Pattern Recognition Letters
journal homepage: www.elsevier.com

# SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification

Sounak Dey[a,**], Anjan Dutta[a], J. Ignacio Toledo[a], Suman K.Ghosh[a], Josep Lladós[a], Umapada Pal[b]

[a]*Computer Vision Center, Computer Science Dept., Universitat Autònoma de Barcelona, Edifici O, Campus UAB, 08193 Bellaterra, Spain*
[b]*Computer Vision and Pattern Recognition Unit, Indian Statistical Institute, 203, B. T. Road, Kolkata-700108, India*

ABSTRACT

Offline signature verification is one of the most challenging tasks in biometrics and document forensics. Unlike other verification problems, it needs to model minute but critical details between genuine and forged signatures, because a skilled falsification might only differ from a real signature by some specific kinds of deformation. This verification task is even harder in writer independent scenarios which is undeniably fiscal for realistic cases. In this paper, we model an offline writer independent signature verification task with a convolutional Siamese network. Siamese networks are twin networks with shared weights, which can be trained to learn a feature space where similar observations are placed in proximity. This is achieved by exposing the network to a pair of similar and dissimilar observations and minimizing the Euclidean distance between similar pairs while simultaneously maximizing it between dissimilar pairs. Experiments conducted on cross-domain datasets emphasize the capability of our network to handle forgery in different languages (scripts) and handwriting styles. Moreover, our designed Siamese network, named SigNet, provided better results than the state-of-the-art results on most of the benchmark signature datasets.

## 1. Introduction

Signature is one of the most popular and commonly accepted biometric hallmarks that has been used since the ancient times for verifying different entities related to human beings, *viz.* documents, forms, bank checks, individuals, etc. Therefore, signature verification is a critical task and many efforts have been made to remove the uncertainty involved in the manual authentication procedure, which makes *signature verification* an important research line in the field of machine learning and pattern recognition [1, 2]. Depending on the input format, signature verification can be of two types: (1) online and (2) offline. Capturing online signature needs an electronic writing pad together with a stylus, which can mainly record a sequence of coordinates of the electronic pen tip while signing. Apart from the writing coordinates of the signature, these devices are also capable of fetching the writing speed, pressure, etc., as additional information, which are used in the online verification process. On the other hand, the offline signature is usually captured by a scanner or any other type of imaging devices, which basically produces two dimensional signature images. As signature verification has been a popular research topic through decades and substantial efforts are made both on offline as well as on online signature verification purpose.

Online verification systems generally perform better than their offline counter parts [3] due to the availability of complementary information such as stroke order, writing speed, pressure,etc. However, this improvement in performances comes at the cost of requiring a special hardware for recording the pen-tip trajectory, rising its system cost and reducing the real application scenarios. There are many cases where authenticating offline signature is the only option such as check transaction and document verification. Because of its broader application area, in this paper, we focus on the more challenging task- automatic offline signature verification. Our objective is to propose a convolutional Siamese neural network model to discriminate the genuine signatures and skilled forgeries.

Offline signature verification can be addressed with (1) writer dependent and (2) writer independent approaches [4]. The

[**] Corresponding author: Tel.: +34 93 581 18 28; Fax: +34 93 581 16 70; *e-mail:* sdey@cvc.uab.es (Sounak Dey)

writer independent scenario is preferable over writer dependent approaches, as for a functioning system, a writer dependent system needs to be updated (retrained) with every new writer (signer). For a consumer based system, such as bank, where every day new consumers can open their account this incurs huge cost. Whereas, in writer independent case, a generic system is built to model the discrepancy among the genuine and forged signatures. Training a signature verification system under a writer independent scenario, divides the available signers into train and test sets. For a particular signer, signatures are coupled as *similar* (genuine, genuine) or *dissimilar* (genuine, forged) pairs. From all the tuples of a single signer, equal number of tuples similar and dissimilar pairs are stochastically selected for balancing the number of instances. This procedure is applied to all the signers in the train and test sets to construct the training and test examples for the classifier.

In this regard a signature verifier can be efficiently modelled by a Siamese network which consists of twin convolutional networks accepting two distinct signature images coming from the tuples that are either *similar* or *dissimilar*. The constituting convolutional neural networks (CNN) are then joined by a cost function at the top, which computes a distance metric between the highest level feature representation on each side of the network. The parameters between this twin networks are shared, which in turns guarantees that two extremely similar images could not possibly be mapped by their respective networks to very different locations in feature space because each network computes the same function.

