Offline Signature Verification by Combining Graph Edit Distance and Triplet Networks

Paul Maergner[1], Vinaychandran Pondenkandath[1], Michele Alberti[1], Marcus Liwicki[1], Kaspar Riesen[2], Rolf Ingold[1], and Andreas Fischer[1,3]

[1] University of Fribourg, DIVA group, 1700 Fribourg, Switzerland, Email: `{firstname}.{lastname}@unifr.ch`
[2] University of Applied Sciences and Arts Northwestern Switzerland, Institute for Information Systems, 4600 Olten, Switzerland, Email: `kaspar.riesen@fhnw.ch`
[3] University of Applied Sciences and Arts Western Switzerland, Institute of Complex Systems, 1700 Fribourg, Switzerland, Email: `andreas.fischer@hefr.ch`

Abstract. Biometric authentication by means of handwritten signatures is a challenging pattern recognition task, which aims to infer a writer model from only a handful of genuine signatures. In order to make it more difficult for a forger to attack the verification system, a promising strategy is to combine different writer models. In this work, we propose to complement a recent structural approach to offline signature verification based on graph edit distance with a statistical approach based on metric learning with deep neural networks. On the MCYT and GPDS benchmark datasets, we demonstrate that combining the structural and statistical models leads to significant improvements in performance, profiting from their complementary properties.

Keywords: Offline signature verification · Graph edit distance · Metric learning · Deep convolutional neural network · Triplet network

1 Introduction

To this day, handwritten signatures have remained a widely used and accepted means of biometric authentication. Automatic signature verification is an active field of research, accordingly, and the current state of the art achieves levels of accuracy similar to that of other biometric verification systems [12, 15]. Usually, two cases of signature verification are differentiated: the *offline* case, where only a static image of the signature is available, and the *online* case, where additional dynamic information like the velocity is available. Due to the lack of this information, offline signature verification applies to more use cases, but it is also considered the more challenging task.

Most state-of-the-art approaches to offline signature verification rely on statistical pattern recognition, i.e. signatures are represented using fixed-size feature vectors. These vector representations are often generated using handcrafted feature extractors leveraging either *local information*, such as local binary patterns, histogram of oriented gradients, or Gaussian grid features taken from signature contours [23], or *global information*, e.g. geometrical features like Fourier descriptors, number of branches in the skeleton, number of holes, moments, projections, distributions, position of barycenter, tortuosities, directions, curvatures

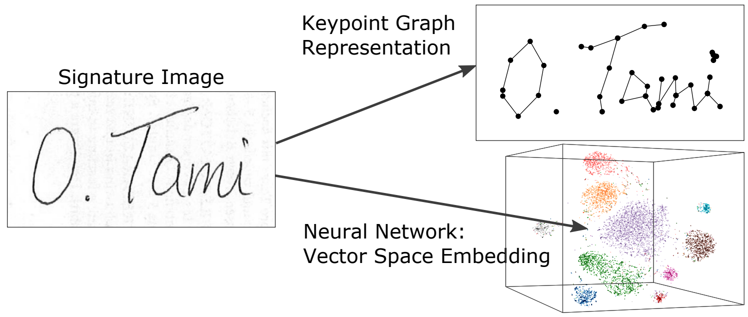


Fig. 1: Proposed structural and statistical signature image representations

and chain codes [15, 19]. More recently, with the advent of deep learning, we observe a shift away from handcrafted features towards learning features directly from the images using deep convolutional neural networks (CNN) [11].

Another way of approaching signature verification is by using graphs and structural pattern recognition. Graphs offer a more powerful representation formalism that can be beneficial for signature verification. For example, by capturing local information in nodes and their relations in the global structure using edges. But the representational power of graphs comes at the price of high computational complexity. This is probably why graphs have only been used rather rarely for signature verification in the past. Examples include the work of Sabourin et al. [22] (signatures represented based on stroke primitives), Bansal et al. [4] (modular graph matching approach), and Fotak et al. [9] (basic concepts of graph theory). More recently, a structural approach for signature verification has been introduced by Maergner et al. [16]. They propose a general signature verification framework based on the graph edit distance between labeled graphs. They employ a bipartite approximation framework [20] to reduce the computational complexity and report promising verification results using so-called keypoint graphs.

