline signature verification tasks. The discriminative information can be learned from the max-pooling operation, which captures the difference not only between the different users but also between the genuine signatures and skilled forgeries by fusing displacement features and pooling features.

For building the verification system, we first train a CNN based model on original signature to discriminate different users. Then, we extract the displacement features from the pre-trained CNN and train another CNN to discriminate the genuine signatures and skilled forgeries. After training the CNN, we take the trained model as a feature extractor to extract the fused features for training writer-dependent classifiers to build a verification system.

For future work, we consider fine-tuning the proposed feature extractor on different target datasets and applying the proposed method to the online signature verification systems. In addition, we plan to build an end-to-end verification system for different signature verification tasks.

Acknowledgment

This work was supported by JSPS KAKENHI (Grant Number JP18K11373 and JP17H06100) and China Scholarship Council (Grant Number 201706330078).

References

[1] A. Hamadene, Y. Chibani, One-class writer-independent offline signature verification using feature dissimilarity thresholding, IEEE Trans. Inf. Forensic Secur. 11 (6) (2016) 1226–1238. doi:10.1109/TIFS.2016.2521611.

[2] M. Diaz, M. A. Ferrer, D. Impedovo, M. I. Malik, G. Pirlo, R. Plamondon, A perspective analysis of handwritten signature technology, ACM Comput. Surv. 51 (6) (2019) 117:1–117:39. doi:10.1145/3274658.

[3] L. G. Hafemann, R. Sabourin, L. S. Oliveira, Offline handwritten signature verification-literature review, in: Proc. IPTA, 2017, pp. 1–8. doi:10.1109/IPTA.2017.8310112.

[4] M. I. Malik, M. Liwicki, From terminology to evaluation: Performance assessment of automatic signature verification systems, in: Proc. ICFHR, 2012, pp. 613–618. doi:10.1109/ICFHR.2012.205.

[5] Y. Zheng, B. K. Iwana, S. Uchida, Mining the displacement of max-pooling for text recognition, Pattern Recognit. 93 (2019) 558–569. doi:10.1016/j.patcog.2019.05.014.

[6] Y. Zheng, W. Ohyama, B. K. Iwana, S. Uchida, Capturing micro deformations from pooling layers for offline signature verification, in: Proc. ICDAR, 2019, pp. 1111–1116. doi:10.1109/ICDAR.2019.00180.

[7] M. B. Yılmaz, B. Yanıkoğlu, Score level fusion of classifiers in off-line signature verification, Inf. Fusion 32 (2016) 109–119. doi:10.1016/j.inffus.2016.02.003.

[8] N. Otsu, A threshold selection method from gray-level histograms, IEEE Trans. Syst., Man, Cybern. 9 (1) (1979) 62–66. doi:10.1109/TSMC.1979.4310076.

[9] L. G. Hafemann, R. Sabourin, L. S. Oliveira, Learning features for offline handwritten signature verification using deep convolutional neural networks, Pattern Recognit. 70 (2017) 163–176. doi:10.1016/j.patcog.2017.05.012.

[10] M. R. Pourshahabi, M. H. Sigari, H. R. Pourreza, Offline handwritten signature identification and verification using contourlet transform, in: Proc. SoCPaR, 2009, pp. 670–673. doi:10.1109/SoCPaR.2009.132.

[11] M. A. Ferrer, J. B. Alonso, C. M. Travieso, Offline geometric parameters for automatic signature verification using fixed-point arithmetic, IEEE Trans. Pattern Anal. Mach. Intell. 27 (6) (2005) 993–997. doi:10.1109/TPAMI.2005.125.

[12] M. A. Ferrer, J. F. Vargas, A. Morales, A. Ordonez, Robustness of offline signature verification based on gray level features, IEEE Trans. Inf. Forensic Secur. 7 (3) (2012) 966–977. doi:10.1109/TIFS.2012.2190281.

[13] L. G. Hafemann, L. S. Oliveira, R. Sabourin, Fixed-sized representation learning from offline handwritten signatures of different sizes, Int. J. Doc. Anal. Recognit. 21 (3) (2018) 219–232. doi:10.1007/s10032-018-0301-6.

[14] Z. Zhang, X. Liu, Y. Cui, Multi-phase offline signature verification system using deep convolutional generative adversarial networks, in: Proc. ISCID, Vol. 2, 2016, pp. 103–107. doi:10.1109/ISCID.2016.2033.

[15] Y. Zheng, Y. Zheng, W. Ohyama, D. Suehiro, S. Uchida, Ranksvm for offline signature verification, in: Proc. ICDAR, 2019, pp. 928–933. doi:10.1109/ICDAR.2019.00153.

[16] V. L. Souza, A. L. Oliveira, R. M. Cruz, R. Sabourin, A white-box analysis on the writer-independent dichotomy transformation applied to offline handwritten signature verification, Expert Syst. Appl. (2020) 113397doi:10.1016/j.eswa.2020.113397.