Different hand crafted features have been proposed for offline signature verification tasks. Many of them take into account the global signature image for feature extraction, such as, block codes, wavelet and Fourier series etc [5]. Some other methods consider the geometrical and topological characteristics of local attributes, such as position, tangent direction, blob structure, connected component and curvature [3]. Projection and contour based methods [6] are also quite popular for offline signature verification. Apart from the above mentioned methods, approaches fabricated on direction profile [6, 7], surroundedness features [8], grid based methods [9], methods based on geometrical moments [10], and texture based features [11] have also become famous in signature verification task. Few structural methods that consider the relations among local features are also explored for the same task. Examples include graph matching [12] and recently proposed compact correlated features [13]. On the other hand, Siamese like networks are very popular for different verification tasks, such as, online signature verification [14], face verification [15, 16] etc. Furthermore, it has also been used for one-shot image recognition [17], as well as for sketch-based image retrieval task [18]. Nevertheless, to the best of our knowledge, till date, convolutional Siamese network has never been used to model an offline signature verifier, which provides our main motivation.

The main contribution of this paper is the proposal of a convolutional Siamese network, named *SigNet*, for offline signature verification problem. This, in contrast to other methods based on hand crafted features, has the ability to model generic signature forgery techniques and many other related properties that envelops minute inconsistency in signatures from the training data. In contrary to other one-shot image verification tasks, the problem with signature is far more complex because of subtle variations in writing styles independent of scripts, which could also encapsulate some degrees of forgery. Here we mine this ultra fine anamorphosis and create a generic model using *SigNet*.

The rest of the paper is organized as follows: In Section 2 we describe the SigNet and its architechture. Section 3 presents our experimental validation and compares the proposed method with available state-of-the-art algorithms. Finally, in Section 4, we conclude the paper with a defined future direction.

## 2. SigNet: Siamese Network for Signature Verification

In this section, at first, the preprocessing performed on signature images is explained in Section 2.1. This is followed by a detailed description of the proposed Siamese architecture in Section 2.2.

### *2.1. Preprocessing*

Since batch training a neural network typically needs images of same sizes but the signature images we consider have different sizes ranges from 153 × 258 to 819 × 1137. We resize all the images to a fixed size 155×220 using bilinear interpolation. Afterwards, we invert the images so that the background pixels have 0 values. Furthermore, we normalize each image by dividing the pixel values with the standard deviation of the pixel values of the images in a dataset.

### *2.2. CNN and Siamese Network*

Deep Convolutuional Neural Networks (CNN) are multilayer neural networks consists of several convolutional layers with different kernel sizes interleaved by pooling layers, which summarizes and downsamples the output of its convolutions before feeding to next layers. To get nonlinearity rectified linear units are also used. In this work, we used different convolutional kernels with sizes starting with 11 × 11 to 3 × 3. Generally a differentiable loss function is chosen so that Gradient descent can be applied and the network weights can be optimized. Given a differentiable loss function, the weights of different layers are updated using back propagation. As the optimization can not be applied to all training data where training size is large batch optimizations gives a fair alternative to optimize the network.

Siamese neural network is a class of network architectures that usually contains two identical subnetworks. The twin CNNs have the same configuration with the same parameters and shared weights. The parameter updating is mirrored across both the subnetworks. This framework has been successfully used for dimensionality reduction in weakly supervised metric learning [15] and for face verification in [16]. These subnetworks are joined by a loss function at the top, which computes a similarity metric involving the Euclidean distance between the feature representation on each side of the Siamese network. One such loss function that is mostly used in Siamese network is the *contrastive loss* [15] defined as follows:

