# Hybrid User-Independent and User-Dependent Offline Signature Verification with a Two-Channel CNN

Mustafa Berkay YILMAZ
berkayyilmaz@akdeniz.edu.tr

Kağan ÖZTÜRK
kaganozturk1992@gmail.com

Akdeniz University, Antalya, Türkiye

## Abstract

*Signature verification task needs relevant signature representations to achieve low error rates. Many signature representations have been proposed so far. In this work we propose a hybrid user-independent/dependent offline signature verification technique with a two-channel convolutional neural network (CNN) both for verification and feature extraction. Signature pairs are input to the CNN as two channels of one image, where the first channel always represents a reference signature and the second channel represents a query signature. We decrease the image size through the network by keeping the convolution stride parameter large enough. Global average pooling is applied to decrease the dimensionality to 200 at the end of locally connected layers.*

*We utilize the CNN as a feature extractor and report 4.13% equal error rate (EER) considering 12 reference signatures with the proposed 200-dimensional representation, compared to 3.66% of a recently proposed technique with 2048-dimensional representation using the same experimental protocol. When the two methods are combined at score level, more than 50% improvement (1.76% EER) is achieved demonstrating the complementarity of them. Sensitivity of the model to gray-level and binary images is investigated in detail. One model is trained using gray-level images and the other is trained using binary images. It is shown that the availability of gray-level information in train and test data decreases the EER* e.g. *from 11.86% to 4.13%.*

## 1. Introduction

Signature verification is the task of determining whether a query signature is signed by the claimed identity or someone else. It can be online (dynamic) or offline (handwritten). Forgery signatures can be broadly divided into two categories as random and skilled forgeries. Skilled forgeries are signed by forgers after an enough effort of training on genuine samples. Random forgeries are signed without any information about the claimed identity and they can usually be detected with very small error rates.

Our focus in this work is offline signature verification where the query is either genuine signature or skilled forgery. Format of the signature image can be binary, gray-level or color. To the best of our knowledge, no public color signature database is available. However the distinction between binary and gray-level image is critical, as a gray-level image carries more information than the binary one such as the pressure of the stroke.

Performance of signature verification systems is generally measured by equal error rate (EER) which is the error rate when the false accept (FA) and false reject (FR) rates are equal. If the EER is not reported, distinguishing error rate (DER) can be calculated which is simply the average of FA and FR.

## 2. Related works

There are many methods proposed for the problem of offline signature verification. However, a direct comparison between these works is usually not possible because of experimental protocol inconsistencies in various aspects. Differences include the databases, image formats (gray-level or binary), subsets of the databases reserved for training and testing purposes, selection of reference samples, number of reference samples, to use skilled forgeries in training or not, to use random forgeries in testing or not, calculation of decision thresholds, hyper-parameter choice and many other factors. In order to be able to directly compare different works, one should have their implementations or at least the outputs at which level to compare (preprocessing, feature, classifier, score or decision).

Recent works that utilize pre-determined (handcrafted) signature representations achieve EER of 7%, with binary GPDS-160 [3] subset using 12 reference signatures per subject [17]. Scores of user-based and user-independent classifiers utilizing different features such as histogram of oriented gradients (HoG), local binary patterns (LBP) and scale invariant feature transform (SIFT) are fused to obtain

a final decision on query signatures [17].

Hu *et al*. consider gradient based LBP, statistical gray-level co-occurrence matrix (SGLCM) and simplified HoG for user-independent offline signature verification [7]. Random forest is utilized as the classifier. A fusion of the proposed features achieves 7.42% EER when random 140 subjects are tested in gray-level GPDS with 12 references per user.

Sparse dictionary learning and coding are employed to provide a feature space for offline signature verification [21]. K-SVD dictionary learning algorithm is employed in order to create a writer-oriented lexicon. Gray-level GPDS-300 is used to evaluate the method. When 5 references are considered per user, 7.21% EER is achieved. If user-based ideal decision thresholds are calculated from test scores, EER drops to 2.70%.

One of the early works on deep offline signature verification extracts conventional features such as width, height, tri-surface, six-fold surface, modified direction feature to feed into a deep belief network [12]. However, results are reported only using conventional classifiers such as feed forward neural network and support vector machine (SVM). Khalajzadeh et al. use CNN for Persian signature recognition to discriminate between different subjects' signatures [9].

The first work to discuss and report results on deep offline signature verification in the presence of skilled forgeries considers a basic deep structure called PCANet [16]. Signature representation is learnt by PCANet from a separate set of users. EER of around 20% is reported with 5 reference signatures using binary GPDS-160.

A novel classification method called deep multitask metric learning (DMML) is proposed for offline signature verification [13]. Using the idea of multitask and transfer learning, DMML trains a distance metric for each class together with other classes simultaneously. DMML has a structure with a shared layer acting as a user-independent approach, followed by separated layers which learn user-dependent factors. It is compared with SVM on various offline signature datasets including gray GPDS- 960 using HoG and Discrete Radon Transform (DRT) features. Around 0.5% improvement is reported with skilled forgeries, compared to [4]. Their best result of 15% error on this dataset is obtained with 75 users and 10 references per user.

