UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

FELIX EDUARDO HUAROTO PACHAS

An Offline Writer-Independent Signature Verification Method with Robustness Against Scalings and Rotations

Thesis presented in partial fulfillment
of the requirements for the degree of
Master of Computer Science

Advisor: Prof. Dr. Eduardo Simões Lopes Gastal

Porto Alegre
June 2022

CIP — CATALOGING-IN-PUBLICATION

Huaroto Pachas, Felix Eduardo

An Offline Writer-Independent Signature Verification Method with Robustness Against Scalings and Rotations / Felix Eduardo Huaroto Pachas. – Porto Alegre: PPGC da UFRGS, 2022.

76 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2022. Advisor: Eduardo Simões Lopes Gastal.

1. Signature Verification. 2. Offline Signature Verification. 3. Writer independent models. I. Gastal, Eduardo Simões Lopes. II. Título.

ABSTRACT

Handwritten signatures are still one of the most used and accepted methods for user authentication. They are used in a wide range of human daily tasks, including applications from banking to legal processes. The *signature verification* problem consists of verifying whether a given handwritten signature was generated by a particular person, by comparing it (directly or indirectly) to genuine signatures from that person. In this research work, a new offline writer-independent signature verification method is introduced (named **VerSig-R**), based on a combination of handcrafted Moving Least-Squares features and features transferred from a convolutional neural network. In our experiments, **VerSig-R** outperforms state-of-the-art techniques on Western-style signatures (CEDAR dataset), while also obtaining competitive results on South Asian-style handwriting (Bangla and Hindi datasets). Furthermore, a wide range of experiments demonstrate that **VerSig-R** is the most robust in relation to differences in scale and rotation of the signature images. This work also presents a discussion on dataset bias and on cross-dataset performance of **VerSig-R**, as well as a small user study showing that the proposed technique outperforms the expected human accuracy on the signature-verification task. Finally, a discussion on the impact of the number of signature examples (per writer) used during training on performance and execution time is presented.

Keywords: Signature Verification. Offline Signature Verification. Writer independent models.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

With the rapid increase of developed software integrated into many kinds of processes, some research effort has been made with the objective of having powerful and effective techniques to authenticate legitimate users in order to access sensitive information, both personal and professional. Some approaches to authenticate users are based on "something you know," such as PINs or passwords, and/or "something you have," such as smart cards. These approaches have the drawback that users can easily forget passwords or lose smart cards. Also, users can be exposed to social engineering tactics, revealing their PIN or password to malicious third parties (HADNAGY, 2010). In this sense, biometric authentication is a good solution, since in this case the authentication process is based on "something you are" or "something you do." Specifically, biometric authentication could be classified into two types: physiological (fingerprints, face recognition, iris recognition, etc.) and behavioral (voice or speech recognition, keystroke dynamics and handwritten signatures) (MOHAMMED et al., 2015).

In general, people are mostly accostumed to automated physiological biometric recognition, using them in daily life tasks such as unlocking a mobile phone (with facial or fingerprint recognition), or verifying physical presence at university or work by fingerprint recognition. However, there are some activities in which physiological biometrics are not sufficient, due to legal restrictions or social conventions. These activities still required a socially and legally accepted behavioral biometric recognition technique: **Handwritten signatures**, although the signature verification process is performed mostly in a non-automated way and often based only on the judgment of the administrative staff (sometimes with no scientific rigor nor experience). Some examples of these activities are: signing a contract or any kind of banking and legal processes. On the other hand, non-automated handwritten signature verification processes are user-friendly and non-invasive, since it does not require any additional step for the user apart from performing the signature itself (MOHAMMED et al., 2015; KUMAR; BHATIA, 2016).