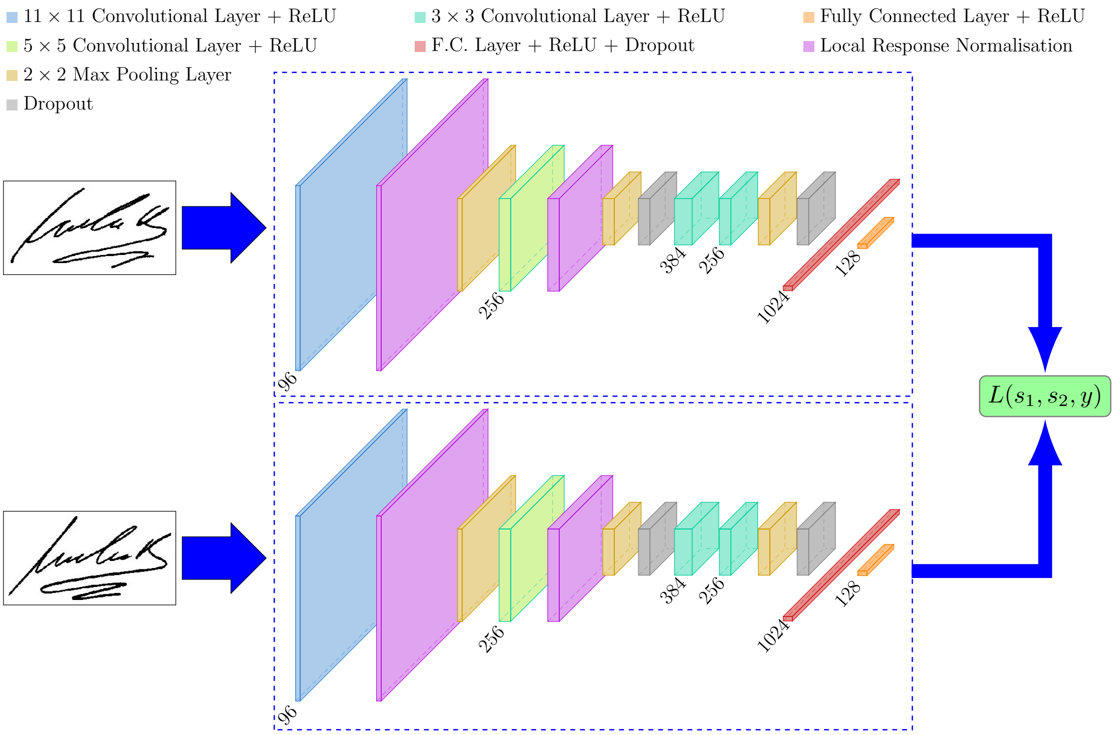
11 × 11 Convolutional Layer + ReLU
5 × 5 Convolutional Layer + ReLU
2 × 2 Max Pooling Layer
Dropout
3 × 3 Convolutional Layer + ReLU
F.C. Layer + ReLU + Dropout
Fully Connected Layer + ReLU
Local Response Normalisation
96
256
384
256
1024
128
96
256
384
256
1024
128
$L(s_1, s_2, y)$

**Fig. 1. Architecture of SigNet: the input layer, *i.e.* the 11×11 convolution layer with ReLU, is shown in blue, whereas all the 3×3 and 5×5 convolution layers are depicted in cyan and green respectively. All the local response normalization layers are shown in magenta, all the max pooling layers are depicted in brick color and the dropout layers are exhibited in gray. The last orange block represents the high level feature output from the constituting CNNs, which are joined by the loss function in Eqn. 1. (Best viewed in pdf)**

$$L(s_1, s_2, y) = \alpha(1-y)D_w^2 + \beta y \max(0, m - D_w)^2 \qquad (1)$$

where $s_1$ and $s_2$ are two samples (here signature images), $y$ is a binary indicator function denoting whether the two samples belong to the same class or not, $\alpha$ and $\beta$ are two constants and $m$ is the margin equal to 1 in our case. $D_w = \|f(s_1; w_1) - f(s_2; w_2)\|_2$ is the Euclidean distance computed in the embedded feature space, $f$ is an embedding function that maps a signature image to real vector space through CNN, and $w_1$, $w_2$ are the learned weights for a particular layer of the underlying network. Unlike conventional approaches that assign binary similarity labels to pairs, Siamese network aims to bring the output feature vectors closer for input pairs that are labelled as similar, and push the feature vectors away if the input pairs are dissimilar. Each of the branches of the Siamese network can be seen as a function that embeds the input image into a space. Due to the loss function selected (Eqn. 1), this space will have the property that images of the same class (genuine signature for a given writer) will be closer to each other than images of different classes (forgeries or signatures of different writers). Both branches are joined together by a layer that computes the Euclidean distance between the two points in the embedded space. Then, in order to decide if two images belong to the similar class (genuine, genuine) or a dissimilar class (genuine, forged) one needs to determine a threshold value on the distance.

### 2.3. Architecture

We have used a CNN architecture that is inspired by Krizhevsky *et al.* [19] for an image recognition problem. For the easy reproducibility of our results, we present a full list of parameters used to design the CNN layers in Table 1. For convolution and pooling layers, we list the size of the filters as $N \times H \times W$, where $N$ is the number of filters, $H$ is the height and $W$ is the width of the corresponding filter. Here, *stride* signifies the distance between the application of filters for the convolution and pooling operations, and *pad* indicates the width of added borders to the input. Here it is to be mentioned that padding is necessary in order to convolve the filter from the very first pixel in the input image. Throughout the network, we use Rectified Linear Units (ReLU) as the activation function to the output of all the convolutional and fully connected layers. For generalizing the learned features, Local Response Normalization is applied according to [19], with the parameters shown in the corresponding row in Table 1. With the last two pooling layers and the first fully connected layer, we use a Dropout with a rate equal to 0.3 and 0.5, respectively.

