Pattern Recognition Letters
journal homepage: www.elsevier.com

Combining Graph Edit Distance and Triplet Networks for Offline Signature Verification

Paul Maergner[a,**], Vinaychandran Pondenkandath[a], Michele Alberti[a], Marcus Liwicki[b], Kaspar Riesen[c], Rolf Ingold[a], Andreas Fischer[a,d]

[a]University of Fribourg, Department of Informatics, DIVA Group, 1700 Fribourg, Switzerland
[b]Luleå University of Technology, EISLAB Machine Learning, 971 87 Luleå, Sweden
[c]University of Applied Sciences and Arts Northwestern Switzerland, Institute for Information Systems, 4600 Olten, Switzerland
[d]University of Applied Sciences and Arts Western Switzerland, Institute of Complex Systems, 1700 Fribourg, Switzerland

ABSTRACT

Offline signature verification is a challenging pattern recognition task where a writer model is inferred using only a small number of genuine signatures. A combination of complementary writer models can make it more difficult for an attacker to deceive the verification system. In this work, we propose to combine a recent structural approach based on graph edit distance with a statistical approach based on deep triplet networks. The combination of the structural and statistical models achieve significant improvements in performance on four publicly available benchmark datasets, highlighting their complementary perspectives.

1. Introduction

Handwritten signatures remain a widely used and accepted mean of biometric authentication even in the modern world. Hence, there is an interest in verifying the genuineness of signatures. To this day, automatic signature verification remains an active field of research (Diaz et al., 2019; Hafemann et al., 2017b) and the levels of accuracy achieved by state-of-the-art systems is similar to that of other biometric verification systems (Impedovo and Pirlo, 2008). The pattern recognition community is distinguishing between two cases of signature verification: the *offline* case, where only static images of the signatures are available, and the *online* case, where dynamic information like the velocity and pressure is additionally available.

The majority of current state-of-the-art approaches to offline signature verification use statistical pattern recognition, i.e. fixed-size feature vectors are used to represent signatures. In the past, these vector representations have been generated using handcrafted feature extractors, which leverage either *local information*, e.g. histogram of oriented gradients, local binary patterns, or Gaussian grid features taken from signature contours (Yilmaz et al., 2011), or *global information*, such as number of branches in the skeleton, number of holes, geometrical features like Fourier descriptors, position of barycenter, moments, projections, distributions, tortuosities, directions, curvatures and chain codes (Impedovo and Pirlo, 2008; Plamondon and Lorette, 1989). In recent years, however, with the rise of deep learning, the state of the art shifts toward learning features directly from fixed-size signature images using neural networks (Diaz et al., 2019; Hafemann et al., 2017b). These networks rely on convolutional neural network architectures (CNN) of various kinds (Hafemann et al., 2017a; Rantzsch et al., 2016; Zhang et al., 2016).

Structural pattern recognition using graphs for pattern representation offers another way of approaching signature verification. Graphs provide a powerful representation formalism that can be beneficial for signature verification. For example, graphs could use nodes to represent local information and edges to model the nodes' relation in the global structure.

The problem of graph dissimilarity computation is often solved via error-tolerant *graph matching* algorithm (Conte et al., 2004; Foggia et al., 2014). One approach is to find a mapping that minimizes a given cost function. However, the problem of optimizing this cost is known to be NP-complete (Zeng et al., 2009). This means that the run time may be intractable even for rather small graphs, which may be one of the main reasons why graphs have rarely been used for signature verification in the past.

In recent years, however, several *approximate*, or *suboptimal*,

[**]Corresponding author: Tel.: +41 26 300 92 92;
e-mail: paul.maergner@unifr.ch (Paul Maergner)

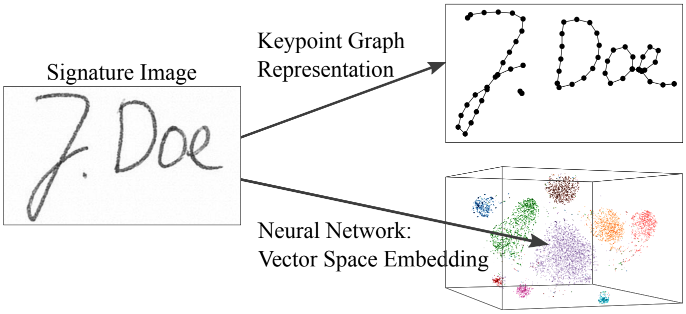
Signature Image
Keypoint Graph Representation
Neural Network: Vector Space Embedding

Fig. 1: Proposed structural and statistical signature image representations

algorithms for graph matching have been proposed (Zeng et al., 2009; Boeres et al., 2004; Justice and Hero, 2006; Riesen and Bunke, 2009). These algorithms offer polynomial, rather than exponential, run times. Yet, they do not guarantee to find the global minimum of the matching cost, but only a local one.

Two alternative families of error-tolerant graph matching that differ in their basis from more traditional approaches, are *graph embeddings* and *graph kernels*. An important class of graph embedding are *Spectral methods* (Xiao et al., 2009, 2010). However, there are many other graph embedding methods, for example, methods based on entropy computations (Han et al., 2015). Graph kernels provide an implicit graph embedding. An important group are *Random walk kernels* that measure the similarity of two graphs by the number of random walks in both graphs that have all or some labels in common (Bai et al., 2017).

Some early works using graphs for signature verification are representations based on stroke primitives (Sabourin et al., 1994), a modular graph matching approach (Bansal et al., 2009), and basic concepts of graph theory (Fotak et al., 2011). More recently, a general signature verification framework based on the graph edit distance between labeled graphs has been introduced by Maergner et al. (2017). In that work, the computational complexity of graph-based pattern analysis is reduced by employing the bipartite approximation framework proposed by Riesen and Bunke (2009). This approach has been combined with a complementary structural approach called *inkball models* in Maergner et al. (2018a).