Comparing image pairs via CNN is a well-studied topic. Siamese network is commonly used for this task. In siamese network there are two branches that share the same parameters. This architecture takes two input images, one image for each branch, concatenate the outputs of these branches and apply some fully connected layer to decide whether these two images are similar or not. Siamese network is applied to the problem of online signature verification in an early work [1]. During training the two sub-networks extract features from two signatures, while the joining neuron measures the distance between the two feature vectors. Verification consists of comparing an extracted feature vector with a stored feature vector for the signer. Signatures closer to this stored representation than a chosen threshold are accepted, all other signatures are rejected as forgeries.

Siamese network is recently applied to user-independent offline signature verification task [2]. Binary GPDS-300 database is used for evaluation among some other databases. Randomly selected 150 subjects are used to train the network. For testing, there is no explicit notion of reference signature. From each of the writers, 276 genuine-genuine and genuine- forgery signature pairs are selected to report an EER of 23.17%.

Signature image representations are learned in a user-independent format by Hafemann *et al*. using CNN [6]. A novel formulation is proposed to include knowledge of skilled forgeries from a subset of users in the feature learning process. This aims to capture visual cues that distinguish genuine signatures and forgeries regardless of the user. Main objective of the CNN is to discriminate between the users in the development set. To drive the features to be good in distinguishing skilled forgeries, a multi-task framework is proposed by considering two terms in the cost function for feature learning. The first term drives the model to distinguish between different users, while the second term drives the model to distinguish between genuine signatures and skilled forgeries. After training, the CNN is used to extract feature representations of 2048 dimensions for signatures from the test set and train user-dependent classifiers. For the evaluation, GPDS960-gray [4] is utilized with gray-level signature images. Last 531 subjects are considered for training whereas first 160 subjects are considered for testing. When SVM with radial basis function (RBF) learns user-dependent features, 3.61% EER is reported with 12 reference signatures per user. If user-based ideal thresholds are calculated from the test scores, EER drops to 1.72%. This work is useful for user-dependent verification, as the output of the CNN just represents one of the subjects in the training set as either genuine or forgery.

Zhang *et al*. use deep convolutional generative adversarial networks to learn signature representations from training data [20]. After training the network, they use the discriminator's convolutional features from all layers as signature features and then train hybrid user-dependent/user-independent classifier. Results show that user-dependent approach performs better than user-independent approach. They report around 14% DER with gentle Adaboost classifier on gray-level GPDS-960 database with a combination of user-dependent/independent approaches.

Other than offline signature verification, CNN has been used to learn similarity measure on image patches for stereo matching by Zbontar and LeCun [15]. Two types of net-

works are examined, one tuned for speed and the other for accuracy. Fast architecture is a siamese network. The similarity score is obtained by extracting a vector from each of the two input patches and computing the cosine similarity between them. Accurate architecture is derived from the first by replacing the cosine similarity measure with a number of fully connected layers. The last fully-connected layer produces a single number which, after being transformed with the sigmoid nonlinearity, is interpreted as the similarity score between the input patches. The fast architecture can compute the disparity maps up to 90 times faster than the accurate architecture with only a small increase in error.

Zagoruyko and Komodakis offer a two-channel CNN for comparing two images [18]. Input of this network is created via concatenating two images as a two-channel image. Unlike the siamese network there are no shared branches thus it is faster to train, but at test time it is more expensive than siamese network. The reason is in a siamese network one can extract features via shared branches for each image and then create all combination of pairs to give decision applying some fully connected layers. In two-channel network it can not be done since it starts applying operations to concatenated image from first layer. Although this process makes the network slower at test time, it is faster to train and it can achieve better accuracy.

In a more recent work, some further improvements are offered for using two-channel CNNs [19]. Improvements include using a global average pooling (GAP) layer and normalized cross-correlation with a similar model proposed in [18]. GAP layer is placed only once before the softmax layer [11]. Max-pooling layers are replaced by convolutions of increasing stride [14]. In traditional CNNs, the feature maps of last convolutional layer are vectorized and fed into fully connected layer. GAP layer take the average of each feature map. While fully connected layer has a lot of parameters, there is no parameter in the GAP layer. Thus it has great regularization effect and can avoid overfitting. In this work we extract the feature maps after the GAP layer to learn user-based representations, in addition to [19].

# 3. Proposed method

Siamese networks are used for distance metric learning by utilizing the same set of weights in both sub-networks. However they do not learn to treat reference and query signature samples in different ways. During the training process, reference should somehow be dictated to the network as the ground truth. Query input may or may not look like the reference and it is either genuine or forgery.