Handwriting and signature verification are not trivial problems since, although performing a signature is a behavioral trait learnt and perfected over a lifetime period by the writer, it presents extremely high intraclass variability: no signature is identical to another, even if performed by the same writer with some seconds of difference. In addition to this, as expected, interclass variability is present when comparing signatures from two different writers. So, in order to classify two signatures as belonging to the same

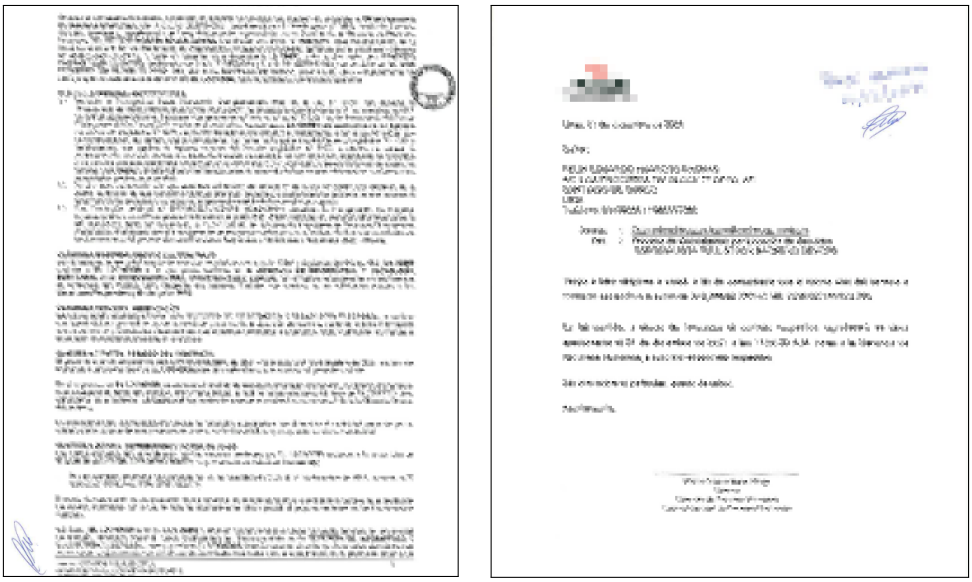
class/writer (genuine signature pair) or two different classes/writers (forgery signature pair), the intraclass dispersion ideally should be as low as possible, while the interclass dispersion needs to be as high as possible (KALERA; SRIHARI; XU, 2004).

In general, Handwritten Signature Verification consists of verifying whether a particular handwritten signature is genuine (i.e., was generated by a particular person of interest) or if it is a forgery (i.e., was generated by someone else trying to mimic the genuine signature of the first person, in order to impersonate them). When considering automated handwritten signature verification, this problem can be approached using either *online* or *offline* information (MOHAMMED et al., 2015) depending on the way handwritten signature information is collected. In the online case, signature information is collected using a helper device which could be a tablet, a special pen or a handwriting digitizer, these devices collect writing-time signature information, i.e. x and y coordinates for each signature point at each moment in time, the total time employed to perform the signature, pen inclination, applied pressure, etc. This online information is supposed to be unique for each writer, identifying them based on these behavioral biometrics. On the other hand, in the offline case, the handwritten signature is only available as a static image, which is normally a scanned piece of paper containing the signature (Figure 3.2a). Then, behavioral biometric information about the signer has to be collected indirectly from the handwritten signature image, turning the signature verification into a more difficult problem to solve (MOHAMMED et al., 2015), since behavioral information about the user is not available directly (KALERA; SRIHARI; XU, 2004). Nonetheless, offline signature verification it is still the most used and socially approved approach, since many legal and banking processes base their legality on the verification of the truthfulness of static handwritten signatures (without the necessity of any helper or additional device, which would complicate the process), but still performing the signature verification in a non-automated way.

Handwritten Signature Verification could also be classified as to be either *writer dependent* or *writer independent* (KUMAR; BHATIA, 2016; MOHAMMED et al., 2015). A writer-dependent approach for signature verification builds a distinct and unique classification model for each writer, in order to classify signatures as either genuine or forgeries for each writer. This approach tends to achieve better results, because the classifiers are provided with writer-specific signature information, but in contrast this kind of classifier needs a lot more genuine signature examples from each writer (KUMAR; BHATIA, 2016; MOHAMMED et al., 2015). Achieving this could be tedious, challenging or even impos-

Figure 1.1: Example of signatures performed by the author in real-life documents.