In the present paper, we present an extension of the work published in Maergner et al. (2018b). The original work investigated whether structural and statistical signature models have complimentary strength and hence work well together in a multiple classifier system (see Fig. 1). This has been done by combining an approach based on graph edit distance with a convolutional neural network using the triplet loss function (Hoffer and Ailon, 2015). We aim at further investigating this combination in this journal extension. The focus hereby is the practical application of the combined structural and statistical approach on a challenging real-world problem and a more comprehensive evaluation. Furthermore, we investigate possible improvements in neural network architecture and training.

Compared to the original publication, we utilized a more powerful network architecture, named DenseNet-121 (Huang et al., 2017), to compare against the previously used architecture, ResNet-18 (He et al., 2016). The reason behind this choice is the particular nature of the DenseNet architecture, which allows features from lower layers to be propagated directly to the higher layers of the network. This is known to work well in natural images and in this work, we aim to find out whether this generalizes to the signature verification domain. Furthermore, we investigate an additional pretraining step for the neural networks in which we train for classification before training for similarity. This pretraining step has been shown to increase the network performances especially when for each class, there is little amount of labeled data available (Pondenkandath et al., 2018). Finally, we are using two more test sets and more evaluation metrics to compare our framework against more published results in our experimental evaluation. That is, our evaluation is now performed on four publicly available datasets.

This paper is structured as follows. The graph-based approach is reviewed in Section 2 and the neural-network-based approach is described in Section 3. Eventually, the signature verification system using both approaches is detailed in Section 4. Finally, we present and discuss our experimental results in Section 5 and deduce our conclusions in Section 6.

2. Structural Graph-Based Approach

Our structural approach for signature verification has been proposed in Maergner et al. (2017) and is based on graph edit distance. That is, the dissimilarity of two signatures is measured by comparing two keypoint graphs that are created from the corresponding signature images. In order to compute the graph edit distance, a suboptimal algorithm (Riesen and Bunke, 2009) is actually employed. The individual steps, viz. the graph extraction and graph comparison, are briefly described in the following two subsections. A more detailed description can be found in Maergner et al. (2017).

2.1. Keypoint Graphs

Formally, a labeled graph is defined as a four-tuple $g = (V, E, \mu, \nu)$, where V is the finite set of nodes, $E \subseteq V \times V$ is the set of edges, $\mu : V \rightarrow L_V$ is the node labeling function, and $\nu : E \rightarrow L_E$ is the edge labeling function.

Keypoint graphs are labeled graphs created from points extracted from an image of handwriting. Specifically, the nodes represent points on the skeleton of the handwriting and they are labeled with their coordinates. These points include end- and junction-points of the skeleton as well as points sampled between these keypoints in equidistant intervals of length D. The nodes that are actually connected on the skeleton are linked with unlabeled and undirected edges. The node labels are finally normalized by subtracting the average of all node labels of the respective graph so that the new average node label is $(0, 0)$. Fig. 1 contains an example of a keypoint graph.

2.2. Graph Edit Distance

Graph edit distance (GED) is one of the most flexible graph matching approaches since it can compare any kind of labeled graph given an appropriate cost function. GED determines the lowest-cost edit path that transforms graph $g_1 = (V_1, E_1, \mu_1, \nu_1)$ into graph $g_2 = (V_2, E_2, \mu_2, \nu_2)$. Hereby, an edit path is a sequence of edit operations. Generally, these edit operations are substitutions, deletions, and insertions of nodes and edges. A cost function assigns a cost to each of these edit operations. The major disadvantage is that GED is part of the group of NP-hard problems and its computational complexity is exponential in the number of nodes in the two graphs, $O(|V_1|^{|V_2|})$.

To lower the computational complexity, we leverage the bipartite approximation framework proposed by Riesen and Bunke (2009). This approximation reformulates the computation of GED to an instance of a *linear sum assignment problem* with cubic complexity, $O((V_1 + V_2)^3)$. We are using the lower bound of the GED as introduced in Riesen et al. (2014).

For our particular signature verification task, the following cost function is used. The node substitution cost is set to the Euclidean distance between the respective node labels. The node deletion and insertion cost is set to a constant value C_{node}. The edge substitution cost is zero and the edge deletion and insertion cost is fixed to a constant value C_{edge}.

Lastly, the graph-based dissimilarity score d_{GED} is obtained by dividing the actual GED by the maximum GED, which is the cost of deleting all nodes and edge in the first graph and inserting all nodes and edges of the second graph. This ensures that the dissimilarity score d_{GED} lies in the interval [0, 1] regardless the size of the graphs. Formally, we define the graph-based dissimilarity of two signature images as:

$$d_{\text{GED}}(r, t) = \frac{\text{GED}(g_r, g_t)}{\text{GED}_{\max}(g_r, g_t)}, \tag{1}$$

where r and t are two signature images, g_r and g_t are the corresponding keypoint graphs, $\text{GED}(g_r, g_t)$ is the lower bound of the GED between g_r and g_t, and $\text{GED}_{\max}(g_r, g_t)$ is the maximum GED between g_r and g_t.

3. Statistical Neural Network-Based Approach

In the last decade, convolutional neural networks (CNN) have become state of the art in a large variety of applications, especially in computer-vision tasks. In fact, already back in 2012 CNNs have been proven to be suited to work with images (Krizhevsky et al., 2012). Over the years there have been many developments and improvements and nowadays, CNNs represent the backbone of most vision-based applications. The idea of our neural network-based approach is to train a deep CNN to embed images of signatures into a high-dimensional vector space where the distance of two signatures reflect their similarity, i.e. two maps of signatures of the same user should be close together and signature maps from different users should be in different areas of the vector space. This is achieved by employing a triplet-based learning method. In recent years, several image matching problems were tackled using this approach with promising success (Balntas et al., 2016; Hoffer and Ailon, 2015; Zagoruyko and Komodakis, 2015). As we do have images of signatures, we can formulate the signature verification task as an image matching problem and proceed to train our network with the triplet-based method. A visualization of a vector space created by triplet training can be seen in Fig. 1, where points of the same class are clustered together.