In this paper, we argue that structural and statistical signature models are quite different, with complementary strengths, and thus well-suited for multiple classifier systems. As illustrated in Fig. 1, we propose to combine the graph-based approach of Maergner et al. [16] with a statistical model inspired by recent advances in the field of deep learning, namely metric learning by means of a deep CNN [13] with the triplet loss function [14]. Such deep triplet networks can be used to embed signature images in a vector space, where signatures of the same user have a small distance and signatures of different users have a large distance. To our knowledge, this is the first combination of a graph-based approach and a deep neural network based approach for the task of signature verification.

In the remainder, the structural approach is described in Section 2, the statistical approach in Section 3, and the proposed combined system in Section 4. Afterwards, we present our experimental results in Section 5 and draw conclusions in Section 6.

2 Structural Graph-Based Approach

The structural approach used in this paper has been proposed by Maergner et al. in [16]. Two signature images are compared by first binarizing and skeletonizing the image, then creating keypoint graphs from each skeleton image, and lastly comparing the two graphs using an approximation of the graph edit distance. In the following subsections, we briefly review these steps. For a more detailed description, see [16].

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, $\mu : V \to L_V$ is the node labeling function, and $\nu : E \to L_E$ is the edge labeling function.

Keypoint graphs are created from points extracted from the skeleton image. Specifically, the nodes in the graph stand for certain points on the skeleton and are labeled with their coordinates. These points are end- and junction-points of the skeleton as well as additional points sampled in equidistant intervals of D. Unlabeled and undirected edges connect the nodes that are connected on the skeleton. The node labels are centered so that their average is $(0, 0)$. See Fig. 1 for an example of a keypoint graph.

2.2 Graph Edit Distance

Graph edit distance (GED) offers a way to compare any kind of labeled graph given an appropriate cost function. This makes GED one of the most flexible graph matching approaches. It calculates the cost of the lowest-cost edit path that transforms graph $g_1 = (V_1, E_1, \mu_1, \nu_1)$ into graph $g_2 = (V_2, E_2, \mu_2, \nu_2)$. An edit path is a sequence of edit operations, for each of which a certain cost is defined. Commonly, substitutions, deletions, and insertions of nodes and edges are considered as edit operations. The main disadvantage of GED is its computational complexity since it is exponential in the number of nodes in the two graphs, $O(|V_1|^{|V_2|})$.

This issue can be addressed by using an approximation of GED. In this paper, the bipartite approximation framework proposed by Riesen and Bunke [20] is applied. The computation of GED is reduced to an instance of a *linear sum assignment problem* with cubic complexity, $O\big((V_1 + V_2)^3\big)$. For signature verification, the lower bound introduced in [21] is considered.

The cost function is defined in the following way. The cost of a node substitution is the Euclidean distance between the node labels. For node deletion and insertion, a constant cost C_{node} is used. For edges, the substitution cost is set to zero. The edge deletion and insertion cost is set to a constant value C_{edge}.

Finally, the graph edit distance is normalized by dividing by the maximum graph edit distance, viz. the cost of deleting all nodes and edges from the first graph and inserting all the nodes and edges of the second graph. Thus, the graph-based dissimilarity is in $[0, 1]$ and describes how large the graph edit distance is when compared with the maximum graph edit distance. Formally, the graph-based dissimilarity of two signature images is defined as follows:

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

where g_r and g_t are the keypoint graphs of the signatures images r and t respectively, $\mathrm{GED}(g_r, g_t)$ is the lower bound of the graph edit distance between g_r and g_t, and $\mathrm{GED}_{\max}(g_r, g_t)$ is the maximum graph edit distance between g_r and g_t.

3 Statistical Neural Network-Based Approach

We train a deep CNN [13] using a triplet-based learning method to embed images of signatures into a high-dimensional space where the distance of two signatures reflect their similarity, i.e. two signatures of the same user are close together and signatures from different users are far apart. An exemplary visualization of the vectors produced by such model is shown in Fig. 1, where points of the same class are grouped together in clusters. This approach has been investigated in the recent past for several image matching problems with promising success, including [3, 14, 24].