(a) Vertical signature (b) Slightly rotated signature



Source: (a-b) Real documents signed by the author.

sible if one considers for example a large institution trying to implement a Handwritten Signature Verification system. Furthermore, in the writer-dependent approach each new user (or a user who decided to change their signature) has to be enrolled in the system in order to generate a specific model for them.

On the other hand, in the writer-independent approach a single classification model is built upon a set of handwritten signatures. This classification model is able to deal with the classification of any new signature without the necessity to create a new specific model or any kind of retraining (KUMAR; BHATIA, 2016), which leads to a better generalization for the classification task. This approach usually operates on top of *signature pairs*; that is, given a pair of images (each image containing a single handwritten signature), the classification model classifies the pair as either genuine (if both signatures are believed to be written by the same person), or as a forgery (if the signatures seem to be from two different writers). This approach also performs well even when a small number of signatures is available per writer (OLIVEIRA; JUSTINO; SABOURIN, 2007), due to the fact that the model is trained on signature-pair examples of several writers (DEY et al., 2017).

Previous research (KUMAR; BHATIA, 2016; MOHAMMED et al., 2015; OLIVEIRA; JUSTINO; SABOURIN, 2007; DEY et al., 2017) only consider datasets with signature images in a horizontal position (Figure 3.2a), and with similar sizes. Conversely, in real life scenarios, writers could sign a document in any direction and with arbitrary sizes, from

a slightly rotated signature to a fully vertical signature (see Figure 1.1), even digitization operations could introduce some rotation and scaling variations to the signature image. In this sense, a good handwritten signature verification technique should be rotation and scaling invariant, i.e. the accuracy of the technique must not be affected by the size and rotation angle of the signature image being evaluated when comparing with a genuine signature image.

In this work a new, offline, writer-independent approach for signature verification is proposed. The proposed approach is named **VerSig-R** which stands for **[Ver]ification of [Sig]natures that is [R]obust to Scalings and Rotations**. As shown by our experiments (Table 4.3), **VerSig-R** significantly outperforms state-of-the-art techniques on the CEDAR dataset (Western-style handwriting) (KALERA; SRIHARI; XU, 2004), while performing on-par with the state-of-the-art on the Bangla and Hindi datasets (South Asian-style handwriting) (PAL et al., 2016). Furthermore, VerSig-R outperforms the state-of-the-art when one considers the more challenging situation, i.e., where the signatures are subjected to variations in scale and rotation. As such, VerSig-R is more robust against this kind of data variability (Section 4.4).

VerSig-R works by describing each signature image by a set of hybrid features: some handcrafted and some obtained from a pretrained CLIP neural network (RADFORD et al., 2021). The handcrafted features were designed based on Moving Least-Squares goodness-of-fit, with simplicity and robustness in mind. Furthermore, an alternative pipeline for obtaining the CLIP features for each signature image is proposed, this alternative pipeline is supported by an ablation study (Table 4.4). Finally, for the classification task **VerSig-R** uses a simple linear Support Vector Machine (SVM) classifier with no hyperparameter tunning.

The **contributions** of the present work include:

- A new offline writer-independent signature verification approach, which outperforms the state-of-the-art for western-style handwritten datasets and is robust against changes in scale and rotation of the signature images. This method (**VerSig-R**) uses a novel set of handcrated features based on the idea of Moving Least-Squares (MLS) (LANCASTER; SALKAUSKAS, 1981) and a set of features transferred from the CLIP neural network with a new preprocessing pipeline proposed by the author;
- A discussion of unintended bias on signature-verification datasets and their impact on metrics calculated over it, with a proposal of how to remove such bias (defining

new Unbiased datasets), supported by experiments;

- The proposal of variations on existing datasets in order to consider scalings and rotations of the images, providing additionally comprehensive experiments on scale and rotation invariance of existing signature-verification methods;
- A study about the impact of the number of references during training on the performance of the classifier as well as the impact of using only random forgeries during the training phaseAlso, a cross-dataset validation in order to determine the generalization power of the trained classifiers and an examination of the execution time across all the mentioned experiments.
- A small user study, in order to quantify human ability to determine whether a signature image is a genuine or forgery, considering untrained individuals.