3.1. Network Architecture

We are considering two state-of-the-art CNN architectures:

- *ResNet-18* proposed by He et al. (2016), which is the 18 layer deep variant of a CNN that uses skip connections between layers to tackle the vanishing gradient problem.
- *DenseNet-121* introduced by Huang et al. (2017), which is an 121 layer deep variant of a CNN that contains four so-called dense blocks. In these dense blocks, each layer is connected with each following layer through skip connections.

These networks have been chosen for their wide-spread successful applications in different computer-vision problems i.e, they both set new milestones in the ImageNet challenge (Russakovsky et al., 2015) and influenced the research and development of the most recent architectures.

3.2. Transfer Learning

It is common knowledge in the computer vision community that using transfer learning reduces training time and improves performance, especially when limited training data is available (He et al., 2018; Tan et al., 2018). In our scenario, we do transfer learning from ImageNet (Jia Deng et al., 2009) which is a well-known large-scale dataset of natural images. In practice, we use the weights provided by PyTorch[1] and use them as initial values for the subsequent classification pretraining (see Section 3.3).

3.3. Classification Pretraining

In this work we are considering the classification pretraining introduced by Pondenkandath et al. (2018). In this *classification pretraining* step, the neural network is trained for classification with cross-entropy loss on the same training set before training for similarity. Thus, we first train the network to distinguish (and classify) specific users and only after train it for user-agnostic similarity with triplet learning. The premise is that this classification pretraining procedure is beneficial especially in cases where there is limited classification ground truth available. In Table 5 we present the numerical measurements of its effects and which are then discussed in Section 5.4.2.

3.4. Triplet-Based Learning

We follow a triplet-based learning method (Hoffer and Ailon, 2015). A triplet is a tuple of three signatures $\{a, p, n\}$ where a is the anchor (a signature), p is the positive sample (another signature from the same user) and n is the negative sample (a signature from a different user). Given a large number of these triplets, the neural network is trained to minimize the following loss function:

$$L(\delta_+, \delta_-) = max(\delta_+ - \delta_- + \mu, 0), \tag{2}$$

where δ_+ and δ_- are the Euclidean distances between anchor-positive and anchor-negative pairs in the feature space and μ is the margin used.

[1] `https://github.com/pytorch/vision/tree/master/torchvision/models`

3.5. Signature Image Matching

The neural network is represented as a function f that embeds the image of a signature into a latent space as previously described. We define the dissimilarity of two signature images r and t as the Euclidean distance of their embedding vectors. Formally,

$$d_{\text{neural}}(r, t) = \|f(r) - f(t)\|_2. \tag{3}$$

4. Combined Signature Verification System

A signature verification system calculates a dissimilary score between *reference* signatures of the claimed user and an unseen signature. If this dissimilarity score (see Eq. 6 or 7) is below a certain threshold the signature is accepted as genuine, otherwise, the signature is rejected as a forgery.

4.1. User-based Normalization

Each user has an individual intra-user variability in their signatures. We counter this by normalizing each dissimilarity score using the average dissimilarity score between the reference signatures of the current user as suggested in Maergner et al. (2017). Formally,

$$\hat{d}(r, t) = \frac{d(r, t)}{\delta(R)}, \tag{4}$$

where t is a questioned signature image, $r \in R$ is a reference signature image, R is the set of all reference signature images of the current user, and

$$\delta(R) = \frac{1}{|R|} \sum_{r \in R} \min_{s \in R \setminus r} d(r, s). \tag{5}$$

4.2. Signature Verification Score

The final signature verification score is the minimum dissimilarity over all reference signatures R of the claimed user to the questioned signature t. Formally,

$$d(R, t) = \min_{r \in R} \hat{d}(r, t) \tag{6}$$

4.3. Multiple Classifier System

In order to combine the graph-based dissimilarity and the neural network based dissimilarity, we define a multiple classifier system (MCS) as the sum of the two dissimilarities. Before the combination is carried out, each dissimilarity score is normalized using a Z-score computed on all reference signature images in the dataset. Formally, we define

$$d_{\text{MCS}}(R, t) = \min_{r \in R} \left(\hat{d}^*_{\text{GED}}(r, t) + \hat{d}^*_{\text{neural}}(r, t) \right), \tag{7}$$

where $\hat{d}^*$ refers to the Z-score normalized dissimilarity scores.

5. Experimental Evaluation

In this section, the experimental evaluation of our signature verification system is introduced. We describe the used datasets, the employed evaluation metrics, the training process, and finally, the results on four publicly available datasets are compared against the state of the art.

5.1. Datasets

We evaluate the performance of our two methods individually and combined on four publicly available datasets.

GPDSsynthetic-Offline is a large synthetic dataset (Ferrer et al., 2015) that replaces the popular GPDS-960 dataset and earlier variants, which are no longer available (see GPDS website (Ferrer, 2016)). The dataset contains 4,000 synthetic users with 24 genuine signatures and 30 simulated forgeries each. Different modeled pens have been used to generate the signatures. The simulated resolution of the images is 600 dpi. We employ five subsets of this dataset:

- **GPDS-last10:** last 10 user (3991 to 4000).
- **GPDS-last100:** last 100 user (3901 to 4000).
- **GPDS-last1000:** last 1000 user (3001 to 4000).
- **GPDS-last3925:** last 3925 user (76 to 4000).
- **GPDS-75:** first 75 user (1 to 75).

GPDS-last10, GPDS-last100, GPDS-last1000, and GPDS-last3925 are used for training and tuning, while GPDS-75 is exclusively used for testing and comparing against the state of the art.