In this work, we propose using a two-channel CNN to perform concurrent user-independent verification and signature feature extraction instead of metric learning. First image channel is always used to input a (genuine) reference signature and second channel is used to input a query signature. Our model allows concurrent feature extraction and user-independent verification at a single forward propagation, allowing robust and efficient verification. Extracted features can then be used to train user-dependent models if there are enough reference samples for the user of interest.

(a) Original signature (b) Preprocessed form

Figure 1: An example signature image from GPDS-960 and its preprocessed form.

## 3.1. Two-channel CNN

### 3.1.1 Preprocessing

CNN needs fixed size images as its input, so signature images should somehow be resized to be in a uniform size. In this work we apply a simple preprocessing. We first invert the gray-level values by extracting them from 255, so that background is represented by 0 values instead of 255 values of a white paper in original image. Next we binarize the signature image temporarily to detect and eliminate small connected components (components with less than 40 pixels) with an assumption that so small components are simply noise.

Before input to the CNN, query signature on the second channel is always aligned to the reference signature on the first channel. Alignment is achieved by applying different rotations and translations to the query, and choosing the transformation which gives the lowest difference metric. As the metric; we extract basic 8-neighbor LBP histograms both from the reference and the aligned query and look at the Euclidean distance between the histograms.

We then crop the bounding box by finding the minimum and maximum $x$ and $y$ coordinates of on-pixels (pencil strokes) and deleting the empty rows and columns before and after them. At last we resize the image to $100 \times 150$. An example original signature image and its preprocessed form are shown in Figure 1.

### 3.1.2 CNN structure

We learn how to decide if a query signature Q is genuine or not in existence of a reference R (known to be genuine) with the help of a two-channel CNN $\phi(R, Q)$. We use two similar architectures for user-independent gray-level and binary signature verification. For both architectures, input size is $100 \times 150 \times 2$ containing the reference and the query.

In contrast to traditional CNN models, our gray-level network is similar to the network architecture proposed in [14]. We replace max-pooling layers by convolutional layers of increasing stride. It is shown in [14] that, for $p$-norm

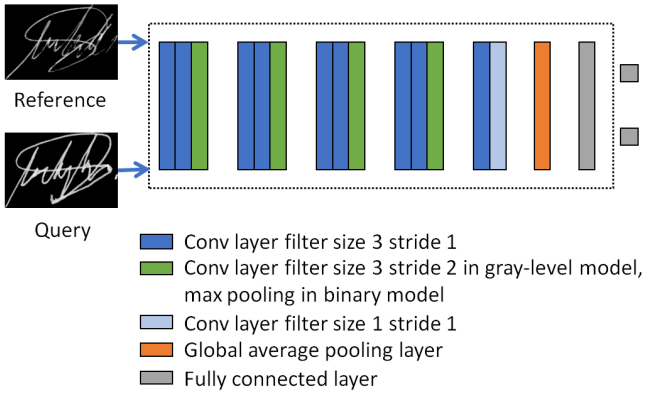


Figure 2: Two-channel CNN model proposed in this work.

subsampling applied to the feature map $f$ produced by some layer of a CNN, max-pooling is just a special case of using convolutional layers for dimensionality reduction where $p \to \infty$.

The network $\phi(R, Q)$ consists of 14 convolutional layers, 1 global average pooling layer and 1 fully connected layer. It is regularized by 5 dropout layers of probability 0.5. At the end there is a two-way softmax layer to decide whether the input is a genuine-genuine or genuine-forgery pair. We use Adam optimizer [10] to train the network. Batch normalization [8] is utilized before each rectified linear unit (ReLU) activation function.

We observe that with binary images, the above-mentioned CNN starts overfitting after some iterations. For this reason, we train another network with binary images after training and testing the gray-level model. We reduce the capacity of the network by replacing the convolutional layers of stride 2 with max-pooling layers along with the number of hidden units of some layers. Two-channel CNN model used in this work is illustrated in Figure 2. Details of CNN structures for gray-level and binary models are given in Table 1.

### 3.2. User-independent verification

User-independent verification (UI) is performed with the goal of obtaining the probability of a query signature $Q$ belonging to user $y$ in the presence of a reference signature $R_n^y \in \mathbf{R}^y$, where $\mathbf{R}^y$ is the reference set of user $y$. We show this probability as $P(y|R_n^y, Q)$. Output of the CNN with a reference and a query as the input is assumed to estimate this probability:

$$P(y|R_n^y, Q) \approx \phi(R_n^y, Q). \tag{1}$$

Suppose that $N = |\mathbf{R}^y|$ is the cardinality of set $\mathbf{R}^y$. We have as many scores from the CNN as $N$, input to match the query $Q$ with each reference separately. We calculate the average score to make a decision on $Q$ as follows:

Table 1: Gray-level and binary CNN models. Convolutions $C_3$, $C_6$, $C_9$ and $C_{12}$ of gray-level model are replaced with MaxPooling in binary model. $C_4$, $C_5$, $C_7$, $C_8$, $C_{10}$ and $C_{11}$ have the more number of hidden units in gray-level model, and the less in binary model.

