Proceedings of ICDAR 2019 : 15th International Conference on Document Analysis and Recognition, 20-25 september 2019, Sydney, Australia, which should be cited to refer to this work.

Offline Signature Verification using Structural Dynamic Time Warping

Michael Stauffer*, Paul Maergner†, Andreas Fischer†‡, Rolf Ingold† and Kaspar Riesen*
*University of Applied Sciences and Arts Northwestern Switzerland,
Institute for Information Systems, Riggenbachstr. 16, 4600 Olten, Switzerland
Email: {michael.stauffer,kaspar.riesen}@fhnw.ch
†University of Fribourg,
Department of Informatics, 1700 Fribourg, Switzerland
Email: {andreas.fischer,paul.maergner,rolf.ingold}@unifr.ch
‡University of Applied Sciences and Arts Western Switzerland,
Institute of Complex Systems, 1700 Fribourg, Switzerland

***Abstract*—In recent years, different approaches for handwriting recognition that are based on graph representations have been proposed (e.g. graph-based keyword spotting or signature verification). This trend is mostly due to the availability of novel fast graph matching algorithms, as well as the inherent flexibility and expressivity of graph data structures when compared to vectorial representations. That is, graphs are able to directly adapt their size and structure to the size and complexity of the respective handwritten entities. However, the vast majority of the proposed approaches match the graphs from a global perspective only. In the present paper, we propose to match the underlying graphs from different local perspectives and combine the resulting assignments by means of Dynamic Time Warping. Moreover, we show that the proposed approach can be readily combined with global matchings. In an experimental evaluation, we employ the novel method in a signature verification scenario on two widely used benchmark datasets. On both datasets, we empirically confirm that the proposed approach outperforms state-of-the-art methods with respect to both accuracy and runtime.**

***Keywords*-Signature Verification; Dynamic Time Warping; Graph Matching; Graph Representation; Ensemble Methods;**

I. Introduction

Handwritten signatures have been traditionally used as a personal authentication measure in business and legal transactions since decades [1]–[3]. Hence, handwritten signatures are ubiquitous in our everyday life, and thus, include a large risk for possible misuse. To mitigate this risk, automatic signature verification systems can be employed. These systems basically compare a questioned signature with a set of reference signatures in order to distinguish between genuine and forged signatures [4]–[6]. However, the acquisition of reference signatures is often expensive or limited, and thus, signature verification is regarded as a challenging task (even for human experts).

In general, signature verification can be divided into *online* (also termed *dynamic*) and *offline* (also termed *static*) signature verification [1]–[3]. In the former case, signatures are acquired by means of an electronic input device, such as for example a digital pen or via input on a touch screen. As a result, dynamic temporal information of the handwriting process (e.g. acceleration, speed, and pressure) can be acquired during the signing process. In the latter case, handwritten signatures are digitised by scanning the signed artefact. Therefore, the verification task is solely based on the (x, y)-positions of the handwritten strokes, and thus, offline signature verification is commonly regarded as the more challenging task. The present paper is focusing on offline signature verification only.

The vast majority of signature verification systems make use of *statistical* (i.e. vectorial) representations [1]–[3]. That is, feature vectors or sequences of feature vectors are extracted from scanned images of handwritten signatures. In early approaches, these features are based on handwriting characteristics like outline [4], projection profiles [7], slant direction [8], [9], or the contour [5], [10]. However, also more generic features have been employed for signature representation such as for example *Local Binary Patterns (LBP)* [11], [12] and *Histogram of oriented Gradients (HoG)* [11]. More recently, features are extracted from signature images by means of *Convolutional Neural Networks (CNN)* [6] and other Deep Learning techniques [13]. Regardless the actual type of feature, different statistical classifiers and/or matching schemes for sequential data are eventually employed for signature verification like for example *Support Vector Machines (SVM)* [4], [11], [12], *Dynamic Time Warping (DTW)* [7], [10], or *Hidden Markov Models (HMM)* [4], [9].

By using graphs, the inherent topological characteristics of a handwritten signature can be directly represented in a natural and comprehensive way [14]–[18]. In [16]–[18], for instance, graphs are used to represent characteristic points of the handwriting stroke and their structural relationships. More formally, nodes are used to represent so-called *keypoints* (e.g. end- and junction-points), while edges are used to represent strokes between keypoints. Consequently, both the structure and the size of the graph can directly be adapted to the size and complexity of the underlying signature. Moreover, by means of edges one is able to represent the relationships that exist between substructures

of the handwriting. The power and flexibility of graphs is accompanied, however, with a general increase of the computational complexity of many basic procedures. The computation of a similarity or dissimilarity of graphs, for instance, is of much higher complexity than computing a vector (dis)similarity. Yet, several fast matching algorithms have been proposed in the last decade that allow to compare also larger graphs within reasonable time (e.g. [19]–[21]).

In previous graph-based signature verification approaches [16]–[18], graphs are matched from a global perspective. That is, two graphs (representing two signature instances) are matched as single entities to obtain a global matching score. In the present paper we propose to match the graphs from different local perspectives. To this end, a sliding window is used to extract several subgraphs from the underlying graphs. Eventually, the extracted subgraphs are matched by means of a recently proposed graph dissimilarity measure, i.e. the *Polar Graph Embedding distance (PGEd)* [21]. Hence, pairwise PGE distances are computed for each pair of sliding window positions. Finally, the sequences of subgraphs are optimally aligned to each other by means of DTW that operates on the matrix of PGEd's. We denote this procedure as *Structural Dynamic Time Warping (SDTW)* from now on. Clearly, SDTW can be combined with global graph matching measures in order to build an *ensemble*. In an experimental evaluation we show that the proposed method improves both accuracy and runtime when compared with previous graph-based signature verification approaches as well as other state-of-the-art signature verification systems.