The first convolutional layers filter the 155 × 220 input signature image with 96 kernels of size 11 × 11 with a stride of 1 pixels. The second convolutional layer takes as input the (response-normalized and pooled) output of the first convolutional layer and filters it with 256 kernels of size 5×5. The third and fourth convolutional layers are connected to one another without any intervention of pooling or normalization of layers. The third layer has 384 kernels of size 3 × 3 connected to the

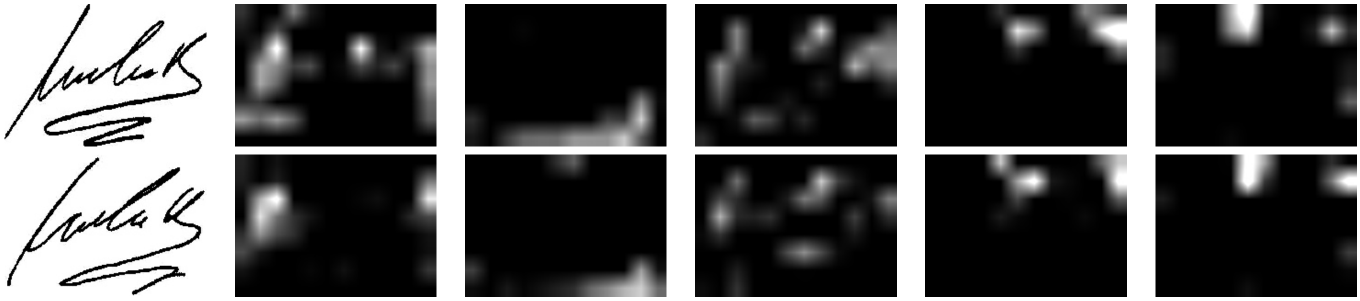

**Fig. 2. A pair of genuine (top left) and forged (bottom left) signatures, and corresponding response maps with five different filters that have produced higher energy activations in the last convolution layer of SigNet.**

**Table 1. Overview of the constituting CNNs**

| Layer | Size | Parameters |
|---|---|---|
| Convolution | $96 \times 11 \times 11$ | stride = 1 |
| Local Response Norm. | - | $\alpha = 10^{-4}, \beta = 0.75$<br>$k = 2, n = 5$ |
| Pooling | $96 \times 3 \times 3$ | stride = 2 |
| Convolution | $256 \times 5 \times 5$ | stride = 1, pad = 2 |
| Local Response Norm. | - | $\alpha = 10^{-4}, \beta = 0.75$<br>$k = 2, n = 5$ |
| Pooling + Dropout | $256 \times 3 \times 3$ | stride = 2, $p = 0.3$ |
| Convolution | $384 \times 3 \times 3$ | stride = 1, pad = 1 |
| Convolution | $256 \times 3 \times 3$ | stride = 1, pad = 1 |
| Pooling + Dropout | $256 \times 3 \times 3$ | stride = 2, $p = 0.3$ |
| Fully Connected + Dropout | 1024 | $p = 0.5$ |
| Fully Connected | 128 | |

**Table 2. Training Hyper-parameters**

| Parameter | Value |
|---|---|
| Initial Learning Rate (LR) | 1e-4 |
| Learning Rate Schedule | LR ← LR × 0.1 |
| Weight Decay | 0.0005 |
| Momentum ($\rho$) | 0.9 |
| Fuzz factor ($\epsilon$) | 1e-8 |
| Batch Size | 128 |

(normalized, pooled, and dropout) output of the second convolutional layer. The fourth convolutional layer has 256 kernels of size 3×3. This leads to the neural network learning fewer lower level features for smaller receptive fields and more features for higher level or more abstract features. The first fully connected layer has 1024 neurons, whereas the second fully connected layer has 128 neurons. This indicates that the highest learned feature vector from each side of SigNet has a dimension equal to 128.