| Layers | Hidden units | Filter size | Stride |
|---|---|---|---|
| Convolution $C_1$ & $C_2$ | 30 | 3 | 1 |
| $C_3$ or MaxPooling | 30 ($C_3$) | 3 | 2 |
| Dropout (0.5) | | | |
| Convolution $C_4$ & $C_5$ | 60 or 30 | 3 | 1 |
| $C_6$ or MaxPooling | 60 ($C_6$) | 3 | 2 |
| Dropout (0.5) | | | |
| Convolution $C_7$ & $C_8$ | 100 or 60 | 3 | 1 |
| $C_9$ or MaxPooling | 100 ($C_9$) | 3 | 2 |
| Dropout (0.5) | | | |
| Convolution $C_{10}$ & $C_{11}$ | 150 or 100 | 3 | 1 |
| $C_{12}$ or MaxPooling | 150 ($C_{12}$) | 3 | 2 |
| Dropout (0.5) | | | |
| Convolution $C_{13}$ | 200 | 3 | 1 |
| Convolution $C_{14}$ | 200 | 1 | 1 |
| GAP | | | |
| FullyConnected | 200 | | |
| Dropout (0.5) | | | |
| FullyConnected (softmax) | 2 | | |

$$P_{ui}(y|\mathbf{R}^y, Q) \approx \sum_{n=1}^{N} \phi(R_n^y, Q)/N. \tag{2}$$

If $P_{ui}(y|\mathbf{R}^y, Q) \geq \theta_{ui}$, then Q is decided as a genuine signature where $\theta_{ui}$ is a decision threshold for UI verification. UI approach has the advantage that no user-specific model has to be trained and stored so that the system can be queried infinitely many individual test subjects as long as some reference set is provided along with. There is no concern of model management and update when a user provides new reference signatures over time. Another advantage is that when the number of reference signatures is only one, we can still obtain an effectual verification score.

### 3.3. User-dependent verification

User-dependent verification (UD) is performed by training UD classifiers. Signature representations are obtained as the output of the GAP layer before the fully-connected layer of CNN with a reference and a query as the input. In this case we have as many representations for a query signature as the number of references.

#### 3.3.1 Feature extraction

Output of the global average pooling layer $\phi_{GAP}$ is used to represent a query image $Q$ paired with $R_n^y$ of the claimed

identity $y$ as $\phi_{GAP}(R_n^y, Q)$. We have as many representations (features) for a query signature as $N$ where the feature set becomes $F_Q = \cup_{n=1}^{N} \phi_{GAP}(R_n^y, Q)$. Note that the dimensionality is 200 after GAP layer.

#### 3.3.2 UD model training

We utilize SVM with RBF kernel to train UD models. During training, all $N \times (N-1)$ genuine-genuine inter-reference pairs are used as positive samples where the second reference pretends a genuine query:

$$\mathbf{S}^{+} = \cup_{n=1}^{N} \phi_{GAP}(R_n^y, R_m^y) \forall m \neq n. \quad (3)$$

Genuine-forgery pairs from other users are randomly selected as negative training samples:

$$\mathbf{S}^{-} = \bigcup_{y'} \bigcup_{m=1}^{M} \phi_{GAP}(R_m^{y'}, Q^{y'}) \text{ for some } y' \neq y, \quad (4)$$

where $Q^{y'}$ is known to be a forgery signature of some user $y'$ other than $y$. Note that we can always assume that we have some training subjects for whom we have both genuine and forgery samples. This way SVM can learn a tight decision boundary between the pair representations of current user and other users.

During testing, we have $N$ different representations for an unknown query signature $Q$, so we have as many scores from the SVM decision function of user $y$ ($f^y(.), \mathbb{R}^{200} \Rightarrow \mathbb{R}$) as $N$, matching each reference separately. We obtain separate scores for each of such representations and find the average score to make decision on $Q$:

$$P_{ud}(y|\mathbf{R}^{\mathbf{y}}, Q) \approx \sum_{n=1}^{N} f^y(\phi_{GAP}(R_n^y, Q))/N. \quad (5)$$

If $P_{ud}(y|\mathbf{R}^{\mathbf{y}}, Q) \geq \theta_{ud}$ then $Q$ is decided as a genuine signature where $\theta_{ud}$ is UD decision threshold.

We normalize the features in a user-based manner by dividing each feature to the maximum value (scalar) $S_{max}$ observed in training samples $\mathbf{S}^{+} \cup \mathbf{S}^{-}$. Before training with SVM, all values are divided to $S_{max}$. Similarly, all query features are divided to $S_{max}$ during testing.