This work is organized as follows: Chapter 2 introduces the main concepts and a literature revision on the signature verification field; Chapter 3 presents **VerSig-R** in a detailed manner while Chapter 4 analyzes the results obtained from a wide range of experiments. Finally, our conclusions and proposals for future work are listed both in Chapter 5 and Chapter 6 respectively.

2 BACKGROUND AND RELATED WORK

A lot of research effort has been done in the field of signature verification, either using online or offline information, as well as proposing writer-dependent or writer-independent approaches (MOHAMMED et al., 2015; KUMAR; BHATIA, 2016). Most of this research effort has been focused in extracting good feature-representations of an individual genuine signature, in order to discriminate effectively genuine signatures from forgeries. A forgery is a signature generated by a person (the forger) trying to mimic the genuine signature of another legitimate person (the writer) in order to impersonate him.

Forgeries can be ascribed to one of three categories, in order of decreasing complexity: skilled, unskilled, or random (KUMAR; BHATIA, 2016; MOHAMMED et al., 2015). A *skilled forgery* is one in which the forger tries to mimic the original writer's signature after having seen it, and has time to practice performing the signature in order to reproduce it accurately. On the other hand, an *unskilled (or simple) forgery* is one where the forger does not have access to the actual signature of the original writer, but knows only his or her name. Finally, a *random forgery* is produced knowing neither the original signature nor the name of the original writer (in this case signature pairs are usually generated by pairing genuine signatures from two different writers (KUMAR; BHATIA, 2016)). The focus of the present work is to deal mainly with skilled forgeries, which is the most difficult case to approach since interclass variability between genuine and forgery signatures is extremely smaller. However, the present work also includes some experiments with random forgeries in order to determine if **VerSig-R** is capable of training with random forgeries only while performing accurately on skilled forgeries during the test/validation phase. This is motivated by the fact that, in real-world situations, it is often complicated to have a large number of skilled forgeries during the training phase.

Since the main focus of this research work is to deal with skilled forgeries, signature image datasets containing this kind of forgery samples were used to train and test the **VerSig-R** method. For example, the **CEDAR** dataset (KALERA; SRIHARI; XU, 2004), which is provided by the Center of Excellence for Document Analysis and Recognition from Buffalo University;[1] and **BHSig260** (PAL et al., 2016), which in fact contains two different datasets, one composed by Hindi writers and the other composed by Bangla writers. Additionally, two other datasets used in the present work are the **MCYT** (ORTEGA-GARCIA et al., 2003; FIERREZ-AGUILAR et al., 2004) and **GPDS** (FERRER; DIAZ-

[1] <http://www.cedar.buffalo.edu/NIJ/data/signatures.rar>

CABRERA; MORALES, 2015; FERRER; DIAZ-CABRERA; MORALES, 2013) datasets, the last one being generated synthetically. A set of statistics about the just mentioned datasets such as the number of writers composing them, and the number of reference samples for genuine and forgery class per writer, can be encountered in Table 4.1.

Although *online* and *offline* signature verification methods, as described in Chapter 1, pursue the same objective, they are not directly comparable because of the way they extract the signature's information (KALERA; SRIHARI; XU, 2004). It is worth to mention that online signature verification methods currently perform better than offline methods (MOHAMMED et al., 2015), because of the advantage of having dynamic information about the signature. However, online methods are less usable and user-friendly, because normally it introduces a helper device such as an electronic pen and a digitizer, making it difficult for the writer to perform their actual signature as done with pen and paper. The following sections present a discussion of previous research works for online and offline signature verification, as well as writer-dependent and writer-independent approaches. Special emphasis is given to offline writer-independent techniques.