We initialize the weights of the model according to the work of Glorot and Bengio [20], and the biases equal to 0. We trained the model using RMSprop for 20 epochs, using momentum rate equal to 0.9, and mini batch size equal to 128. We started with an intial learning rate (LR) equal to $1e-4$ with hyper parameters $\rho = 0.9$ and $\epsilon = 1e-8$. All these values are shown in Table 2. Our entire framework is implemented using Keras library with the TensorFlow as backend. The training was done using a GeForce GTX 1070 and a TITAN X Pascal GPU, and it took 2 to 9 hours to run approximately, depending on different databases.

Figure 2 shows five different filter activations in the last convolutional layer on a pair of (genuine, forged) signatures, which have received comparatively higher discrimination. The first row corresponds to the genuine signature image, whereas, the second row corresponds to the forged one and these two signatures are correctly classified as dissimilar by SigNet. Each column starting from the second one shows the activations under the convolution of the same filter. The responses in the respective zones show the areas or signature features that are learned by the network for distinguishing these two signatures.

## 3. Experiments

In order to evaluate our signature verification algorithm, we have considered *four* widely used benchmark databases, *viz.*, (1) CEDAR, (2) GPDS300, (3) GPDS Synthetic Signature Database, and (4) BHSig260 signature corpus. *The source code of SigNet will be available once the paper gets accepted for publication.*

### *3.1. Datasets*

#### *3.1.1. CEDAR*

CEDAR signature database[1] contains signatures of 55 signers belonging to various cultural and professional backgrounds. Each of these signers signed 24 genuine signatures 20 minutes apart. Each of the forgers tried to emulate the signatures of 3 persons, 8 times each, to produce 24 forged signatures for each of the genuine signers. Hence the dataset comprise $55 \times 24 = 1,320$ genuine signatures as well as $1,320$ forged signatures. The signature images in this dataset are available in gray scale mode.

[1] Available at `http://www.cedar.buffalo.edu/NIJ/data/signatures.rar`

#### 3.1.2. GPDS300

GPDS300 signature corpus[2] comprises 24 genuine and 30 forged signatures for 300 persons. This sums up to $300 \times 24 = 7,200$ genuine signatures and $300 \times 30 = 9,000$ forged signatures. The 24 genuine signatures of each of the signers were collected in a single day. The genuine signatures are shown to each forger and are chosen randomly from the 24 genuine ones to be imitated. All the signatures in this database are available in binary form.

#### 3.1.3. GPDS Synthetic

GPDS synthetic signature database[3] is built based on the synthetic individuals protocol [21]. This dataset is comprised of 4000 signers, where each individual has 24 genuine and 30 forged signatures resulting in $4000 \times 24 = 96,000$ genuine and $4000 \times 30 = 120,000$ forged signatures.

#### 3.1.4. BHSig260

The BHSig260 signature dataset[4] contains the signatures of 260 persons, among them 100 were signed in Bengali and 160 are signed in Hindi [11]. The authors have followed the same protocol as in GPDS300 to generate these signatures. Here also, for each of the signers, 24 genuine and 30 forged signatures are available. This results in $100 \times 24 = 2,400$ genuine and $100 \times 30 = 3,000$ forged signatures in Bengali, and $160 \times 24 = 3,840$ genuine and $160 \times 30 = 4,800$ forged signatures in Hindi. Even though this dataset is available together, we experimented with our method separately on the Bengali and Hindi dataset.

### 3.2. Performance Evaluation

A threshold $d$ is used on the distance measure $D(x_i, x_j)$ output by the SigNet to decide whether the signature pair $(i, j)$ belongs to the *similar* or *dissimilar* class. We denote the signature pairs $(i, j)$ with the same identity as $\mathcal{P}_{\text{similar}}$, whereas all pairs of different identities as $\mathcal{P}_{\text{dissimilar}}$. Then, we can define the set of all *true positives* (TP) at $d$ as

$$TP(d) = \{(i, j) \in \mathcal{P}_{\text{similar}}, \text{ with } D(x_i, x_j) \leq d\}$$

Similarly the set of all *true negatives* (TN) at $d$ can be defined as

$$TN(d) = \{(i, j) \in \mathcal{P}_{\text{dissimilar}}, \text{ with } D(x_i, x_j) > d\}$$

Then the true positive rate $TPR(d)$ and the true negative rate $TNR(d)$ for a given signature, distance $d$ are then defined as

$$TPR(d) = \frac{|TP(d)|}{|\mathcal{P}_{\text{similar}}|}, \; TNR(d) = \frac{|TN(d)|}{|\mathcal{P}_{\text{dissimilar}}|}$$

where $\mathcal{P}_{\text{similar}}$ is the number of similar signature pairs. The final accuracy is computed as

$$\text{Accuracy} = \max_{d \in D} \frac{1}{2}(TPR(d) + TNR(d)) \qquad (2)$$

which is the maximum accuracy obtained by varying $d \in D$ from the minimum distance value to the maximum distance value of $D$ with step equal to 0.01.