### 3.4. Concurrent user-independent/dependent verification

We also explore the results obtained by score level fusion of UI and UD classifiers. This corresponds to a classifier combination of UI neural network and UD SVM. We have linear combination of the scores obtained in sections 3.2 and 3.3 as follows:

$$P_{uid}(y|\mathbf{R}^{\mathbf{y}}, Q) = \alpha P_{ui}(y|\mathbf{R}^{\mathbf{y}}, Q) + (1-\alpha) P_{ud}(y|\mathbf{R}^{\mathbf{y}}, Q), \quad (6)$$

where the weight $\alpha \in [0, 1]$ is a real number learnt from a validation set, as detailed in Section 4.2.

## 4. Experimental results

### 4.1. Database

We use the publicly available largest signature database GPDS960-gray to evaluate our system [4]. We investigate the sensitivity of the proposed method to gray-level and binary signature images in detail. Database is manually converted into binary (from gray-level) to investigate the effect of image format. In summary we use both gray-level and binary signatures in training and testing.

GPDS960-gray signature database consists of 881 users, 21144 genuine signatures and 26317 imitations. The signatures are in png format and have been scanned at 600 dpi. In Section 4.2 we explicitly provide the first and last subjects' IDs that constitute different subsets. Each subject has 24 genuine samples and at most 30 forgery samples. Number of forgery samples is less than 30 for a few subjects.

### 4.2. Experimental protocol

#### 4.2.1 Training and validation sets of two-channel CNN

We use the last 475 subjects of GPDS-960 (with respect to their user IDs inclusive [460 - 960]) for training the proposed multi-channel CNN, defined as set $\tau$. For validation, we use the previous 100 subjects (with IDs inclusive [358 - 459]), defined as $V_1$. CNN structure and hyper-parameters are determined by looking at the accuracy of the CNN on $V_1$.

We consider genuine-genuine and genuine-forgery pairs from each training subject. For each subject we have $24 \times 23 = 552$ genuine-genuine pairs and at most $24 \times 30 = 720$ genuine-forgery pairs. During training, we randomly select 552 of genuine-forgery pairs to prevent overfitting to training subjects and make the training process faster. In total we have $552 \times 2 \times 475$ training pairs. Both $\tau$ and $V_1$ are either gray-level or binary during training.

#### 4.2.2 Selecting UD model hyper-parameters

We use a second validation set consisting of the previous 146 subjects (with IDs [205 - 357]) to select user-dependent SVM hyper-parameters (cost ($C$) of error and $\gamma$ of RBF kernel), defined as $V_2$. We learn the SVM parameters by training UD SVMs with the first 5 genuine samples as reference set and testing with the remaining genuine samples and skilled forgeries, for each user. We calculate the EER

for each hyper-parameter setting and select the SVM parameters that minimize the EER. $V_2$ is utilized as either gray-level or binary, whichever the test set is.

#### 4.2.3 Selecting the combination parameter $\alpha$

We use $V_2$ to learn the combination weight $\alpha$ and combine our UI and UD model scores. Similarly we use $V_2$ to learn combination weights of our final score (UD model scores trained with two-channel CNN's 200-dimensional features combined with UI CNN score) and UD scores obtained by 2048-dimensional features proposed in [6], at their score level.

#### 4.2.4 Selecting signature pair representations from other users

In UD model, genuine-forgery signature pairs belonging to other subjects are utilized as negative samples. This way the characteristics of one subject can be distinguished from other subjects. We exclude the subject of interest from the test set and get 5 genuine-forgery pair samples from each of the remaining subjects. This way it is guaranteed that no negative training sample is used from the same subject. Another option is to use one-class SVMs [5] which is not considered in this work.

#### 4.2.5 Test set

We evaluate the performance of UI CNN, UD SVM and combination of them using the test set $T$. Test set consists of the first 160 subjects of GPDS-960 (with IDs [2 - 204]). It is divided into two disjoint parts as $T_1$ and $T_2$. $T_1$ represents the set of candidate genuine reference samples whereas $T_2$ represents the unseen query samples.

We randomly generate two partitions of genuine samples, each having 12 samples per user. For each partition, we randomly select $N$ genuine samples three times. In total, we perform 6 tests for each such $T_1$ and $T_2$ distinction.

This protocol is repeated with UI CNN and UD SVM but the partitions and reference samples are randomly determined once ahead of time for both approaches, and for all users. We consider N=1, 5 and 12 in this work. For N=12, reference set is exactly the overall partition so there is no random reference set selection. Remaining 12 genuine samples and all skilled forgeries (varying between 24 and at most 30 per user) constitute $T_2$ as unseen query samples to measure the performance of the system. We do not consider any random forgeries during testing.

$T$ is utilized either in gray-level or binary form. We test gray-level and binary CNN models with both kinds of formats to investigate the effect of image format on the verification performance. Separation of the database into subsets is shown in Figure 3.