3.1 Triplet-Based Learning

A triplet is a tuple of three signatures $\{a, p, n\}$ where a is the anchor (reference signature), p is the positive sample (a signature from the same user) and n is the negative sample (a signature from another user). The neural network is then trained to minimize the loss function defined as:

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

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

3.2 Signature Image Matching

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

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

4 Combined Signature Verification System

A signature verification system has to decide whether an unseen signature image is a genuine signature of the claimed user. This decision is being made by calculating a dissimilarity score between the *reference* signature of the claimed user and the unseen signature. The signature is accepted if this dissimilarity score (see Eq. 5 or 6) is below a certain threshold, otherwise the signature is rejected.

4.1 User-based Normalization

It is expected that the users have different intra-user variability. Therefore, each dissimilarity score is normalized using the average dissimilarity score between the reference signatures of the current user as suggested in [16]. 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 users, and

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

4.2 Signature Verification Score

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

$$d(R,t) = \min_{r \in R} \hat{d}(r,t) \tag{5}$$

4.3 Multiple Classifier System

We propose a multiple classifier system (MCS) as a linear combination of the graph-based dissimilarity and the neural network based dissimilarity. Z-score normalization based on all reference signature images in the current data set is applied to each dissimilarity score before the combination. Formally, we define

$$d_{\mathrm{MCS}}(R,t) = \min_{r \in R} \left(\hat{d}^{*}_{\mathrm{GED}}(r,t) + \hat{d}^{*}_{\mathrm{neural}}(r,t) \right), \tag{6}$$

where $\hat{d}^{*}$ is the z-score normalized dissimilarity score.

5 Experimental Evaluation

We evaluate the performance on two publicly available benchmark data sets by measuring the *equal error rate* (EER). The EER is the point where the false acceptance rate and the false rejection rate are equal in the *detection error tradeoff (DET)* curve. Two kinds of forgeries are tested: *skilled forgeries* (SF), which are forgeries created with information about the user's signature, and so-called *random forgeries*[4] (RF), which are genuine signatures of other users that are used in a brute force attack.

[4] This term is mainly used in the pattern recognition community and it might be confusing for readers from other fields. For more details, see [17].

5.1 Data Sets

In our evaluation, we use the following publicly available signature data sets:

- **GPDSsynthetic-Offline:** Ferrer et al. introduced this data set in [5]. It contains 24 genuine signatures and 30 skilled forgeries for $4,000$ synthetic users. This data set replaces previous signatures databases from the GPDS group, which are not available anymore.
  We use four subsets of this data set: one containing the first 75 users, and three containing the last 10, 100, or 1000 users. These subsets are called *GPDS-75*, *GPDS-last10*, *GPDS-last100*, and *GPDS-last1000* respectively.
- **MCYT-75:** This data set is part of the MCYT baseline corpus introduced by Ortega-Garcia et al. in [7, 18]. It contains 75 users with 15 genuine signatures and 15 skilled forgeries each.

5.2 Tasks

We distinguish two tasks depending on the number of references available for each user. Five genuine signatures per user (*R5*) or ten genuine signatures per user (*R10*). In both cases, the remaining genuine signatures are used for testing in both the skilled forgery (SF) and in the random forgery (RF) evaluation. The SF evaluation is performed using all available skilled forgeries for each user. The RF evaluation is carried out using the first genuine signature of all other users in the data set as random forgeries. For example for the GPDS-75 R10 tasks, that gives us $75 \cdot 10 = 750$ reference signatures, $75 \times 14 = 1,050$ genuine signatures, $75 \times 30 = 2,250$ skilled forgeries, and $75 \times 74 = 5,550$ random forgeries.