### 3.3. Experimental Protocol

Since our method is designed for writer independent signature verification, we divide each of the datasets as follows. We randomly select $M$ signers from the $K$ (where $K > M$) available signers of each of the datasets. We keep all the original and forged signatures of these $M$ signers for training and the rest of the $K - M$ signers for testing. Since all the above mentioned datasets contain 24 genuine signatures for each of the authors, there are only ${}^{24}C_2 = 276$ (genuine, genuine) signature pairs available for each author. Similarly, since most of the datasets contain 30 (for CEDAR 24) forged signatures for each signer, there are only $24 \times 30 = 720$ (for CEDAR $24 \times 24 = 576$) (genuine, forged) signature pairs can be obtained for each author. For balancing the similar and dissimilar classes, we randomly choose only 276 (genuine, forged) signature pairs from each of the writers. This protocol results in $M \times 276$ (genuine, genuine) as well as (genuine, forged) signature pairs for training and $(K - M) \times 276$ for testing. Table 3 shows the values of $K$ and $M$ for different datasets, that are considered for our experiments.

**Table 3. $K$ and $M$ values of different datasets**

| Datasets | $K$ | $M$ |
|---|---|---|
| CEDAR | 55 | 50 |
| GPDS300 | 300 | 150 |
| GPDS Synthetic | 4000 | 3200 |
| Bengali | 100 | 50 |
| Hindi | 160 | 100 |

Although most of the existing datasets contain forged signatures, in real life scenarios, there can be cases where getting training samples from forgers might be difficult. Thus, a system trained with genuine-forged signature pairs will be inadequate to deal with such set up. One way to deal with this type of situations is to use only the genuine signatures of other signer as forged signatures (called as *unskilled* forged signatures). To be applicable in such scenarios, we have performed an experiment only on the GPDS-300 dataset, where the genuine signatures of other writers are used as unskilled forged signatures. However, during testing, we have used genuine-forged pairs of the same signers, *i.e.*, we tested our system for it's ability to distinguish between genuine and forged signatures of the same person.

### 3.4. Results and Discussions

Table 4 shows the accuracies of our proposed SigNet together with other state-of-the-art methods on different datasets discussed in Section 3.1. It is to be noted that SigNet outperformed the state-of-the-art methods on three datasets, *viz.* GPDS Synthetic, Bengali, and CEDAR dataset. A possible reason for the lower performance on the GPDS300 is the less number of signature samples for learning with many different signature styles. However, on GPDS Synthetic, our proposed network outperformed the same method proposed by Dutta *et al.* [13] possibly because there were plenty of training samples for learning the available signature styles. Moreover, it can be

[2] Available at `http://www.gpds.ulpgc.es/download`

[3] Available at `http://www.gpds.ulpgc.es/download`

[4] The dataset is available at `https://goo.gl/9QfByd`

**Table 4. Comparison of the proposed method with the state-of-the-art methods on various signature databases.**

| Databases | State-of-the-art Methods | #Signers | Accuracy | FAR | FRR |
|---|---|---|---|---|---|
| CEDAR Signature Database | Word Shape (GSC) (Kalera *et al.* [5]) | 55 | 78.50 | 19.50 | 22.45 |
| | Zernike moments (Chen and Srihari [22]) | 55 | 83.60 | 16.30 | 16.60 |
| | Graph matching (Chen and Srihari [12]) | 55 | 92.10 | 8.20 | 7.70 |
| | Surroundedness features (Kumar *et al.* [8]) | 55 | 91.67 | 8.33 | 8.33 |
| | Dutta *et al.* [13] | 55 | **100.00** | **0.00** | **0.00** |
| | SigNet | 55 | **100.00** | **0.00** | **0.00** |
| GPDS 300 Signature Corpus | Ferrer *et al.* [7] | 160 | 86.65 | 12.60 | 14.10 |
| | Vargas *et al.* [23] | 160 | 87.67 | 14.66 | 10.01 |
| | Solar *et al.* [24] | 160 | 84.70 | 14.20 | 16.40 |
| | Kumar *et al.* [8] | 300 | 86.24 | 13.76 | 13.76 |
| | Dutta *et al.* [13] | 300 | **88.79** | **11.21** | **11.21** |
| | SigNet | 300 | 76.83 | 23.17 | 23.17 |
| | SigNet (unskilled forged) | 300 | 65.36 | 34.64 | 34.64 |
| GPDS Synthetic Signature Corpus | Dutta *et al.* [13] | 4000 | 73.67 | 28.34 | 27.62 |
| | SigNet | 4000 | **77.76** | **22.24** | **22.24** |
| Bengali | Pal *et al.* [11] | 100 | 66.18 | 33.82 | 33.82 |
| | Dutta *et al.* [13] | 100 | 84.90 | 15.78 | 14.43 |
| | SigNet | 100 | **86.11** | **13.89** | **13.89** |
| Hindi | Pal *et al.* [11] | 100 | 75.53 | 24.47 | 24.47 |
| | Dutta *et al.* [13] | 100 | **85.90** | **13.10** | **15.09** |
| | SigNet | 100 | 84.64 | 15.36 | 15.36 |