MCYT-75 is part of the MCYT baseline corpus (Ortega-Garcia et al., 2003; Fierrez-Aguilar et al., 2004). It contains 75 users with 15 genuine signatures and 15 forgeries each. The signatures have been scanned at 600 dpi. This dataset is used exclusively for testing and comparing against state of the art.

The **UTSig** dataset is a relatively new Persian signature dataset (Soleimani et al., 2016c). It contains 115 users and 27 genuine signatures, 3 opposite-hand signatures[2], and 42 forgeries for each user. The users signed in 6 different bounding boxes to simulate different conditions. The signatures have been scanned at 600 dpi. This dataset is used exclusively for testing and comparing against state of the art.

The **CEDAR** dataset contains 55 users (Kalera et al., 2004). For each user, it contains 24 genuine signatures and 24 forgeries. The signatures have been scanned at 300 dpi. This dataset is used exclusively for testing and comparing against state of the art.

5.2. Types of Forgeries

For each user, a set of genuine signatures is used as references. Our results are labeled with Rx where x stands for the number of references actually used for verification. We always use the first x genuine signatures as references for reproducibility. Our classifiers use these references to distinguish between genuine signatures and forgeries. We are considering two types of forgeries, which are common in the pattern recognition community, *skilled forgeries* (SF) and so-called *random forgeries*[3](RF). SF are provided together with each benchmark

[2] The opposite-hand signatures should be treated as forgeries according to the authors of the dataset.

[3] This term is mainly used in the pattern recognition community and it might be confusing for readers from other fields. For more details, see Malik and Liwicki (2012).

dataset (see Section 5.1). While the details vary, the common theme is that SF are created with some information about the user's signature. On the contrary, RF are genuine signatures of other users that are used to simulate a brute force attack. We are using one genuine signature from each other user as random forgeries.

5.3. Evaluation Metrics

The main performance measure for our verification systems is the equal error rate (EER), which is the error rate at the decision threshold when the false rejection rate (FRR) is equal to the false acceptance rate (FAR). FRR is the percentage of genuine signatures rejected and FAR is the percentage of forgeries accepted. If calculated on their own, FRR and FAR require a threshold (see Section 5.5). We distinguish EER and FAR based on the type of forgery, viz. EER_{SF} and FAR_{SF} when dealing with skilled forgeries, and EER_{RF} and FAR_{RF} when considering random forgeries. We also calculate the average error rate (AER): $\text{AER}_{\text{SF}} = (\text{FRR} + \text{FAR}_{\text{SF}})/2$. Additionally, we consider two types of EER. The common way is to determine the EER by using the same global threshold for all users. We label this as the global EER, formally $\text{EER}_{\text{RF}}^{\text{global}}$ and $\text{EER}_{\text{SF}}^{\text{global}}$. Another way is to calculate the EER individually per user and average the result over all user. This user-specific EER is, however, ignoring the problem of user adaptation. We refer to this metric as $\text{EER}_{\text{RF}}^{\text{user}}$ and $\text{EER}_{\text{SF}}^{\text{user}}$.

5.4. Setup

In a real-world scenario, it is unlikely to have access to a large database of real signatures for training. A possible solution is the use of a synthetic signature database. We want to focus on a use-case with realistic difficulty. Therefore, we are using the synthetic GPDS dataset for training (see Section 5.1). Specifically, we train using the GPDS-last10, GPDS-last100, GPDS-last1000, and GPDS-last3925 datasets. These subsets are disjoint with the GPDS-75 dataset that is used for testing. We want to emphasize that with the exception of GPDS-75, the evaluation is carried out on different datasets containing real data. This is certainly a challenging approach, but it allows a better look at how the system would perform if no dataset specific training data is available.

5.4.1. Graph Parameter Validation

To determine the best parameters for our graph-based method, we performed a grid search with the following parameters: $D \in \{25, 50, 100\}$, $C_{\text{node}} \in \{12.5, 25, 50, 100\}$, and $C_{\text{edge}} \in \{0, 12.5, 25, 50, 100\}$. The best parameters have been selected by calculating $\text{EER}_{\text{SF}}^{\text{global}}$ on the GPDS-last100 dataset. The best results have been achieved using the following parameters $D = 25$, $C_{\text{node}} = 25$, and $C_{\text{edge}} = 25$. We use these parameters in the following experiments as our *proposed GED* system.

5.4.2. Neural Network Training

We have validated two different network architectures (ResNet-18 and DenseNet-121, see Section 3.1) in conjunction with two different pretraining approaches (see Section 3.3) and four different training sets (GPDS-last10, GPDS-last100, GPDS-last1000, GPDS-last3925, see Section 5.1) resulting in $2 \cdot 2 \cdot 4 = 16$ neural networks. Only the genuine signatures from these synthetic datasets are considered for training. For each user, 16 signatures are used for training and the remaining 8 signatures are used for validation. The networks are trained to distinguish between the different users of the dataset. We optimize the network using the Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.01 and a momentum of 0.9. The networks have been trained using the open-source toolkit DeepDIVA (Alberti et al., 2018).

Table 5 shows the validation results of the 16 models using the $\text{EER}_{\text{RF}}^{\text{global}}$ metric on the GPDS-last100 dataset. The classification pretraining consistently leads to better results. The best results are achieved using the DenseNet-121 architecture with classification pretraining trained on the GPDS-last1000 dataset. This network is used in the following experiments as our *proposed CNN* system. The *proposed MCS* system is the combination (see Section 4.3) of the proposed GED and proposed CNN systems.