The remainder of this paper is organised as follows. In Section II, we formally introduce the novel graph matching approach SDTW. The application of the proposed dissimilarity measure in a signature verification system is then described in Section III. In Section IV, we empirically compare the proposed method with state-of-the-art approaches on two widely used benchmark datasets. Finally, we draw conclusions in Section V.

II. Structural Dynamic Time Warping

In this section, we introduce a novel graph matching paradigm, named *Structural Dynamic Time Warping (SDTW)*. This paradigm makes use of two different concepts, i.e. *Polar Graph Embedding distance (PGEd)* (see Subsection II-A) and *Dynamic Time Warping (DTW)* (see Subsection II-B). In particular, two graphs are matched by means of a sliding window approach, as illustrated in Fig. 1. At each pair of window positions, the subgraphs in the respective windows are matched using the concept of PGEd. That is, we compute a dissimilarity on two local graph embeddings. Then, we shift the centre of the polar segmentations from left to right over both signatures and compare all pairs of polar segmentation. On the resulting distance matrix, the different subgraphs can then optimally be aligned along one common time axis by means of DTW. In the following two subsections, both concepts are reviewed in greater detail.

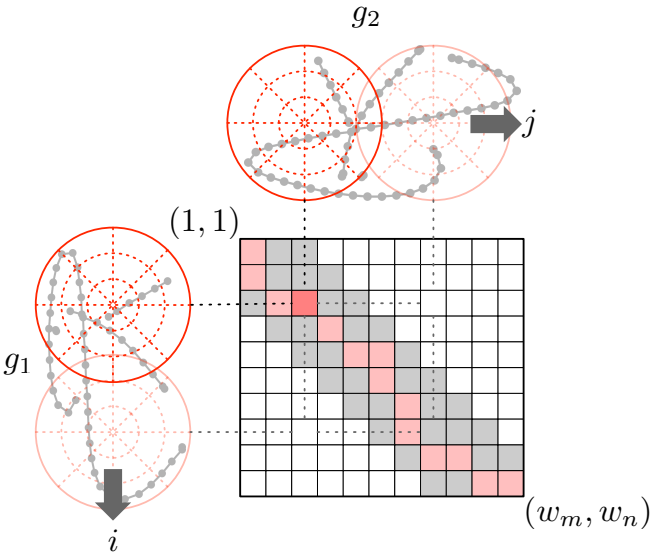

Figure 1: Matching a source graph g_1 and a target graph g_2 by means of Structural Dynamic Time Warping. For each pair of two sliding windows the subgraphs are matched with each other by means of a polar graph embedding distance. The optimal alignment of the sequences of subgraphs is finally computed via Dynamic Time Warping. The optimal alignment is shown in red color (the greyed entries indicate the Sakoe-Chiba band [22]).

A. Polar Graph Embedding Dissimilarity

In this subsection, we review and adapt a recently proposed graph dissimilarity, i.e. *Polar Graph Embedding distance (PGEd)* [21]. PGEd is based on three different processing steps. First, the graphs are segmented into a polar coordinate system. Second, histograms are derived based on the node and edge distributions of each polar graph segment. Third, the distance between the resulting histograms is computed by means of the χ^2-distance. In the following three paragraphs, we briefly describe each processing step.

1) Polar Graph Segmentation: Generally, a graph g is defined as a four-tuple $g = (V, E, \mu, \nu)$ where V and E are finite sets of nodes and edges, and $\mu : V \rightarrow L_V$ and $\nu : E \rightarrow L_E$ are labelling functions for nodes and edges, respectively. In the present paper, nodes are uniquely labelled with two-dimensional numerical labels (i.e. (x, y)-coordinates of the handwriting stroke), while edges remain unlabelled, i.e. $L_V = \mathbb{R}^2$ and $L_E = \emptyset$.

In our novel framework, we apply the polar graph embedding on subgraphs of g, i.e. the nodes (including their edges) that are actually located in a bounding circle at a

given window position. To transform such a subgraph g into a polar coordinate system, we first define a centre (x_c, y_c), where x_c refers to the window position and y_c is equal to the centre of mass of the complete graph. Next, each node label $\mu(v) = (x, y) \in \mathbb{R}^2$ is transformed to $\mu'(v) = (\rho, \theta)$ by

$$\rho = \sqrt{(x - x_c)^2 + (y - y_c)^2} \quad ,$$
$$\theta = \text{atan2}((y - y_c)/(x - x_c)) \quad ,$$

where ρ denotes to the distance from the polar centre to the node's original position, and $\theta \in [-\pi, \pi[$ refers to the angle measured from the x-axis to the node's original position.

We now define a bounding circle C with radius ρ_γ, as shown in Fig. 2a. Formally,

$$\rho_\gamma = \gamma \rho_{\max} \quad ,$$

where $\gamma \in]0, 1]$ is a user-defined parameter and $\rho_{\max}$ is the maximal distance between a polar centre and all nodes of the graph.