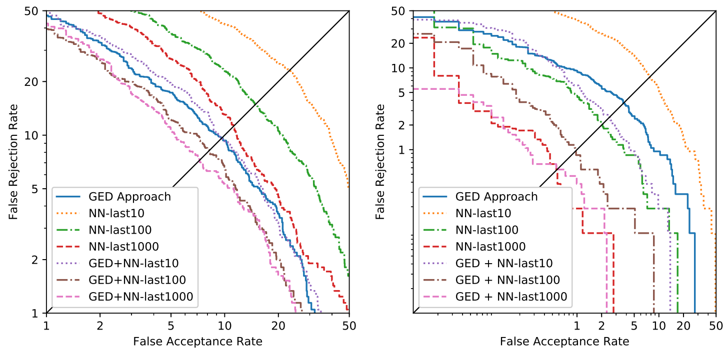
5.3 Setup

Graph Parameter Validation For the keypoint graph extraction, we use $D = 25$, which has been proposed in [16]. The cost function parameters C_{node} and C_{edge} are validated on the GPDS-last100 data set using the random forgery evaluation. No skilled forgeries are used. We perform a grid search over $C_{\text{node}} \in \{10, 15, \ldots, 60\}$ and $C_{\text{edge}} \in \{10, 15, \ldots, 60\}$. The best results have been achieved using $C_{\text{node}} = 25$ and $C_{\text{edge}} = 45$. We use these parameters in our experiments on GPDS-75 and MCYT-75.

Neural Network Training We use the ResNet18 architecture [13], which is an 18 layer deep variant of a convolutional neural network that uses shortcut connections between layers to tackle the vanishing gradient problem.

We train three different models using the DeepDIVA[5] framework [1] for the task of embedding the signature images in the vector space, where each of the models differs with respect to how much data is used for training (GPDS-last10, GPDS-last100, or GPDS-last1000). We call these systems NN-last10, NN-last100, and NN-last1000 respectively. For each person in the data set, there are

[5] https://github.com/DIVA-DIA/DeepDIVA (April 29, 2018)



(a) Skilled Forgeries (b) Random Forgeries

Fig. 2: DET curves for GPDS-75 R10

24 genuine images. We use 16 of them for training and the remaining 8 for validating the performance of the model. Skilled forgeries are not used for training.

The network is trained using the Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.01 and momentum of 0.9.

5.4 Results on MCYT-75 and GPDS-75

The EER results on GPDS-75 and MCYT-75 for both RF and SF are shown in Table 1. In all but one case, the combination of the GED approach and the neural network achieves better results than the best individual system. The neural networks trained on GPDS-last100 and GPDS-last1000 are on its own significantly better on the RF task. We can see that NN-last1000 is more specialized on the RF task on the GPDS-75 data set while losing performance on the MCYT-75 data set. Two DET curves are shown in Fig. 2.

5.5 Comparison with State-of-the-Art

Many different evaluation protocols are used for signature verification. To allow a fair comparison, we have to follow the same protocol. In the following, we present EER results using two different protocols and compare our results with other published results.

Comparison on GPDS-75 and MCYT-75 This evaluation is performed by selecting 10 reference signatures randomly[6] and average the results over 10 runs. Table 2 shows our results using the same protocol compared with the previously published results: results published in [16] and results presented on the GPDS website[7], which have been achieved using the system published in [6]. The proposed combination of the GED approach and NN-last1000 achieves the lowest EER in all tasks except for random forgeries on MCYT-75.

[6] We use the same random selections for all our results.
[7] http://www.gpds.ulpgc.es/downloadnew/download.htm (April 29, 2018)

Table 1: **EER on GPDS-75/MCYT-75**. Results on skilled forgeries (SF) and on random forgeries (RF) using the first 5 or 10 genuine as references (R5/R10).

System	**GPDS-75**				**MCYT-75**			
	RF		**SF**		**RF**		**SF**	
	R5	**R10**	**R5**	**R10**	**R5**	**R10**	**R5**	**R10**
GED approach	4.90	3.71	11.69	9.60	5.86	2.65	20.09	13.60
NN-last10	10.40	7.71	25.87	23.11	6.47	4.79	19.56	17.16
GED + NN-last10	4.00	2.47	12.04	9.51	3.19	1.59	16.53	11.29
NN-last100	3.28	2.05	17.96	14.84	3.59	1.59	20.36	12.80
GED + NN-last100	2.16	0.95	9.82	8.18	**2.79**	1.41	**15.56**	**10.40**
NN-last1000	0.68	**0.56**	13.29	11.20	3.73	1.15	19.02	13.78
GED + NN-last1000	**0.65**	**0.56**	**9.24**	**7.24**	2.92	**0.79**	17.69	11.11

Table 2: **Comparison on GPDS-75/MCYT-75**. Average EER results over 10 random selections of ten reference signatures. Evaluated on GPDS-75 and MCYT-75 for random forgeries (RF) and skilled forgeries (SF).