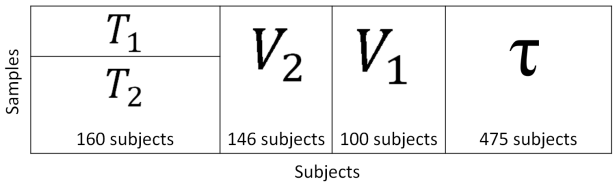


Figure 3: Separation of the database into subsets.

#### 4.2.6 Calculation of EER

We first investigate global and then user-based thresholds for user-independent verification ($\theta_{ui}$) and user-dependent verification ($\theta_{ud}$). These thresholds are determined directly from the test scores. In the third scenario, $\theta_{ui}$ and $\theta_{ud}$ are calculated from $V_2$ to explore the adaptability of it on different test subjects. In that case we report DER (average of FA and FR). Generalization of global verification EER threshold to another dataset is a challenge. Estimation of user-based ideal thresholds is itself a research topic and is not explored in this work. The most realistic choice is thus the third threshold which is calculated from $V_2$.

### 4.3. Results

Results with the described test protocol where $\tau$ and $V_1$ (training) are both gray-level can be found in Table 2 for UI and UD, Table 3 for the combination of UI and UD. Results with binary $\tau$ and $V_1$ can be found in Table 4 for UI and UD, Table 5 for the combination of them. Note that UD model can not be trained with one reference because obtaining a genuine-genuine training pair is impossible with the exception of a self-pair when $N = 1$.

We obtain the best results when both the training and test images are in gray-level, as such images carry more information compared to binary. In most cases, UD model provides better results. However even without dealing with UD model training, UI alone can perform well in particular cases.

UI model seems to be robust against the number of references; even with 1 reference it can provide acceptable results for gray-level model. UD model performs better as the number of references increases. As expected, user-based thresholds always give the best results. However, obtaining such thresholds is a difficult problem itself. Using a global EER threshold gives the second best results in most cases. In order to measure the generalizability of such a threshold, we also explore calculating a global threshold from verification data. In this case results tend to worsen a bit.

In order to compare the results with a recently proposed similar work, UD results with the features extracted using the CNN model (SigNet-F) proposed in [6] are shown in Table 6. UD model is trained as described in Section 3.3, only with the difference that single-input SigNet-F extracts the representations instead of the proposed two-channel CNN.

Table 2: Results with gray-level $\tau$ and $V_1$ for UI and UD.

| | $N$ | Global threshold ($V_2$) DER | | Global threshold($T$) EER | | User-based thresholds EER | |
|---|---|---|---|---|---|---|---|
| | | UI | UD | UI | UD | UI | UD |
| Gray $V_2$ and $T$ | 1 | $12.76 \pm 0.19\%$ | - | $8.74 \pm 0.34\%$ | - | $6.81 \pm 0.17\%$ | - |
| | 5 | $12.19 \pm 0.24\%$ | $5.96 \pm 0.26\%$ | $7.39 \pm 0.22\%$ | $6.52 \pm 0.68\%$ | $5.75 \pm 0.75\%$ | $4.72 \pm 0.33\%$ |
| | 12 | $12.00 \pm 0.05\%$ | $8.79 \pm 0.08\%$ | $7.20 \pm 0.24\%$ | $4.29 \pm 0.14\%$ | $5.78 \pm 0.67\%$ | $2.88 \pm 0.18\%$ |
| Binary $V_2$ and $T$ | 1 | $37.10 \pm 0.50\%$ | - | $32.74 \pm 0.44\%$ | - | $29.74 \pm 0.64\%$ | - |
| | 5 | $36.58 \pm 0.29\%$ | $34.49 \pm 0.42\%$ | $31.92 \pm 0.31\%$ | $23.49 \pm 0.65\%$ | $27.26 \pm 0.35\%$ | $19.65 \pm 0.42\%$ |
| | 12 | $36.63 \pm 0.03\%$ | $17.64 \pm 0.13\%$ | $31.22 \pm 0.42\%$ | $17.95 \pm 0.50\%$ | $26.80 \pm 1.07\%$ | $15.03 \pm 0.21\%$ |

Table 3: Results with gray-level $\tau$ and $V_1$ for the combination of UI and UD.

| | $N$ | Global threshold ($V_2$) DER | Global threshold($T$) EER | User-based thresholds EER |
|---|---|---|---|---|
| Gray$V_2$ and $T$ | 5 | $5.23 \pm 0.21\%$ | $5.38 \pm 0.14\%$ | $3.92 \pm 0.28\%$ |
| | 12 | $4.82 \pm 0.06\%$ | $4.13 \pm 0.31\%$ | $2.94 \pm 0.28\%$ |
| Binary$V_2$ and $T$ | 5 | $40.68 \pm 0.45\%$ | $21.57 \pm 0.35\%$ | $18.21 \pm 0.46\%$ |
| | 12 | $20.81 \pm 0.75\%$ | $18.08 \pm 0.43\%$ | $14.73 \pm 0.02\%$ |