As shown in Fig. 2b, the bounding circle C is used to divide the graph into $n = P_r \times P_\phi$ polar segments where P_r and P_ϕ define the number of different radii and angles, respectively. Hence, every node $v \in V$ with polar coordinates (ρ, θ) can be assigned to one of the n corresponding polar segments $b_k \in \{b_1 \ldots, b_n\}$, i.e. every polar segment b_k is defined by two radii $\rho_{k_{\min}}$ and $\rho_{k_{\max}}$, and two angles $\theta_{k_{\min}}$ and $\theta_{k_{\max}}$. Formally, given the segmentation indices $i \in \{1, \ldots, P_r\}$ and $j \in \{1, \ldots, P_\phi\}$, the k-th[1] polar segment is defined by,

$$\rho_{k_{\min}} = \frac{\rho_\gamma}{P_r} \cdot (i - 1) \text{ and } \rho_{k_{\max}} = \frac{\rho_\gamma}{P_r} \cdot i \quad ,$$
$$\theta_{k_{\min}} = \frac{360^\circ}{P_\phi} \cdot (j - 1) \text{ and } \theta_{k_{\max}} = \frac{360^\circ}{P_\phi} \cdot j \quad .$$

2) Graph Quantisation: Based on the polar segments, the subgraphs are encoded in fixed sized histograms. In particular, the concept of *Histogram of oriented Gradients (HoG)* is adapted to undirected graphs to create a histogram with radial directions of the corresponding nodes and edges.

Formally, we first define a maximal number P of directions that is used to define the radial range of every bin. Hence, for every edge (u, v) in the k-th segment[2], we measure the Euclidean distance d between its adjacent nodes u and v, as well as the angle θ of the edge with respect to the x-axis. Next, distance d is assigned to the two enclosing bins b_{k_i} and b_{k_j} with respect to their radial difference to θ. Formally,

$$b_{k_i} \mathrel{+}= 1 - \frac{\theta - \theta_i}{\epsilon} d \quad \text{and} \quad b_{k_j} \mathrel{+}= \frac{\theta - \theta_i}{\epsilon} d \quad ,$$

where ϵ is the radial range per bin, i.e. $\epsilon = \frac{360^\circ}{P}$, while $\theta_i = i \cdot \epsilon$.

An example is given in Fig. 2c. In this case $P = 10$, and thus, the radial range of a bin is given by $\epsilon = \frac{360^\circ}{10} = 36^\circ$. However, the radial direction of an edge (e.g. 20°) is in most cases in in-between two bins (e.g. b_{k_1} with 0° and b_{k_2} with 36°), and thus, d is accumulated between these two bins. Note that in the current case, graphs consist of undirected edges, and thus, every edge is taken into account in both directions.

In the last step, the resulting histograms with P bins for each of the n segments are concatenated to form one global histogram $H = \{b_{1_1}, \ldots, b_{1_P}, \ldots, b_{n_1}, \ldots, b_{n_P}\}$ which is finally normalised by the $l1$-norm.

3) Graph Dissimilarity: We measure the dissimilarity of two histograms H and H' by means of the χ^2-distance. Formally,

$$PGEd(H, H') = \sum_{i=1}^{n} \sum_{j=1}^{P} \frac{(b_{i_j} - b'_{i_j})^2}{(b_{i_j} + b'_{i_j})} \qquad (1)$$

where H and H' are the histograms representing the source and target graph, respectively.

B. Dynamic Time Warping

In contrast with the method proposed in [21], where one single histogram per graph is computed, we derive sequences of histograms for each graph. That is, we move the centre (x_c, y_c), or more precisely the x_c position, from left to right. For each centre position an additional histogram is extracted from the graph (see Fig. 1). Depending on the actual position as well as the employed radius ρ_γ the resulting histograms encode node- and edge distributions of different subgraphs.

First, we derive the number of window positions w_m and w_n from the width of the source graph g_1 and target graph g_2, as follows.

$$w_m = \frac{1}{\omega} \quad ,$$
$$w_n = \frac{1}{\omega} \cdot \frac{\text{Width of } g_2}{\text{Width of } g_1} \quad ,$$

where $\omega \in]0, 1]$ is a user-defined step size.

At each window position $i \in \{1, 2, \ldots, w_m\}$ for g_1 and $j \in \{1, 2, \ldots, w_n\}$ for g_2, we derive a new polar center (x_c, y_c). Based on (x_c, y_c), we then derive two polar histograms H_i and H'_j for subgraphs g_1 and g_2, respectively. That is, for g_1 and g_2 we obtain two sequences of histograms $X = \{H_1, \ldots, H_{w_m}\}$ and $Y = \{H'_1, \ldots, H'_{w_n}\}$, respectively. The alignment cost between each histogram

[1] i.e. $k = (i - 1) \cdot P_\phi + j$

[2] Note that we also consider edges, for which only one of the adjacent nodes is part of the present polar segment.