System	**GPDS-75 R10**		**MCYT-75 R10**	
	RF	**SF**	**RF**	**SF**
Ferrer et al. [6][7]	0.76*	16.01	**0.35***	11.54
Maergner et al. [16]	2.73	8.29	2.83	12.01
Proposed GED approach	2.75	8.31	2.67	11.42
Proposed NN-last1000	0.44	10.79	1.57	12.24
Proposed GED + NN-last1000	**0.41**	**6.49**	1.05	**9.15**

*: All genuine signatures of other users as RF

Table 3: **Comparison on MCYT-75 R5/R10**. EER results for skilled forgeries (SF) and random forgeries (RF) using an *a posteriori* user-dependent score normalization. The first 5 or 10 genuine signatures are used as references for R5 and R10 respectively.

System	**MCYT-75 R5**		**MCYT-75 R10**	
	RF	**SF**	**RF**	**SF**
Alonso-Fernandez et al. [2]	9.79*	23.78	7.26*	22.13
Fierrez-Aguilar et al. [7]	2.69**	11.00	1.14**	9.28
Gilperez et al. [10]	2.18*	**10.18**	1.18*	**6.44**
Maergner et al. [16]	2.40	14.49	1.89	11.64
Proposed GED approach	2.45	14.84	1.89	12.27
Proposed NN-last100	2.14	15.02	1.77	13.16
Proposed GED + NN-last100	**0.92**	10.67	**0.25**	10.13

*: All genuine signatures of other users as RF
**: First 5 genuine signatures from each other user as RF.

Comparison on MCYT-75 A group of publications has presented results on the MCYT-75 data set using the a posteriori user-depended score normalization introduced in [8]. By applying this normalization, all user scores are aligned so that the EER threshold is the same for all users. Table 3 shows the published results as well as our results using the same normalization. The combination of GED and NN-last100 achieves results in the middle ranks for the SF task and the overall best results for the RF task.

6 Conclusions and Outlook

Combining structural and statistical models has significantly improved the signature verification performance on the MCYT-75 and GPDSsynthetic-Offline benchmark datasets. The structural model based on approximate graph edit distance achieved better results against skilled forgeries, while the statistical model based on metric learning with deep triplet networks achieved better results against a brute-force attack with random forgeries. The proposed system was able to combine these complementary strengths and has proven to generalize well to unseen users, which have not been used for model training and hyperparameter optimization.

We can see several lines of future research. For the structural method, more graph-based representations and cost functions may be explored in the context of graph edit distance. For the statistical method, synthetic data augmentation may lead to a more accurate vector space embedding. Finally, we believe that there is a great potential in combining even more structural and statistical classifiers into one large multiple classifier system. Such a system is expected to further improve the robustness of biometric authentication.

Acknowledgment

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

References

1. Alberti, M., Pondenkandath, V., Würsch, M., Ingold, R., Liwicki, M.: DeepDIVA: A Highly-Functional Python Framework for Reproducible Experiments. Submitted at Int. Conference on Frontiers in Handwriting Recognition (2018)
2. Alonso-Fernandez, F., Fairhurst, M., Fierrez, J., Ortega-Garcia, J.: Automatic measures for predicting performance in off-line signature. In: Proc. 14th Int. Conf. on Image Processing. pp. 369–372 (2007)
3. Balntas, V., Riba, E., Ponsa, D., Mikolajczyk, K.: Learning local feature descriptors with triplets and shallow convolutional neural networks. In: Proc. of the British Machine Vision Conference (BMVC) (Sept 2016)
4. Bansal, A., Gupta, B., Khandelwal, G., Chakraverty, S.: Offline signature verification using critical region matching. Int. Journal of Signal Processing, Image Processing and Pattern Recognition **2**(1), 57–70 (2009)