Table 4: Results with binary $\tau$ and $V_1$ for UI and UD.

| | $N$ | Global threshold ($V_2$) DER | | Global threshold($T$) EER | | User-based thresholds EER | |
|---|---|---|---|---|---|---|---|
| | | UI | UD | UI | UD | UI | UD |
| Gray $V_2$ and $T$ | 1 | $32.11 \pm 0.58\%$ | - | $32.15 \pm 0.61\%$ | - | $28.69 \pm 0.69\%$ | - |
| | 5 | $30.52 \pm 0.46\%$ | $14.21 \pm 0.43\%$ | $30.38 \pm 0.44\%$ | $14.03 \pm 0.24\%$ | $25.90 \pm 0.60\%$ | $11.01 \pm 0.42\%$ |
| | 12 | $30.52 \pm 0.18\%$ | $13.44 \pm 0.30\%$ | $30.18 \pm 0.36\%$ | $11.15 \pm 0.22\%$ | $25.75 \pm 0.62\%$ | $8.30 \pm 0.08\%$ |
| Binary $V_2$ and $T$ | 1 | $25.87 \pm 0.59\%$ | - | $24.97 \pm 0.80\%$ | - | $21.22 \pm 0.77\%$ | - |
| | 5 | $24.45 \pm 0.37\%$ | $23.43 \pm 0.56\%$ | $22.32 \pm 0.36\%$ | $15.46 \pm 0.37\%$ | $18.95 \pm 0.34\%$ | $11.41 \pm 0.20\%$ |
| | 12 | $24.20 \pm 0.22\%$ | $12.33 \pm 0.05\%$ | $21.64 \pm 0.56\%$ | $12.14 \pm 0.10\%$ | $18.47 \pm 0.33\%$ | $9.31 \pm 0.26\%$ |

Table 5: Results with binary $\tau$ and $V_1$ for the combination of UI and UD.

| | $N$ | Global threshold ($V_2$) DER | Global threshold($T$) EER | User-based thresholds EER |
|---|---|---|---|---|
| Gray$V_2$ and $T$ | 5 | $14.20 \pm 0.43\%$ | $14.10 \pm 0.32\%$ | $10.85 \pm 0.39\%$ |
| | 12 | $13.86 \pm 0.23\%$ | $11.12 \pm 0.29\%$ | $8.26 \pm 0.08\%$ |
| Binary$V_2$ and $T$ | 5 | $23.30 \pm 0.55\%$ | $15.40 \pm 0.35\%$ | $11.31 \pm 0.21\%$ |
| | 12 | $12.15 \pm 0.13\%$ | $11.86 \pm 0.02\%$ | $9.22 \pm 0.15\%$ |

For comparison and combination, $V_2$ and $T$ are considered only as gray-level when SigNet-F is taken into account. We report the results obtained by the score-level combination of the two-channel CNN method's final score (UI combined with UD) with UD score of the SigNet-F representation [6] in Table 7. These results are obtained by training and testing the CNN models (SigNet-F and novel two-channel CNN) with gray-level images.

SigNet-F UD model provides better results with a 2048-dimensional representation but when the threshold is calculated from a validation set as it is done in a real-life scenario, proposed UI+UD combination score can give competing results only with a 200-dimensional representation. Proposed 200-dimensional representation even provide a better result when the threshold is determined from $V_2$ and $N = 5$. Total input size of two-channel CNN is $100 \times 150 \times 2$, less than $150 \times 220$ of single-channel SigNet-F. When the two methods are combined at score level, around 50% improvement is achieved over the state-of-the-art system [6].

## 5. Conclusions and future work

We propose a two-channel CNN model to take as the input a reference and a query signature. It gives comparable results to the state-of-art only with 200 features produced by GAP layer, compared to 2048 features obtained with a single-input SigNet-F CNN model [6]. It is possible to extract signature features and obtain UI score in a single forward pass of the CNN model. We fuse UI CNN score with UD SVM score to perform a robust verification. It is possible to get reasonable results even when $N = 1$, only with UI model (CNN output). Further combination of the proposed method's final score (UI + UD) with the UD score obtained by SigNet-F is explored. State-of-the-art results are achieved (more than 50% improvement, as low as 0.88% EER) with this combination. The two representations are shown to be complementary. In future it would be interesting to propose a CNN which takes reference and query input signatures and outputs a user ID plus a separate forgery indicator neuron. This way a single CNN may achieve the results obtained by the fusion of two different CNN representations.

Effect of image format either as binary or gray-level is investigated. Binary CNN model is simpler compared to the gray-level model. However, UI EER increases when the binary model is tested with gray-level signatures. This can easily be compensated by binarization of the input when the binary model is in use. Nevertheless in a scenario where gray-level and binary signature images are frequently encountered, both types of models can be trained beforehand to be used accordingly.