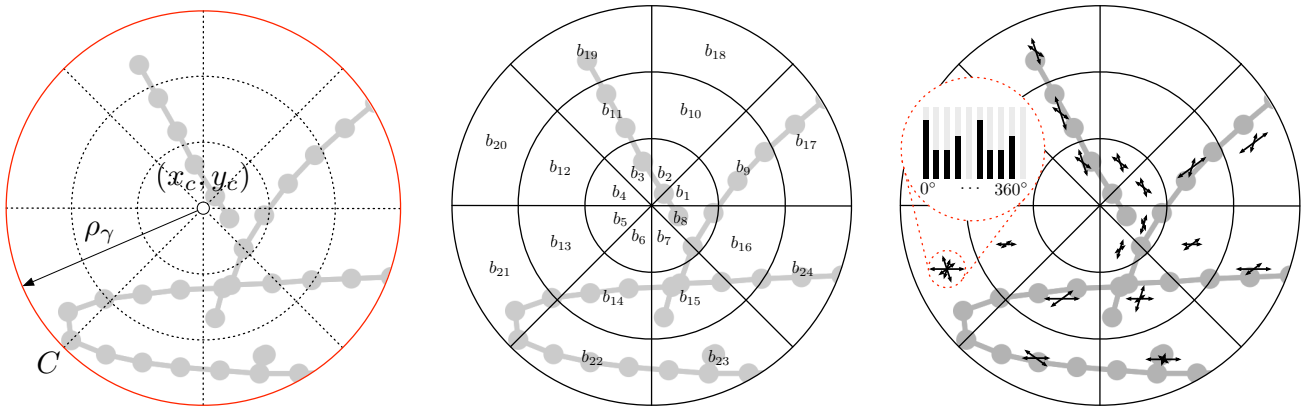

Figure 2: Polar graph embedding consisting of two processing steps: (a) + (b) Segmentation of a subgraph of graph g_1 (see Fig. 1) into $n = P_r \times P_\phi$ polar segments (e.g. $n = 24$ segments given by $P_r = 3$, $P_\phi = 8$, and $\gamma = 1.0$), (c) Construction of histograms based on radial node and edge distribution.

pair $(H_i, H'_j) \in X \times Y$ is then given by $PGEd(H_i, H'_j)$ (see Eq. 1).

Finally, the graph distance $SDTW(g_1, g_2)$ is given by the minimum alignment cost found by dynamic programming. Formally,

$$SDTW(g_1, g_2) = \sum_{k=1}^{K} PGEd(H_{i_k}, H'_{j_k}) \quad ,$$

where K is the length of the optimal warping path $((i_1, j_1), \ldots, (i_K, j_K))$ [23]. The distance $SDTW(g_1, g_2)$ can be normalised with respect to the length of the warping path K. Formally,

$$\overline{SDTW}(g_1, g_2) = \frac{SDTW(g_1, g_2)}{K} \quad .$$

To speed up this procedure we make use of a *Sakoe-Chiba band* [22] that constrains the warping path (grey highlighted in Fig. 1). Formally, given a window position i the lower bound $LB(i)$ and upper bound $UB(i)$ of the Sakoe-Chiba band is given by

$$LB(i) = \frac{n \cdot i}{m} - (\theta \cdot n) \quad ,$$
$$UB(i) = \frac{n \cdot i}{m} + (\theta \cdot n) \quad ,$$

where $\theta \in [0, 1]$ is a user-defined parameter to control the width of the Sakoe-Chiba band.

III. Signature Verification

In this section, we show how to employ SDTW (or $\overline{\text{SDTW}}$)[3] in a recently proposed graph-based signature verification framework [16]. Moreover, we propose to combine the novel SDTW-distances with global graph matching dissimilarities. Finally, we briefly review a normalisation method to reduce intrapersonal variations.

Scanned handwritten signatures are first preprocessed and skeletonised. Hence, graphs are used to represent characteristic points on the skeleton (so-called keypoints), while edges are used to represent strokes between keypoints. For further details regarding image preprocessing and graph representation we refer to [16].

The actual signature verification is then based on matching a questioned signature graph q of claimed user u with all reference signatures graphs $r \in R_u$ of user u by means of $SDTW(q, r)$, see Fig. 3. The resulting graph dissimilarities can then be used to verify whether or not the questioned signature q is genuine or forged. In particular, if the minimal graph dissimilarity to all references is below a certain threshold the unseen signature q is regarded as genuine, otherwise q is regarded as forged.

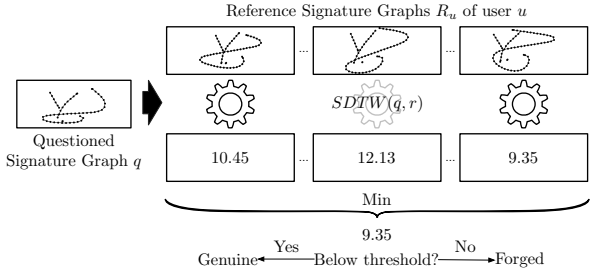

Figure 3: Graph-based signature verification process by means of Structural Dynamic Time Warping (SDTW).

Rather than matching graphs from sequences of local perspectives only, we can also consider global graph dissimilarity measures, which are eventually combined in an *ensemble*. That is, we combine $SDTW$ with global graph matching distances obtained from *Bipartite Graph Edit*

[3]For the sake of convenience we use the notation SDTW only in the remainder of our description.

Distance (BP) [24] and *Polar Graph Embedding (PGE)* [21]. Formally, we make use of the following combinations

$$\alpha\, SDTW(q,r) + \beta\, BP(q,r) \quad ,$$
$$\alpha\, SDTW(q,r) + \beta\, PGE(q,r) \quad ,$$

where $\alpha, \beta \in [0,1]$ are weighting factors that can be adjusted.