5.5. Threshold Selection

In order to calculate the FRR, FAR_{SF}, FAR_{RF}, and AER, we need to select a decision threshold. Three thresholds are determined, one for each of our three proposed systems, namely *proposed GED*, *proposed CNN*, *proposed MCS*. For each of the systems, we use the threshold that leads to the $\text{EER}_{\text{SF}}^{\text{global}}$ on the GPDS-last100 dataset. We use the $\text{EER}_{\text{SF}}^{\text{global}}$ since the threshold for skilled forgeries tends to be closer to the genuine samples than the random forgery threshold. Using a specific proposed system, the same threshold is used for all users, test sets, and experiments (skilled and random forgeries). This is a more challenging approach than selecting a user-specific threshold based on the references in each dataset, especially since none of the users during testing have been seen during training and tuning. Please, note that the before mentioned threshold is only used for FRR, FAR, and AER. The EER is not calculated based on a fixed threshold (see Section 5.3).

5.6. Test Results

Tables 1 to 4 show the results of the three proposed systems on the four test sets (GPDS-75, MCYT-75, UTSig, and CEDAR). Overall, the proposed MCS achieves better results than the individual proposed systems. This indicates that the GED-based system and the CNN-based system have complementary properties that benefit from a combined perspective. Individually, the proposed CNN system achieves significantly better results in the random forgeries evaluation. The proposed GED system obtains better results on skilled forgeries on GPDS-75 and UTSig, similar results on MCYT-75, and worse results on CEDAR when compared with the individual CNN-based system.

The results of our proposed systems are compared with several published state-of-the-art results. Unfortunately, there are several different evaluation protocols that have been used in the past and thus a meaningful comparison is often not trivial. We

Table 1: GPDS-75 dataset: Comparison with other published methods. The best result is highlighted in bold font and the top three results are marked with numbers.

System	#Refs	FRR	$\mathrm{FAR}_{\mathrm{RF}}$	$\mathrm{EER}^{\mathrm{user}}_{\mathrm{RF}}$	$\mathrm{EER}^{\mathrm{global}}_{\mathrm{RF}}$	$\mathrm{FAR}_{\mathrm{SF}}$	$\mathrm{AER}_{\mathrm{SF}}$	$\mathrm{EER}^{\mathrm{user}}_{\mathrm{SF}}$	$\mathrm{EER}^{\mathrm{global}}_{\mathrm{SF}}$
GPDS website (Ferrer, 2016)	10	-	-	-	0.76	-	-	-	16.01
Soleimani et al. (2016a)	10	6.51*	0.11*(3)	-	1.08	18.23*	12.37*	-	12.83
Maergner et al. (2018a)	10	-	-	-	2.05	-	-	-	**6.84** (1)
Narwade et al. (2018)	12	**3.51***(1)	-	-	-	13.91*(3)	8.71*(2)	-	-
Maergner et al. (2018b)	10	-	-	-	0.56 (3)	-	-	-	7.24 (2)
Proposed GED	10	8.86	1.37	1.35 (3)	3.80	10.31 (2)	9.58 (3)	6.89 (2)	9.47
Proposed CNN	10	6.19 (3)	0.04 (2)	0.11 (2)	**0.38** (1)	15.33	10.76	8.13 (3)	10.27
Proposed MCS	10	5.05 (2)	**0.00** (1)	**0.00** (1)	0.47 (2)	**9.11** (1)	**7.08** (1)	**4.76** (1)	7.29 (3)

*: The starred numbers have been calculated for 2500 users (Soleimani et al., 2016a) and 90 users (Narwade et al., 2018).
However, results for 75 users should be similar since this dataset is quite stable for different user counts (Ferrer, 2016).

Table 2: MCYT-75 dataset: Comparison with other published methods. The best result is highlighted in bold font and the top three results are marked with numbers.

System	#Refs	FRR	$\mathrm{FAR}_{\mathrm{RF}}$	$\mathrm{EER}^{\mathrm{user}}_{\mathrm{RF}}$	$\mathrm{EER}^{\mathrm{global}}_{\mathrm{RF}}$	$\mathrm{FAR}_{\mathrm{SF}}$	$\mathrm{AER}_{\mathrm{SF}}$	$\mathrm{EER}^{\mathrm{user}}_{\mathrm{SF}}$	$\mathrm{EER}^{\mathrm{global}}_{\mathrm{SF}}$
Fierrez-Aguilar et al. (2004)	10	-	-	1.14	-	-	-	9.28	-
Alonso-Fernandez et al. (2007)	10	-	-	7.26	-	-	-	22.13	-
Gilperez et al. (2008)	10	-	-	1.18	-	-	-	6.44 (3)	-
Vargas et al. (2011)	10	12.61	1.53	-	2.20	**7.53** (1)	10.07 (2)	-	8.80 (3)
Ooi et al. (2016)	10	-	-	-	-	-	-	-	9.87
Soleimani et al. (2016a)	10	6.13 (2)	**0.00** (1)	-	0.37 (2)	12.71 (2)	**9.42** (1)	-	9.86
Hafemann et al. (2018)	10	-	-	0.03 (2)	**0.19** (1)	-	-	-	**3.64** (1)
Maergner et al. (2018a)	10	-	-	0.52	1.24	-	-	5.78 (2)	8.71 (2)
Narwade et al. (2018)	10	-	-	-	-	-	-	-	9.26
Maergner et al. (2018b)	10	-	-	0.25	0.79 (3)	-	-	10.13	11.11
Proposed GED	10	6.13 (2)	1.17	0.70	2.67	20.80	13.47	7.02	13.24
Proposed CNN	10	8.00 (3)	0.14 (3)	0.09 (3)	1.03	16.80 (3)	12.40	6.84	12.71
Proposed MCS	10	**4.00** (1)	0.13 (2)	**0.00** (1)	0.79 (3)	17.24	10.62 (3)	**3.91** (1)	9.16

Table 3: UTSig dataset: Comparison with other published methods. The best result is highlighted in bold font and the top three results are marked with numbers.