It is interesting that even when the EER is higher for binary UI model tested with gray-level images compared to binary test images, UD model can perform better than testing with binary images. In general if UD and UI models' combination is incorporated when working with binary images, there is an increase in EER *e.g.* from 4.13% to 11.86%; compared to working only with gray-level images. This difference is much more obvious for UI model (Table 2 gray-level test results versus other test results).

Table 6: UD results with the features extracted using SigNet-F CNN [6].

| $N$ | Global threshold ($V_2$) DER | Global threshold ($T$) EER | User-based thresholds EER |
|---|---|---|---|
| 5 | $5.81 \pm 0.63\%$ | $4.44 \pm 0.19\%$ | $2.66 \pm 0.40\%$ |
| 12 | $3.82 \pm 0.55\%$ | $3.66 \pm 0.58\%$ | $2.08 \pm 0.64\%$ |

Table 7: Score-level combination results of two-channel CNN method's final score (UI combined with UD) with UD score of the SigNet-F representation [6].

| $N$ | Global threshold ($V_2$) DER | Global threshold ($T$) EER | User-based thresholds EER |
|---|---|---|---|
| 5 | $2.90 \pm 0.31\%$ | $2.33 \pm 0.17\%$ | $1.16 \pm 0.21\%$ |
| 12 | $1.75 \pm 0.36\%$ | $1.76 \pm 0.37\%$ | $0.88 \pm 0.36\%$ |

## References


[1] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a "siamese" time delay neural network. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 737–744. Morgan-Kaufmann, 1994. 2

[2] S. Dey, A. Dutta, J. I. Toledo, S. K. Ghosh, J. Llados, and U. Pal. SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification. *arXiv:1707.02131*, July 2017. 2

[3] M. A. Ferrer, J. B. Alonso, and C. M. Travieso. Offline geometric parameters for automatic signature verification using fixed-point arithmetic. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):993–997, June 2005. 1

[4] M. A. Ferrer, J. F. Vargas, A. Morales, and A. Ordonez. Robustness of offline signature verification based on gray level features. *IEEE Transactions on Information Forensics and Security*, 7(3):966–977, June 2012. 2, 5

[5] Y. Guerbai, Y. Chibani, and N. Abbas. One-class versus bi-class svm classifier for off-line signature verification. In *2012 International Conference on Multimedia Computing and Systems*, pages 206–210, May 2012. 6

[6] L. G. Hafemann, R. Sabourin, and L. S. Oliveira. Learning features for offline handwritten signature verification using deep convolutional neural networks. *Pattern Recogn.*, 70(C):163–176, Oct. 2017. 2, 6, 8

[7] J. Hu, Z. Guo, Z. Fan, and Y. Chen. Offline signature verification using local features and decision trees. *International Journal of Pattern Recognition and Artificial Intelligence*, 31, 07 2016. 2

[8] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In D. Blei and F. Bach, editors, *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 448–456. JMLR Workshop and Conference Proceedings, 2015. 4

[9] H. Khalajzadeh, M. Mansouri, and M. Teshnehlab. Persian signature verification using convolutional neural networks. *International Journal of Engineering Research and Technology*, 1(2):7–12, 2016. 2

[10] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*, 2015. 4

[11] M. Lin, Q. Chen, and S. Yan. Network In Network. *arXiv:1312.4400*, Dec. 2013. 3

[12] B. Ribeiro, I. Gonçalves, S. Santos, and A. Kovacec. *Deep Learning Networks for Off-Line Handwritten Signature Recognition*, pages 523–532. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. 2

[13] A. Soleimani, B. N. Araabi, and K. Fouladi. Deep multitask metric learning for offline signature verification. *Pattern Recognition Letters*, 80(Supplement C):84 – 90, 2016. 2

[14] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for Simplicity: The All Convolutional Net. *arXiv:1412.6806*, Dec. 2014. 3

[15] J. Žbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *J. Mach. Learn. Res.*, 17(1):2287–2318, Jan. 2016. 2

[16] M. B. Yılmaz. *Offline signature verification with user-based and global classifiers of local features*. PhD thesis, Sabancı University, 2015. 2

[17] M. B. Yılmaz and B. Yanıkoğlu. Score level fusion of classifiers in off-line signature verification. *Information Fusion*, 32(Part B):109 – 119, 2016. SI Information Fusion in Biometrics. 1, 2

[18] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4353–4361, June 2015. 3

[19] S. Zagoruyko and N. Komodakis. Deep compare: A study on using convolutional neural networks to compare image patches. *Computer Vision and Image Understanding*, 2017. 3

[20] Z. Zhang, X. Liu, and Y. Cui. Multi-phase offline signature verification system using deep convolutional generative adversarial networks. *2016 9th International Symposium on Computational Intelligence and Design (ISCID)*, 02:103–107, 2016. 2

[21] E. N. Zois, I. Theodorakopoulos, D. Tsourounis, and G. Economou. Parsimonious coding and verification of offline handwritten signatures. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 636–645, July 2017. 2