In order to reduce interpersonal variations, we employ the following normalisation based on the set of reference signatures R_u of user u [25]. In particular, we calculate a normalisation score μ_{R_u} based on the average of all minima between all reference signatures R_u of user u

$$\mu_{R_u} = \frac{\sum\limits_{s \in R_u} \min\limits_{r \in R_u \setminus \{s\}} SDTW(s,r)}{|R_u|} \quad .$$

The mean of the minimal distances μ_{R_u} can be interpreted as the dissimilarity score that can be expected for user u. Hence, we use μ_{R_u} to derive a signature score s for questioned signature q based on the normalised and minimal distance for user u. Formally,

$$s(q, R_u) = \frac{\min\limits_{r \in R_n} SDTW(q,r)}{\mu_{R_u}} \quad .$$

Note that similar normalisations can be obtained on combined distances (using both BP and PGE).

IV. Experimental Evaluation

A. Datasets

In the following experiments, we evaluate the proposed signature verification approach using SDTW on two signature benchmark datasets, i.e. *GPDSsynthetic-Offline* and *MCYT-75*. In Fig 4, one exemplary signature as well as the corresponding graph representation is given for both datasets.

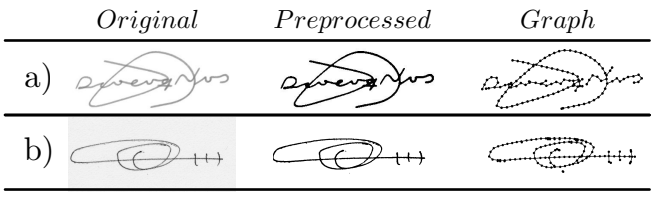

Figure 4: Exemplary offline signature (original and preprocessed) as well as the corresponding graph representation: (a) GPDSsynthetic, (b) MCYT-75.

GPDSsynthetic-Offline is a synthetic offline dataset [26], that replaces the widely used GPDS-960 dataset that is no longer publicly available[4]. The novel dataset contains 4,000 synthetic users with 24 genuine and 30 forged signatures per user. Every signature is generated by modelling different pens. MCYT-75 is part of the MCYT baseline corpus [9], [27]. This dataset contains 75 users with 15 genuine and 15 forged signatures per user. The users signed in a 127 mm $\times$ 97mm field. In Table I, a summary of the main characteristics of the two datasets is given.

Table I: The number of users, the number of genuine and forged signatures per user, and the resolution of the images.

Name	Users	Genuine	Forgeries	dpi
GPDSsynthetic	4000	24	30	600
MCYT-75	75	15	15	600

B. Experimental Setup

Generally, signature verification frameworks are evaluated using two kinds of forgeries, i.e. *Random Forgeries (RF)* and *Skilled Forgeries (SF)*. In case of RF, the forger has no prior knowledge about the signature to be forged. For our experimental evaluation, we make use of the first genuine signature from each other user to build a set of random forgeries. In case of SF, the forger has commonly both knowledge of one or several genuine signatures and time to practice the signature. As a result, skilled forgeries are often very similar when compared with the genuine template.

The proposed signature verification system is evaluated by comparing a questioned signature q with a set of reference signatures R. In general, the larger the reference set R, the better the accuracy of the verification. However, the acquisition of arbitrarily large reference sets is often not possible in practice, and thus, limited to few reference signatures per user. In our experimental evaluation, we make use of $|R| = 5$ (denoted by $R5$) and $|R| = 10$ (denoted by $R10$). If not otherwise specified, we make use of the first five or ten genuine signatures for each user as reference set R, while the remaining genuine signatures are used as positive attempts for the evaluation.

The accuracy of signature verification systems is generally measured by means of two errors, i.e. the *False Rejection Rate (FRR)* and the *False Acceptance Rate (FAR)*. In case of FRR, we measure the percentage of genuine signatures that are falsely rejected by the system, while in case of FAR, we measure the percentage of forgeries that are falsely accepted by the system. Eventually, a system is assessed by means of the *Equal Error Rate (EER)*, which refers to the error rate when the FRR is equal to the FAR.

The experimental evaluation is conducted in two steps. First, parameters are optimised on a single and independent dataset, i.e. a subset of GPDSsynthetic containing the last 100 users of the dataset (denoted by *GPDS-last100* from now on), using a skilled forgery scenario and ten reference signatures (R10). Second, optimised parameters are used to test the proposed signature verification framework on an

[4]For more details, see http://www.gpds.ulpgc.es/downloadnew/download.htm (February 25, 2018)

independent subset of GPDSsynthetic containing the first 75 users of the dataset (denoted by *GPDS-75*), as well as on *MCYT-75* using SF and RF for both R5 and R10.

C. Parameter Optimisation

For both SDTW and $\overline{\text{SDTW}}$, we first optimise different polar segmentations for PGEd (defined via $P_r = \{1, 2, \ldots, 5\}$ and $P_\phi = \{1, 2, \ldots, 10\}$), as well as different radial ranges (defined by $P = \{26, 28, \ldots, 34\}$). The remaining parameters remain fixed for this initial optimisation (we use polar radius $\gamma = 1.0$ and step size $\omega = 0.05$).