5. Ferrer, M.A., Diaz-Cabrera, M., Morales, A.: Static Signature Synthesis: A Neuromotor Inspired Approach for Biometrics. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(3), 667–680 (mar 2015)
6. Ferrer, M.A., Vargas, J.F., Morales, A., Ordonez, A.: Robustness of Offline Signature Verification Based on Gray Level Features. IEEE Transactions on Information Forensics and Security **7**(3), 966–977 (jun 2012)
7. Fierrez-Aguilar, J., Alonso-Hermira, N., Moreno-Marquez, G., Ortega-Garcia, J.: An off-line signature verification system based on fusion of local and global information. In: Biometric Authentication, pp. 295–306. Springer (2004)
8. Fierrez-Aguilar, J., Ortega-Garcia, J., Gonzalez-Rodriguez, J.: Target dependent score normalization techniques and their application to signature verification. IEEE Trans. on Systems, Man, and Cybernetics, Part C **35**(3), 418–425 (2004)
9. Fotak, T., Baca, M., Koruga, P.: Handwritten signature identification using basic concepts of graph theory. WSEAS Trans. on Signal Processing **7**(4), 145–157 (2011)
10. Gilperez, A., Alonso-Fernandez, F., Pecharroman, S., Fierrez, J., Ortega-Garcia, J.: Off-line signature verification using contour features. In: Proc. 11th Int. Conf. on Front. in Handwriting Rec. pp. 1–6 (2008)
11. Hafemann, L.G., Sabourin, R., Oliveira, L.S.: Learning features for offline handwritten signature verification using deep convolutional neural networks. Pattern Recognition **70**, 163–176 (2017)
12. Hafemann, L.G., Sabourin, R., Oliveira, L.S.: Offline handwritten signature verification - literature review. In: Proc of Int. Conf. on Image Processing Theory, Tools and Applications (IPTA). pp. 1–8 (Nov 2017)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proc of Conf. on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
14. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: International Workshop on Similarity-Based Pattern Recognition. pp. 84–92. Springer (2015)
15. Impedovo, D., Pirlo, G.: Automatic signature verification: The state of the art. IEEE Trans. on Systems, Man, and Cybernetics, Part C **38**(5), 609–635 (2008)
16. Maergner, P., Riesen, K., Ingold, R., Fischer, A.: A structural approach to offline signature verification using graph edit distance. In: Proc. of International Conference on Document Analysis and Recognition. pp. 1216–1222. IEEE (2017)
17. Malik, M.I., Liwicki, M.: From Terminology to Evaluation: Performance Assessment of Automatic Signature Verification Systems. Proc. of Int. Conference on Frontiers in Handwriting Recognition pp. 613–618 (2012)
18. 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.: MCYT baseline corpus: a bimodal biometric database. IEEE Proceedings-Vision, Image and Signal Processing **150**(6), 395–401 (2003)
19. Plamondon, R., Lorette, G.: Automatic signature verification and writer identification - the state of the art. Pattern Recognition **22**(2), 107–131 (1989)
20. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. Image and Vision Computing **27**(7), 950–959 (6 2009)
21. Riesen, K., Fischer, A., Bunke, H.: Computing upper and lower bounds of graph edit distance in cubic time. Int. Workshop on Artificial Neural Networks in Pattern Recognition **8774**, 129–140 (2014)
22. Sabourin, R., Plamondon, R., Beaumier, L.: Structural interpretation of handwritten signature images. Int. Journal of Pattern Recognition and Artificial Intelligence **8**(3), 709–748 (1994)

23. Yilmaz, M.B., Yanikoglu, B., Tirkaz, C., Kholmatov, A.: Offline signature verification using classifier combination of HOG and LBP features. In: Proc. Int. Joint Conference on Biometrics. pp. 1–7 (2011)
24. Zagoruyko, S., Komodakis, N.: Learning to compare image patches via convolutional neural networks. In: Proc of IEEE Conf. on Computer Vision and Pattern Recognition. pp. 4353–4361 (2015)