observed that our system trained on genuine-unskilled forged pairs is outperformed by the system trained on genuine-forged examples. This is quite justified and very intuitive as identifying forgeries of a signature needs attention to minute details of one's signature, which can not be captured when unskilled forged signatures (*i.e.*genuine signatures of other signers) are used as training examples.

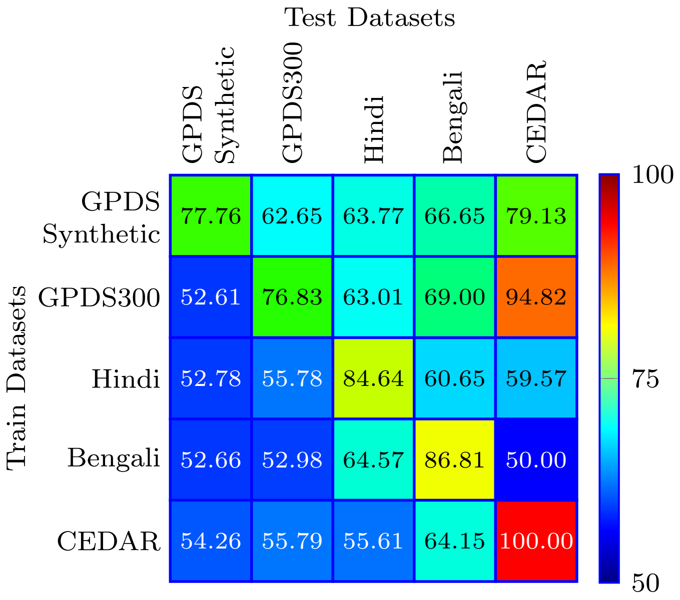

**Fig. 3. Accuracies obtained by *SigNet* with cross dataset settings.**

To get some ideas on the generalization of the proposed network and the strength of the models learned on different datasets, we performed a second experiment with cross dataset settings. To do this, at a time, we have trained a model on one of the above mentioned datasets and tested it on all the other corpus. We have repeated this same process over all the datasets. The accuracies obtained by SigNet on the cross dataset settings are shown in Figure 3, where the datasets used for training are indicated in rows and the datasets used for testing are exhibited along columns. It is to be observed that for all the datasets, the highest accuracy is obtained with a model trained on the same dataset. This implies all the datasets have some distinctive features, despite the fact that, CEDAR, GPDS300 and GPDS Synthetic datasets contain signatures with nearly same style (some handwritten letters with initials etc.). However, this fact is justifiable in case of BHSig260 dataset, because it contains signatures in Indic script and the signatures generally look like normal text containing full names of persons. Therefore, it is probable that the network models some script based features in this case. Furthermore, it is usually noted that the system trained on a comparatively bigger and diverse dataset is more robust than the others, which is the reason why better average accuracies are obtained by the model trained on GPDS Synthetic and GPDS300. These experiments strongly show the possibility of creating signature verification system in those cases where training is not possible due to the dearth of sufficient data. In those situation, a robust pretrained model could be used with a lightweight fine tuning on the available specific data. We also thought of the situation where the forger not knowing the real identity of the person, he or she introduces his signature or scribbling which has more variations than the skilled forged ones. To evaluate this, we used the trained model on GPDS300 (trained with skilled forgery) and tested it on signatures placed against a random forgery (*i.e.*genuine signature of another person) giving an expected increase in performance with 79.19% accuracy rate in GPDS300 dataset (keeping rest of the experimental setup same). This also proves that the model trained to find subtle differences in signature, also performs well when the variations in signatures are large.