Next, we optimise the remaining parameters, i.e. we optimise the polar radius $\gamma = \{0.5, 0.6, \ldots, 1.0\}$, the step size $\omega = \{0.025, 0.050, 0.075, 0.100\}$, and the Sakoe-Chiba band $\theta = \{0.01, 0.02, \ldots, 0.15\}$, in combination with the optimised polar segmentation parameters. In Table II, the best performing parameters are shown for both SDTW and $\overline{\text{SDTW}}$.

Table II: Optimal SDTW parameter settings.

Method	P_r	P_ϕ	P	γ	ω	θ
SDTW	3	5	32	1.0	0.050	0.01
$\overline{\text{SDTW}}$	3	7	30	0.9	0.075	0.03

Finally, we optimise the weighting parameters $\alpha, \beta \in \{0.1, 0.2, \ldots, 0.9\}$ for the ensemble methods. In Table III, the best performing weights are shown for all combinations.

Table III: Optimal ensemble method parameter settings.

Ensemble	α	β
SDTW + BP	0.7	0.3
SDTW + PGE	0.5	0.1
$\overline{\text{SDTW}}$ + BP	0.5	0.4
$\overline{\text{SDTW}}$ + PGE	0.2	0.1

D. Results and Discussion

In Table IV, we compare SDTW and $\overline{\text{SDTW}}$ in a skilled forgery scenario (SF) with two global graph matching algorithms, i.e. BP and PGE. We observe that SDTW and $\overline{\text{SDTW}}$ lead to lower equal error rates in all eight cases when compared with BP. Compared with PGE, in seven out of eight cases improvements are observed with SDTW and $\overline{\text{SDTW}}$ (PGE performs slightly better than SDTW on MCYT-75 and R10). Regarding the ensemble methods, we observe that SDTW + BP is beneficial on GPDS-75, while $\overline{\text{SDTW}}$ + BP is beneficial on MCYT-75.

In Table V, we compare the proposed method in a random forgery scenario (RF) with BP and PGE. We observe that $\overline{\text{SDTW}}$ leads in four out of four cases to lower equal error rates when compared with BP and PGE (for SDTW three improvements and one slight deterioration can be observed). Even smaller equal error rates can be observed by using the ensemble methods. Similar to the SF scenario, we observe that SDTW + BP is beneficial on GPDS-75, while $\overline{\text{SDTW}}$ + BP is beneficial on MCYT-75.

Table IV: Equal error rates in a skilled forgeries scenario with 5 and 10 references on GPDS-75 and MCYT-75.

System	GPDS-75				MCYT-75			
	R5		R10		R5		R10	
BP	12.31		9.47		19.73		13.24	
PGE	12.40		9.64		16.53		10.67	
SDTW	9.91		7.02	(3)	16.36		11.38	
$\overline{\text{SDTW}}$	10.58		7.91		15.11	(2)	10.49	
SDTW + BP	8.53	(1)	6.36	(1)	15.82		10.04	(2)
SDTW + PGE	9.60	(3)	7.16		15.91		10.67	
$\overline{\text{SDTW}}$ + BP	8.71	(2)	6.53	(2)	13.60	(1)	8.98	(1)
$\overline{\text{SDTW}}$ + PGE	10.67		8.31		15.29	(3)	10.31	(3)

Table V: Equal error rates in a random forgeries scenario with 5 and 10 references on GPDS-75 and MCYT-75.

System	GPDS-75				MCYT-75			
	R5		R10		R5		R10	
BP	5.75		3.80		5.73		2.67	
PGE	4.43		3.12		5.19		2.13	(3)
SDTW	3.42		2.18	(3)	5.19		2.14	
$\overline{\text{SDTW}}$	3.66		2.18	(3)	3.86	(3)	1.86	(2)
SDTW + BP	2.94	(1)	1.96	(1)	4.40		1.86	(2)
SDTW + PGE	3.28	(2)	2.09	(2)	4.92		2.13	(3)
$\overline{\text{SDTW}}$ + BP	3.30	(3)	2.09	(2)	3.12	(1)	1.59	(1)
$\overline{\text{SDTW}}$ + PGE	3.68		2.38		3.60	(2)	1.86	(2)

Next, we compare the matching times of the different algorithms (i.e. BP, PGE, and SDTW) on graphs with different sizes, as shown in Table VI[5]. In fact, BP has cubic time complexity with respect to the number of nodes, while PGE offers linear time complexity. The proposed SDTW has quadratic time complexity with respect to the number of window positions given by the underlying DTW alignment [23], while the actual comparison at every window position by means of PGE has linear time complexity [21]. Naturally, we observe a substantial speedup when we compare the runtime of PGE with BP. SDTW is able to keep up with PGE with respect to the matching times. Since the optimal number of window positions is relatively small, we observe relatively low matching times between 10 and 27 ms.

Last but not least, we compare SDTW as well as the proposed ensemble methods with the following five state-of-the-art signature verification systems in Table VII.

- In [8], the Mahalanobis distance is used to compare slant and variability features.
- In [9] similar features as in [8] are used in conjunction with an HMM.

[5]Note that all performance-related experiments have been measured on the same machine (iMac 5K, 4GHz Intel Core i7, 32GB DDR3) in a single thread scenario.

Table VI: Matching time (ms) using different sizes of graphs. With $|\overline{V}|$ we denote the mean number of nodes of the graphs.