[17] L. Nanni, S. Ghidoni, S. Brahnam, Handcrafted vs. non-handcrafted features for computer vision classification, Pattern Recognit. 71 (2017) 158–172. doi:10.1016/j.patcog.2017.05.025.

[18] J. Ruiz-del Solar, C. Devia, P. Loncomilla, F. Concha, Offline signature verification using local interest points and descriptors, in: Proc. CIAPR, 2008, pp. 22–29. doi:10.1007/978-3-540-85920-8_3.

[19] E. N. Zois, L. Alewijnse, G. Economou, Offline signature verification and quality characterization using poset-oriented grid features, Pattern Recognit. (2016) 162–177doi:10.1016/j.patcog.2016.01.009.

[20] M. Okawa, Synergy of foreground-background images for feature extraction: Offline signature verification using fisher vector with fused kaze features, Pattern Recognit. (2018) 480–489doi:10.1016/j.patcog.2018.02.027.

[21] Y. Zheng, Y. Cai, G. Zhong, Y. Chherawala, Y. Shi, J. Dong, Stretching deep architectures for text recognition, in: Proc. ICDAR, 2015, pp. 236–240. doi:10.1109/ICDAR.2015.7333759.

[22] Z. Hu, J. Tang, Z. Wang, K. Zhang, L. Zhang, Q. Sun, Deep learning for image-based cancer detection and diagnosis- a survey, Pattern Recognit. 83 (2018) 134–149. doi:10.1016/j.patcog.2018.05.014.

[23] T. Young, D. Hazarika, S. Poria, E. Cambria, Recent trends in deep learning based natural language processing, IEEE Comput. Intell. Mag. 13 (3) (2018) 55–75. doi:10.1109/MCI.2018.2840738.

[24] W. Guo, H. Gao, J. Shi, B. Long, L. Zhang, B.-C. Chen, D. Agarwal, Deep natural language processing for search and recommender systems, in: Proc. SIGKDD, 2019, pp. 3199–3200. doi:10.1145/3292500.3332290.

[25] A. Soleimani, B. N. Araabi, K. Fouladi, Deep multitask metric learning for offline signature verification, Pattern Recognit. Lett. 80 (2016) 84–90. doi:10.1016/j.patrec.2016.05.023.

[26] S. Lai, L. Jin, Learning discriminative feature hierarchies for off-line signature verification, in: Proc. ICFHR, 2018, pp. 175–180. doi:10.1109/ICFHR-2018.2018.00039.

[27] Y. Zheng, B. K. Iwana, S. Uchida, Discovering class-wise trends of max-pooling in subspace, in: Proc. ICFHR, 2018, pp. 98–103. doi:10.1109/ICFHR-2018.2018.00026.

[28] M. A. Ferrer, M. Diaz, C. Carmona-Duarte, A. Morales, A behavioral handwriting model for static and dynamic signature synthesis, IEEE Trans. Pattern Anal. Mach. Intell. 39 (6) (2016) 1041–1053. doi:10.1109/TPAMI.2016.2582167.

[29] M. K. Kalera, S. Srihari, A. Xu, Offline signature verification and identification using distance statistics, Int. J. Pattern Recognit. Artif. Intell. 18 (07) (2004) 1339–1360. doi:10.1142/S0218001404003630.

[30] A. Soleimani, K. Fouladi, B. N. Araabi, Utsig: A persian offline signature dataset, IET Biom. 6 (1) (2016) 1–8. doi:10.1049/iet-bmt.2015.0058.

[31] S. Pal, A. Alaei, U. Pal, M. Blumenstein, Performance of an off-line signature verification method based on texture features on a large indic-script signature dataset, in: Proc. DAS, 2016, pp. 72–77. doi:10.1109/DAS.2016.48.

[32] L. S. de Souza, H. Hino, K. Fukui, 3d object recognition with enhanced grassmann discriminant analysis, in: Proc. ACCV, 2016, pp. 345–359. doi:10.1007/978-3-319-54526-4_26.

[33] V. L. Souza, A. L. Oliveira, R. Sabourin, A writer-independent approach for offline signature verification using deep convolutional neural networks features, in: Proc. BRACIS, 2018, pp. 212–217. doi:10.1109/BRACIS.2018.00044.

[34] L. G. Hafemann, R. Sabourin, L. S. Oliveira, Analyzing features learned for offline signature verification using deep cnns, in: Proc. ICPR, 2016, pp. 2989–2994. doi:10.1109/ICPR.2016.7900092.

[35] Y. Serdouk, H. Nemmour, Y. Chibani, Handwritten signature verification using the quad-tree histogram of templates and a support vector-based artificial immune classification, Image Vis. Comput. 66 (2017) 26–35. doi:10.1016/j.imavis.2017.08.004.