## 4. Conclusions

In this paper, we have presented a framework based on Siamese network for offline signature verification, which uses writer independent feature learning. This method does not rely on hand-crafted features unlike its predecessors, instead it learns them from data in a writer independent scenario. Experiments conducted on GPDS Syntehtic dataset demonstrate that this is a step towards modelling a generic prototype for real forgeries based on synthetically generated data. Also, our experiments made on cross domain datasets emphasize how well our architecture models the fraudulence of different handwriting style of different signers and forgers with diverse background and scripts. Furthermore, the SigNet designed by us has surpassed the state-of-the-art results on most of the benchmark Signature datasets, which is encouraging for further research in this direction. Our future work in this line will focus on the development of more enriched network model. Furthermore, other different frameworks for verification task will also be explored.

## Acknowledgement

This work has been partially supported by the European Union's research and innovation program under the Marie Skłodowska-Curie grant agreement No. 665919. The TITAN X Pascal GPU used for this research was donated by the NVIDIA Corporation.

## References

[1] R. Plamondon, S. Srihari, Online and off-line handwriting recognition: a comprehensive survey, IEEE TPAMI 22 (1) (2000) 63–84.

[2] D. Impedovo, G. Pirlo, Automatic signature verification: The state of the art, IEEE TSMC 38 (5) (2008) 609–635.

[3] M. E. Munich, P. Perona, Visual identification by signature tracking, IEEE TPAMI 25 (2) (2003) 200–217.

[4] D. Bertolini, L. Oliveira, E. Justino, R. Sabourin, Reducing forgeries in writer-independent off-line signature verification through ensemble of classifiers, PR 43 (1) (2010) 387–396.

[5] M. K. Kalera, S. N. Srihari, A. Xu, Offline signature verification and identification using distance statistics, IJPRAI 18 (7) (2004) 1339–1360.

[6] G. Dimauro, S. Impedovo, G. Pirlo, A. Salzo, A multi-expert signature verification system for bankcheck processing, IJPRAI 11 (05) (1997) 827–844.

[7] M. A. Ferrer, J. B. Alonso, C. M. Travieso, Offline geometric parameters for automatic signature verification using fixed-point arithmetic, IEEE TPAMI 27 (6) (2005) 993–997.

[8] R. Kumar, J. Sharma, B. Chanda, Writer-independent off-line signature verification using surroundedness feature, PRL 33 (3) (2012) 301–308.

[9] K. Huang, H. Yan, Off-line signature verification based on geometric feature extraction and neural network classification, PR 30 (1) (1997) 9–17.

[10] V. Ramesh, M. N. Murty, Off-line signature verification using genetically optimized weighted features, PR 32 (2) (1999) 217–233.

[11] 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: DAS, 2016, pp. 72–77.

[12] S. Chen, S. Srihari, A new off-line signature verification method based on graph, in: ICPR, 2006, pp. 869–872.

[13] A. Dutta, U. Pal, J. Lladós, Compact correlated features for writer independent signature verification, in: ICPR, 2016, pp. 3411–3416.

[14] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, R. Shah, Signature verification using a "siamese" time delay neural network, in: NIPS, 1994, pp. 737–744.

[15] S. Chopra, R. Hadsell, Y. LeCun, Learning a similarity metric discriminatively, with application to face verification, in: CVPR, 2005, pp. 539–546.

[16] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: A unified embedding for face recognition and clustering, in: CVPR, 2015, pp. 815–823.

[17] G. Koch, R. Zemel, R. Salakhutdinov, Siamese neural networks for one-shot image recognition, in: ICML, 2015, pp. 1–8.

[18] Y. Qi, Y. Z. Song, H. Zhang, J. Liu, Sketch-based image retrieval via siamese convolutional neural network, in: ICIP, 2016, pp. 2460–2464.

[19] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: NIPS, 2012, pp. 1097–1105.

[20] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: AISTATS, 2010, pp. 249–256.

[21] M. A. Ferrer, M. Diaz-Cabrera, A. Morales, Synthetic off-line signature image generation, in: ICB, 2013, pp. 1–7.

[22] S. Chen, S. Srihari, Use of exterior contours and shape features in off-line signature verification, in: ICDAR, 2005, pp. 1280–1284.

[23] F. Vargas, M. Ferrer, C. Travieso, J. Alonso, Off-line handwritten signature gpds-960 corpus, in: ICDAR, Vol. 2, 2007, pp. 764–768.

[24] J. Ruiz del Solar, C. Devia, P. Loncomilla, F. Concha, Offline signature verification using local interest points and descriptors, in: CIARP, 2008, pp. 22–29.