System					Average
GPDS-75	$\|\overline{V}\| = 315$	$\|\overline{V}\| = 218$	$\|\overline{V}\| = 168$	$\|\overline{V}\| = 139$	
BP	10,698.0	2,694.5	1,000.2	492.2	3,721.3
PGE	1.6	1.1	0.8	0.7	1.1
SDTW	23.6	16.5	12.8	10.2	15.8
MCYT-75	$\|\overline{V}\| = 330$	$\|\overline{V}\| = 236$	$\|\overline{V}\| = 189$	$\|\overline{V}\| = 160$	
BP	12,597.6	3,423.8	1,389.1	805.3	4,554.0
PGE	2.0	1.5	1.4	1.1	1.5
SDTW	27.0	19.8	16.0	13.6	19.1

- Contour-based features in combination with χ^2-distance have been employed in [5].
- A recent approach [13] makes use of a Deep Multitask Metric Learning (DMML) that combines Histogram of oriented Gradients (HoG) and Discrete Radon Transform (DRT) features.
- The most recent reference system uses a CNN in combination with Spatial Pyramid Pooling (SPP) [6].

Note that for this comparison we measure the EER in two threshold scenarios, i.e. a global and a local scenario. In the global threshold scenario, no user-dependent threshold adoption is conducted, while in the local threshold scenario, a *posteriori* user-dependent score normalisation is applied [9]. Hence, local thresholds lead to lower EER, whereas global thresholds are considered to be more realistic for practical applications.

If we compare $\overline{\text{SDTW}}$ with the non learning-based reference systems (i.e. [5], [8]), we observe a clear improvement especially in the case of random forgeries (using a local threshold scenario). Moreover, we observe that $\overline{\text{SDTW}}$ + BP is among the three best systems in each test case. In fact, only the CNN-based reference system [6] leads to better results than the proposed framework. This is quite remarkable, as the proposed method is optimised on a relatively small and independent training set (i.e. GPDS-last100), while CNN approaches are known for extensive training phases on very large datasets.

Table VII: Equal error rates compared to state-of-the-art methods on MCYT-75 using R10.

System	Global		Local	
	RF	SF	RF	SF
Mahalanobis (Slant,Variability) [8]	-	-	7.26	22.13
HMM (Slant,Envelope) [9]	-	-	1.14	9.28
χ^2 (Contour) [5]	-	-	1.18	6.44
DMML (HoG,DRT) [13]	0.37 (2)	9.86	-	-
CNN (SPP) [6]	0.19 (1)	3.64 (1)	0.03 (1)	-
SDTW	2.14	11.38	1.46	5.87
$\overline{\text{SDTW}}$	1.86	10.49	0.58 (3)	5.42
SDTW + BP	1.86	10.04 (3)	1.15	5.33
SDTW + PGE	2.13	10.67	1.24	5.24 (3)
$\overline{\text{SDTW}}$ + BP	1.59 (3)	8.98 (2)	0.29 (2)	4.09 (1)
$\overline{\text{SDTW}}$ + PGE	1.86	10.31	0.67	4.98 (2)

V. Conclusion

Most of the available signature verification frameworks are based on statistical rather than structural representations of the signatures. However, some promising graph-based approaches have been proposed in recent years. These approaches exploit the possibility of representing the inherent topological characteristic of the handwriting in a natural and comprehensive way. Actually, both the structure and the size of a graph can be adapted to the size and complexity of the underlying signature. Due to several fast graph matching algorithms proposed in the last decade, graphs have emerged as a versatile alternative to more traditional approaches for signature verification.

In the present paper, we follow this line of research and propose a novel graph-based approach for signature verification that combines Dynamic Time Warping (DTW) with a recently proposed linear time graph dissimilarity measure, i.e. Polar Graph Embedding distance (PGEd). In contrast with most of the graph-based signature verification frameworks proposed so far, this novel procedure allows to match graphs from several local perspectives. In particular, we employ a sliding window approach to compare PGEd at different local positions on different subgraphs. The resulting distance matrix is then used to find an optimal alignment between the sequences of subgraphs by means of DTW. We denote this novel procedure by Structural Dynamic Time Warping (SDTW).

In an experimental evaluation on two widely used datasets, i.e. GPDS-75 and MCYT-75, we show that the proposed approach can outperform graph-based state-of-the-art reference systems. In fact, SDTW can be seen as well balanced tradeoff between runtime and accuracy when compared with other graph matching algorithms. Moreover, we show that the proposed approach can be combined with global graph matching approaches. This combinatorial graph matching strategy allows us to keep up with recent Deep Learning approaches without the need of an *a priori* learning step. That is, the proposed graph-based signature verification approach is not depending on a large amount of training data. This is a clear advantage in case of signature verification, where the number of reference signatures is often rather small and/or expensive to acquire.

Acknowledgment

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

References

[1] D. Impedovo and G. Pirlo, "Automatic signature verification: The state of the art," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 2008.

[2] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Offline handwritten signature verification — Literature review," in *International Conference on Image Processing Theory, Tools and Applications*. IEEE, 2017, pp. 1–8.

[3] M. Diaz, M. A. Ferrer, D. Impedovo, M. I. Malik, G. Pirlo, and R. Plamondon, "A Perspective Analysis of Handwritten Signature Technology," *ACM Computing Surveys*, vol. 51, no. 6, pp. 1–39, 2019.