System	#Refs	FRR	$\mathrm{FAR}_{\mathrm{RF}}$	$\mathrm{EER}^{\mathrm{user}}_{\mathrm{RF}}$	$\mathrm{EER}^{\mathrm{global}}_{\mathrm{RF}}$	$\mathrm{FAR}_{\mathrm{SF}}$	$\mathrm{AER}_{\mathrm{SF}}$	$\mathrm{EER}^{\mathrm{user}}_{\mathrm{SF}}$	$\mathrm{EER}^{\mathrm{global}}_{\mathrm{SF}}$
Soleimani et al. (2016c)	12	39.27	0.08 (3)	-	-	21.29 (3)	30.28	-	29.71
Soleimani et al. (2016a)	12	18.96	**0.00** (1)	-	-	16.15 (2)	17.56 (3)	-	17.45 (3)
Soleimani et al. (2016b)	12	16.34	0.01 (2)	-	-	**15.69** (1)	**16.02** (1)	-	**16.00** (1)
Narwade et al. (2018)	9	**7.41** (1)	-	-	-	24.95	16.18 (2)	-	-
Proposed GED	12	14.67	1.25	2.04 (2)	4.90 (3)	21.60	18.14	14.78 (2)	18.96
Proposed CNN	12	14.26 (3)	1.37	2.70 (3)	4.52 (2)	32.71	23.49	20.77 (3)	23.57
Proposed MCS	12	7.88 (2)	1.12	**0.82** (1)	**3.06** (1)	29.49	18.69	**14.09** (1)	17.35 (2)

Table 4: CEDAR dataset: Comparison with other published methods. The best result is highlighted in bold font and the top three results are marked with numbers.

System	#Refs	FRR	$\mathrm{FAR}_{\mathrm{RF}}$	$\mathrm{EER}^{\mathrm{user}}_{\mathrm{RF}}$	$\mathrm{EER}^{\mathrm{global}}_{\mathrm{RF}}$	$\mathrm{FAR}_{\mathrm{SF}}$	$\mathrm{AER}_{\mathrm{SF}}$	$\mathrm{EER}^{\mathrm{user}}_{\mathrm{SF}}$	$\mathrm{EER}^{\mathrm{global}}_{\mathrm{SF}}$
Chen and Srihari (2006)	16	**7.70** (1)	-	-	-	8.20 (2)	**7.95** (1)	-	-
Bharathi and Shekar (2013)	12	9.36 (2)	-	-	-	**7.84** (1)	8.60 (2)	-	-
Hafemann et al. (2018)	10	-	-	0.37 (2)	**1.14** (1)	-	-	-	**3.60** (1)
Proposed GED	10	19.09	0.24 (2)	1.52	5.05	15.76	17.42	11.52 (3)	17.50
Proposed CNN	10	16.10	0.40 (3)	1.31 (3)	2.96 (3)	14.92	15.51	8.56 (2)	15.30 (3)
Proposed MCS	10	12.21 (3)	**0.13** (1)	**0.30** (1)	1.82 (2)	12.35 (3)	12.28 (3)	**5.91** (1)	12.27 (2)

Table 5: Comparison of the different CNNs using $\mathrm{EER}_{\mathrm{RF}}^{\mathrm{global}}$ on GPDS-last100. The best result printed in bold font.

Architecture	Classification Pretraining	Training data: GPDS-			
		last10	last100	last1000	last3925
ResNet-18	No	7.64	0.28	0.21	0.19
ResNet-18	Yes	6.71	0.12	0.14	0.14
DenseNet-121	No	6.64	0.21	0.21	0.21
DenseNet-121	Yes	6.07	0.14	**0.10**	0.14

follow an evaluation protocol that has been used by other recent publications and compare against publications that follow the same (or almost identical) protocol.

On GPDS-75 (Table 1), the proposed MCS system achieves the best result with respect to five of the eight evaluation metrics and all results are within the top-3. This test set is the most similar to the training dataset. These results highlight that the proposed approach can achieve excellent results using specific training data.

On MCYT-75 (Table 2), the proposed MCS system obtains top-3 results with respect to six of the eight evaluation metrics (according to three metrics it achieves the best result). The difference between our results and the best-reported results in some of the metrics is quite large, which might be due to the lack of dataset-specific training. Regarding that our method is trained on synthetic data only, it performs overall very well on this specific signature dataset.

On UTSig (Table 3), the proposed MCS system gets top-2 results with respect to five of the eight evaluation metrics. Regarding the threshold based metrics (FRR, FAR, AER), our systems perform not that good. This indicates that the threshold that has been determined using synthetic Western signatures might not be suited for this dataset containing Persian signatures. However, the EER results are good, suggesting that the approach itself is suited for this dataset.

On CEDAR (Table 4), the proposed MCS system achieves top-3 results with respect to all eight evaluation metrics. However, significantly different evaluation protocols have been used on this dataset in the past. This limits the number of published results that can be used for comparison. Overall, the results of our proposed systems seem to be worse compared to the state of the art. A possible reason might be the lower resolution of the signature images (see Section 5.1). However, the user-specific metrics ($\mathrm{EER}_{\mathrm{RF}}^{\mathrm{user}}$ and $\mathrm{EER}_{\mathrm{SF}}^{\mathrm{user}}$) suggest that the proposed method has potential on this dataset as well.

In summary, the results show nicely that structural and statistical models perform well together. Trained on a synthetic dataset, the approaches achieve remarkable results on four different datasets without any further adaptations. When looking at the differences between the global EER versus user-specific EER, it becomes clear that the optimal decision threshold differs significantly between users. While a certain difference is expected, the difference in our results is quite significant on most datasets. This indicates that our user-based normalization is not sufficient. Improving the alignment between different users with a better user-adaptation could improve the proposed approach. Other approaches train support vector machines (SVM) for each user based on the user's reference signatures, e.g. Hafemann et al. (2018). Following a similar approach in the future could lead to further improvements in our proposed methods.