[36] K. Barkoula, E. N. Zois, E. Zervas, G. Economou, Off-line signature verification based on ordered grid features: An evaluation., in: Proc. AFHA, 2013, pp. 36–40. doi:10.1.1.375.2181.

[37] J. Hu, Y. Chen, Offline signature verification using real adaboost classifier combination of pseudo-dynamic features, in: Proc. ICDAR, 2013, pp. 1345–1349. doi:10.1109/ICDAR.2013.272.

[38] Y. Guerbai, Y. Chibani, B. Hadjadji, The effective use of the one-class svm classifier for handwritten signature verification based on writer-independent parameters, Pattern recognit. 48 (1) (2015) 103–113. doi:10.1016/j.patcog.2014.07.016.

[39] L. G. Hafemann, R. Sabourin, L. S. Oliveira, Writer-independent feature learning for offline signature verification using deep convolutional neural networks, in: Proc. IJCNN, 2016, pp. 2576–2583. doi:10.1109/IJCNN.2016.7727521.

[40] Z.-J. Xing, F. Yin, Y.-C. Wu, C.-L. Liu, Offline signature verification using convolution siamese network, in: Proc. ICGIP, Vol. 10615, 2018, p. 106151I. doi:10.1117/12.2303380.

[41] D. G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2) (2004) 91–110.

[42] H. Bay, T. Tuytelaars, L. V. Gool, Surf: speeded up robust features, in: Proc. of the ECCV, pp. 404–417.

[43] O. Mersa, F. Etaati, S. Masoudnia, B. N. Araabi, Learning representations from persian handwriting for offline signature verification, a deep transfer learning approach, arXiv preprint arXiv:1903.06249doi:10.1109/PRIA.2019.8785979.

[44] M. B. Yılmaz, K. Öztürk, Recurrent binary patterns and cnns for offline signature verification, in: Proc. FTC, 2019, pp. 417–434. doi:10.1007/978-3-030-32523-7_29.

[45] S. Masoudnia, O. Mersa, B. N. Araabi, A.-H. Vahabie, M. A. Sadeghi, M. N. Ahmadabadi, Multi-representational learning for offline signature verification using multi-loss snapshot ensemble of cnns, Expert Syst. Appl. 133 (2019) 317–330. doi:10.1016/j.eswa.2019.03.040.

[46] P. Maergner, N. Howe, K. Riesen, R. Ingold, A. Fischer, Offline signature verification via structural methods: Graph edit distance and inkball models, in: Proc. ICFHR, 2018, pp. 163–168. doi:10.1109/ICFHR-2018.2018.00037.

[47] P. Maergner, V. Pondenkandath, M. Alberti, M. Liwicki, K. Riesen, R. Ingold, A. Fischer, Offline signature verification by combining graph edit distance and triplet networks, in: Proc. S+SSPR, 2018, pp. 470–480. doi:10.1007/978-3-319-97785-0_45.

[48] P. Maergner, N. R. Howe, K. Riesen, R. Ingold, A. Fischer, Graph-based offline signature verification, arXiv preprint arXiv:1906.10401.

[49] R. Kumar, L. Kundu, B. Chanda, J. Sharma, A writer-independent off-line signature verification system based on signature morphology, in: Proc. IITM, 2010, pp. 261–265. doi:10.1145/1963564.1963610.

[50] E. N. Zois, M. Papagiannopoulou, D. Tsourounis, G. Economou, Hierarchical dictionary learning and sparse coding for static signature verification, in: Proc. of the CVPR workshops, 2018, pp. 432–442. doi:10.1109/CVPRW.2018.00084.

[51] E. N. Zois, D. Tsourounis, I. Theodorakopoulos, A. L. Kesidis, G. Economou, A comprehensive study of sparse representation techniques for offline signature verification, IEEE Trans. on Biom., Behav., and Ident. Sci. 1 (1) (2019) 68–81. doi:10.1109/TBIOM.2019.2897802.

[52] E. N. Zois, A. Alexandridis, G. Economou, Writer independent offline signature verification based on asymmetric pixel relations and unrelated training-testing datasets, Expert Syst. Appl. 125 (2019) 14–32. doi:10.1016/j.eswa.2019.01.058.

[53] M. Okawa, From bovw to vlad with kaze features: Offline signature verification considering cognitive processes of forensic experts, Pattern Recognition Letters 113 (2018) 75–82. doi:10.1016/j.patrec.2018.05.019.

[54] A. Soleimani, K. Fouladi, B. N. Araabi, Persian offline signature verification based on curvature and gradient histograms, in: Proc. ICCKE, 2016, pp. 147–152. doi:10.1109/ICCKE.2016.7802131.

[55] M. Diaz, M. A. Ferrer, R. Sabourin, Approaching the intra-class variability in multi-script static signature evaluation, in: Proc. ICPR, 2016, pp. 1147–1152. doi:10.1109/ICPR.2016.7899791.

[56] L. v. d. Maaten, G. Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (Nov) (2008) 2579–2605.