[4] M. A. Ferrer, J. Alonso, and C. Travieso, "Offline geometric parameters for automatic signature verification using fixed-point arithmetic," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 993–997, 2005.

[5] A. Gilperez, F. Alonso-Fernandez, S. Pecharroman, J. Fierrez, and J. Ortega-Garcia, "Off-line signature verification using contour features," in *International Conference on Frontiers in Handwriting Recognition*. Concordia University, 2008.

[6] L. G. Hafemann, L. S. Oliveira, and R. Sabourin, "Fixed-sized representation learning from offline handwritten signatures of different sizes," *International Journal on Document Analysis and Recognition*, vol. 21, no. 3, pp. 219–232, 2018.

[7] A. Piyush Shanker and A. Rajagopalan, "Off-line signature verification using DTW," *Pattern Recognition Letters*, vol. 28, no. 12, pp. 1407–1414, 2007.

[8] F. Alonso-Fernandez, M. Fairhurst, J. Fierrez, and J. Ortega-Garcia, "Automatic Measures for Predicting Performance in Off-Line Signature," in *IEEE International Conference on Image Processing*. IEEE, 2007, pp. I–369–I–372.

[9] J. Fierrez-Aguilar, N. Alonso-Hermira, G. Moreno-Marquez, and J. Ortega-Garcia, "An Off-line Signature Verification System Based on Fusion of Local and Global Information," in *International Workshop on Biometric Authentication*. Springer, 2004, pp. 295–306.

[10] P. S. Deng, H.-Y. M. Liao, C. W. Ho, and H.-R. Tyan, "Wavelet-Based Off-Line Handwritten Signature Verification," *Computer Vision and Image Understanding*, vol. 76, no. 3, pp. 173–190, 1999.

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

[12] M. A. Ferrer, J. F. Vargas, A. Morales, and A. Ordonez, "Robustness of Offline Signature Verification Based on Gray Level Features," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 966–977, 2012.

[13] A. Soleimani, B. N. Araabi, and K. Fouladi, "Deep Multitask Metric Learning for Offline Signature Verification," *Pattern Recognition Letters*, vol. 80, pp. 84–90, 2016.

[14] A. Bansal, P. Nemmikanti, and P. Kumar, "Offline Signature Verification Using Critical Region Matching," in *International Conference on Future Generation Communication and Networking Symposia*. IEEE, 2008, pp. 115–120.

[15] T. Fotak, M. Bača, and P. Koruga, "Handwritten signature identification using basic concepts of graph theory," *WSEAS Transactions on Signal Processing*, vol. 7, no. 4, pp. 117–129, 2011.

[16] P. Maergner, K. Riesen, R. Ingold, and A. Fischer, "A Structural Approach to Offline Signature Verification Using Graph Edit Distance," in *International Conference on Document Analysis and Recognition*. IEEE, 2017, pp. 1216–1222.

[17] P. Maergner, N. Howe, K. Riesen, R. Ingold, and A. Fischer, "Offline signature verification via structural methods: Graph edit distance and inkball models," in *International Conference on Frontiers in Handwriting Recognition*. IEEE, 2018, pp. 163–168.

[18] P. Maergner, V. Pondenkandath, M. Alberti, M. Liwicki, K. Riesen, R. Ingold, and A. Fischer, "Offline signature verification by combining graph edit distance and triplet networks," in *International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*. Springer, 2018, pp. 470–480.

[19] K. Riesen, M. Neuhaus, and H. Bunke, "Bipartite Graph Matching for Computing the Edit Distance of Graphs," in *International Workshop on Graph-Based Representations in Pattern Recognition*, vol. 6. Springer, 2007, pp. 1–12.

[20] A. Fischer, C. Y. Suen, V. Frinken, K. Riesen, and H. Bunke, "Approximation of graph edit distance based on Hausdorff matching," *Pattern Recognition*, vol. 48, no. 2, pp. 331–343, 2015.

[21] M. Stauffer, A. Fischer, and K. Riesen, "Filters for Graph-Based Keyword Spotting in Historical Handwritten Documents," *Pattern Recognition Letters*, no. In Press, 2018.

[22] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[23] D. Berndt and J. Clifford, "Using Dynamic Time Warping to Find Patterns in Time Series," in *Workshop on Knowledge Discovery in Databases*, 1994, pp. 359–370.

[24] K. Riesen and H. Bunke, "Approximate graph edit distance computation by means of bipartite graph matching," *Image and Vision Computing*, vol. 27, no. 7, pp. 950–959, 2009.

[25] A. Fischer, M. Diaz, R. Plamondon, and M. A. Ferrer, "Robust score normalization for DTW-based on-line signature verification," in *International Conference on Document Analysis and Recognition*. IEEE, 2015, pp. 241–245.

[26] M. A. Ferrer, M. Diaz-Cabrera, and A. Morales, "Static signature synthesis: A neuromotor inspired approach for biometrics," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 667–680, 2015.

[27] J. Ortega-Garcia, J. Fierrez-Aguilar, D. Simon, J. Gonzalez, M. Faundez-Zanuy, V. Espinosa, A. Satue, I. Hernaez, J.-J. Igarza, C. Vivaracho, D. Escudero, and Q.-I. Moro, "MCYT baseline corpus: a bimodal biometric database," *Vision, Image, and Signal Processing*, vol. 150, no. 6, p. 395, 2003.