6. Conclusions and Outlook

The performance of a signature verification system on four benchmark datasets is significantly improved when combining structural and statistical models. Individually, the structural model based on graph edit distance performs overall better on skilled forgeries, while the statistical model based on deep triplet networks performs significantly better on random forgeries. These complementary strengths have been combined in our proposed multiple classifier system. Overall, the system generalized well to new data and users that have not been used for any model training or parameter tuning.

Our experiments also show that the proposed system is likely to benefit from an improved user-adaptation. Furthermore, the structural approach could be improved by investigating additional graph-representations. The statistical method might be able to learn a more accurate and general vector space embedding when employing synthetic data augmentation. Finally, the robustness of biometric authentication is likely to further improve when using a large multiple classifier system that combines even more structural and statistical classifiers.

Acknowledgment

This work has been supported by the Swiss National Science Foundation project 200021_162852.

References

Alberti, M., Pondenkandath, V., Würsch, M., Ingold, R., Liwicki, M., 2018. DeepDIVA: A Highly-Functional Python Framework for Reproducible Experiments, in: International Conference on Frontiers in Handwriting Recognition (ICFHR), Niagara Falls, USA. pp. 423–428.

Alonso-Fernandez, F., Fairhurst, M., Fierrez, J., Ortega-Garcia, J., 2007. Automatic measures for predicting performance in off-line signature, in: Proc. of International Conference on Image Processing, pp. 369–372.

Bai, L., Rossi, L., Cui, L., Zhang, Z., Ren, P., Bai, X., Hancock, E., 2017. Quantum kernels for unattributed graphs using discrete-time quantum walks. Pattern Recognition Letters 87, 96–103.

Balntas, V., Riba, E., Ponsa, D., Mikolajczyk, K., 2016. Learning local feature descriptors with triplets and shallow convolutional neural networks, in: Richard C. Wilson, E.R.H., Smith, W.A.P. (Eds.), Proc. of the British Machine Vision Conference (BMVC), BMVA Press. pp. 119.1–119.11.

Bansal, A., Gupta, B., Khandelwal, G., Chakraverty, S., 2009. Offline signature verification using critical region matching. International Journal of Signal Processing, Image Processing and Pattern 2, 57–70.

Bharathi, R.K., Shekar, B.H., 2013. Off-line signature verification based on chain code histogram and support vector machine, in: International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 2063–2068.

Boeres, M., Ribeiro, C., Bloch, I., 2004. A randomized heuristic for scene recognition by graph matching, in: Ribeiro, C., Martins, S. (Eds.), Proc. 3rd Workshop on Efficient and Experimental Algorithms, pp. 100–113.

Chen, S., Srihari, S., 2006. A New Off-line Signature Verification Method based on Graph. Proc. of International Conference on Pattern Recognition (ICPR) , 869–872.

Conte, D., Foggia, P., Sansone, C., Vento, M., 2004. Thirty years of graph matching in pattern recognition. Int. Journal of Pattern Recognition and Art. Intelligence 18, 265–298.

Diaz, M., Ferrer, M.A., Impedovo, D., Malik, M.I., Pirlo, G., Plamondon, R., 2019. A perspective analysis of handwritten signature technology. ACM Comput. Surv. 51, 117:1–117:39.

Ferrer, M.A., 2016. GPDS database website. URL: `http://www.gpds.ulpgc.es/downloadnew/download.htm`. accessed on Jan 28, 2019.

Ferrer, M.A., Diaz-Cabrera, M., Morales, A., 2015. Static Signature Synthesis: A Neuromotor Inspired Approach for Biometrics. IEEE Transactions on Pattern Analysis and Machine Intelligence 37, 667–680.

Fierrez-Aguilar, J., Alonso-Hermira, N., Moreno-Marquez, G., Ortega-Garcia, J., 2004. An off-line signature verification system based on fusion of local and global information, in: Biometric Authentication. Springer, pp. 295–306.

Foggia, P., Percannella, G., Vento, M., 2014. Graph matching and learning in pattern recognition in the last 10 years. Int. Journal of Pattern Recognition and Art. Intelligence 28.

Fotak, T., Baca, M., Koruga, P., 2011. Handwritten signature identification using basic concepts of graph theory. WSEAS Transactions on Signal Processing 7, 145–157.

Gilperez, A., Alonso-Fernandez, F., Pecharroman, S., Fierrez, J., Ortega-Garcia, J., 2008. Off-line signature verification using contour features, in: Proc. of International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 1–6.

Hafemann, L.G., Oliveira, L.S., Sabourin, R., 2018. Fixed-sized representation learning from offline handwritten signatures of different sizes. International Journal on Document Analysis and Recognition (IJDAR) 21, 219–232.

Hafemann, L.G., Sabourin, R., Oliveira, L.S., 2017a. Learning features for offline handwritten signature verification using deep convolutional neural networks. Pattern Recognition 70, 163–176.

Hafemann, L.G., Sabourin, R., Oliveira, L.S., 2017b. Offline handwritten signature verification - literature review, in: Proc. of International Conference on Image Processing Theory, Tools and Applications (IPTA), pp. 1–8.

Han, L., Wilson, R.C., Hancock, E.R., 2015. Generative graph prototypes from information theory. IEEE transactions on pattern analysis and machine intelligence 37, 2013–2027.

He, K., Girshick, R.B., Dollár, P., 2018. Rethinking imagenet pre-training. CoRR abs/1811.08883. URL: `http://arxiv.org/abs/1811.08883`.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: Proc. of Conference on Computer Vision and Pattern Recognition, pp. 770–778.

Hoffer, E., Ailon, N., 2015. Deep metric learning using triplet network, in: International Workshop on Similarity-Based Pattern Recognition, Springer. pp. 84–92.

Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2261–2269.

Impedovo, D., Pirlo, G., 2008. Automatic signature verification: The state of the art. IEEE Trans. on Systems, Man and Cybernetics Part C: Applications and Reviews 38, 609–635.

Jia Deng, Wei Dong, Socher, R., Li-Jia Li, Kai Li, Li Fei-Fei, 2009. ImageNet: A large-scale hierarchical image database, in: Proc. of IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255.

Justice, D., Hero, A., 2006. A binary linear programming formulation of the graph edit distance. IEEE Trans. on Pattern Analysis and Machine Intelligence 28, 1200–1214.

Kalera, M.K., Srihari, S., Xu, A., 2004. Offline signature verification and identification using distance statistics. International Journal of Pattern Recognition and Artificial Intelligence 18, 1339–1360.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, pp. 1097–1105.

Maergner, P., Howe, N., Riesen, K., Ingold, R., Fischer, A., 2018a. Offline signature verification via structural methods: Graph edit distance and inkball models, in: Proc. of International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 163–168.

Maergner, P., Pondenkandath, V., Alberti, M., Liwicki, M., Riesen, K., Ingold, R., Fischer, A., 2018b. Offline signature verification by combining graph edit distance and triplet networks, in: Structural, Syntactic, and Statistical Pattern Recognition, Springer International Publishing. pp. 470–480.

Maergner, P., Riesen, K., Ingold, R., Fischer, A., 2017. A structural approach to offline signature verification using graph edit distance, in: Proc. of International Conference on Document Analysis and Recognition (ICDAR), IEEE. pp. 1216–1222.

Malik, M.I., Liwicki, M., 2012. From Terminology to Evaluation: Performance Assessment of Automatic Signature Verification Systems. Proc. of International Conference on Frontiers in Handwriting Recognition , 613–618.

Narwade, P.N., Sawant, R.R., Bonde, S.V., 2018. Offline handwritten signature verification using cylindrical shape context. 3D Research 9, 48.

Ooi, S.Y., Teoh, A.B.J., Pang, Y.H., Hiew, B.Y., 2016. Image-based handwritten signature verification using hybrid methods of discrete Radon transform, principal component analysis and probabilistic neural network. Applied Soft Computing 40, 274–282.

Ortega-Garcia, J., Fierrez-Aguilar, J., Simon, D., Gonzalez, J., Faundez-Zanuy, M., Espinosa, V., Satue, A., Hernaez, I., Igarza, J.J., Vivaracho, C., Escudero, D., Moro, Q.I., 2003. MCYT baseline corpus: a bimodal biometric database. IEEE Proceedings-Vision, Image and Signal Processing 150, 395–401.

Plamondon, R., Lorette, G., 1989. Automatic signature verification and writer identification - the state of the art. Pattern Recognition 22, 107–131.

Pondenkandath, V., Alberti, M., Eichenberger, N., Ingold, R., Liwicki, M., 2018. Identifying cross-depicted historical motifs, in: International Conference on Frontiers in Handwriting Recognition (ICFHR), Niagara Falls, USA. pp. 333–338.

Rantzsch, H., Yang, H., Meinel, C., 2016. Signature embedding: Writer independent offline signature verification with deep metric learning, in: Advances in Visual Computing, Springer. pp. 616–625.

Riesen, K., Bunke, H., 2009. Approximate graph edit distance computation by means of bipartite graph matching. Image and Vision Computing 27, 950–959.

Riesen, K., Fischer, A., Bunke, H., 2014. Computing upper and lower bounds of graph edit distance in cubic time. International Workshop on Artificial Neural Networks in Pattern Recognition 8774, 129–140.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L., 2015. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) 115, 211–252.

Sabourin, R., Plamondon, R., Beaumier, L., 1994. Structural interpretation of handwritten signature images. International Journal of Pattern Recognition and Artificial Intelligence 8, 709–748.

Soleimani, A., Araabi, B.N., Fouladi, K., 2016a. Deep multitask metric learning for offline signature verification. Pattern Recognition Letters 80, 84–90.

Soleimani, A., Fouladi, K., Araabi, B.N., 2016b. Persian offline signature verification based on curvature and gradient histograms, in: International Conference on Computer and Knowledge Engineering (ICCKE), pp. 147–152.

Soleimani, A., Fouladi, K., Araabi, B.N., 2016c. Utsig: A persian offline signature dataset. IET Biometrics 6, 1–8.

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., Liu, C., 2018. A survey on deep transfer learning: 27th international conference on artificial neural networks, rhodes, greece, october 47, 2018, proceedings, part iii , 270–279doi:`10.1007/978-3-030-01424-7_27`.

Vargas, J.F., Ferrer, M.A., Travieso, C.M., Alonso, J.B., 2011. Off-line signature verification based on grey level information using texture features. Pattern Recognition 44, 375–385.

Xiao, B., Hancock, E.R., Wilson, R.C., 2009. Graph characteristics from the heat kernel trace. Pattern Recognition 42, 2589–2606.

Xiao, B., Hancock, E.R., Wilson, R.C., 2010. Geometric characterization and clustering of graphs using heat kernel embeddings. Image and Vision Computing 28, 1003–1021.

Yilmaz, M.B., Yanikoglu, B., Tirkaz, C., Kholmatov, A., 2011. Offline signature verification using classifier combination of HOG and LBP features, in: Proc. International Joint Conference on Biometrics, pp. 1–7.

Zagoruyko, S., Komodakis, N., 2015. Learning to compare image patches via convolutional neural networks, in: Proc. of IEEE Conference on Computer Vision and Pattern Recognition, pp. 4353–4361.

Zeng, Z., Tung, A., Wang, J., Feng, J., Zhou, L., 2009. Comparing stars: On approximating graph edit distance. Proc. of the VLDB Endowment 2, 25–36.

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