2.1 Online Signature Verification

Online signature verification is centered on capturing the position of the pen tip as a function of time, although any other information could be associated with it like applied pressure, inclination, etc. In order to generate this kind of information in real time, normally a helper device is needed. Such a device could be a tablet, a camera, electronic boards, etc.

For example, Munich and Perona (2003) present a camera-based method which, according to the authors, was the first signature acquisition study using a general purpose camera. In the same direction they claim to achieve similar or better performance than other camera-based identification systems like face or hand shape recognition. The mentioned system requires a video camera, a frame grabber (to digitize the input camera image) and a computer. The main idea is to track the pen tip with a tip detector and then predict the position of the tip in every image taken by the camera based on the current position, velocity and acceleration of the pen. Additionally, the most likely position of the pen tip is estimated for missing frames. The authors used Dynamic Programming Matching (DPM) (SAKOE; CHIBA, 1978), first introduced in the field of speech recognition and also known as Dynamic Time Warping (DTW) (RABINER, 1993), to find the correspondence

between the signature's stroke points of two signatures, using some predefined metrics, e.g., time. Despite the fact that the authors claim to achieve very good results (3.95% verification error for skilled forgeries) and that the method does not require any dedicated hardware, the method still needs some helper devices (camera and frame grabber) and could introduce some artifacts or additional variability since writers are subjected to a stressful and/or unfamiliar environment when performing the signature. This makes the described method unsuitable for real-life scenarios.

For other works regarding online information, the author kindly refers the reader to relevant recent surveys and techniques (MOHAMMED et al., 2015; KUMAR; BHATIA, 2016; HAFEMANN; SABOURIN; OLIVEIRA, 2019a; DIAZ et al., 2019; HAMEED et al., 2021).

2.2 Offline Signature Verification

Offline signature verification is the least invasive signature verification approach since it is only based on a static image of the signature. Writers perform the signature in a piece of paper, then this piece of paper is scanned and used as input for the classifier. No helper devices are needed. The main challenge is to generate a good feature representation of the images of the genuine signatures that minimizes intraclass variability (same writer) and maximizes interclass variability (different writers). Sections 2.2.1 and 2.2.2 present a literature review of offline writer-dependent and writer-independent approaches.

2.2.1 Writer-Dependent Approaches

Writer-dependent approaches generate a unique classifier for each writer being evaluated, i.e., during the training, testing, and validation phases, reference samples from a unique writer are used. This means that, as in any generation of a classification model, any change in the data (for example if the writer decides to change their signature, or the intraclass variability changes sufficiently enough), the classification model needs to be retrained. Also, to register a new user in a signature verification software that uses this approach, a new classification model needs to be trained with information about this user, needing (normally) a large number of signature reference samples from this writer.

Soleymanpour, Rajae and Pourreza (2010) present a writer dependent model using

the Contourlet transform as feature extractor and an SVM as classifier. They tested their model over a Persian (600 signatures, 20 signatures per writer) and a Turkish signature image dataset. The authors claim to achieve 6.5% of FAR (False Acceptance Rate), 2.5% of FRR (False Rejection Rate) and 4.5% of AER (Average Error Rate), which in practical terms could be understood as the EER (Equal Error Rate) metric (see Section 4.2 for the EER definition).

Hafemann, Sabourin and Oliveira (2016) use a Deep Convolutional Neural Network for feature learning, i.e., feature generation, and use these learned features to generate writer-dependent models for the classification task. In this way, the features are obtained from the data instead of generated manually based on the data. In their work the authors use GPDS-960 (VARGAS et al., 2007) and Brazilian PUC-PR datasets (FREITAS et al., 1998), obtaining good FAR metrics for random and simple forgeries but not so good for the skilled ones. These results are perfectly reasonable considering the difficulty level of each kind of dataset.

Hafemann, Sabourin and Oliveira (2019b) propose to formulate the signature verification problem as a meta-learning problem based on learning across tasks. The authors proposed to generate a meta classifier (in a writer-independent way) and subsequently adapt it to a particular user (writer-dependent) in order to perform classification. The authors mention that their proposed method is "as scalable as a writer-independent method" since there is a single meta-model that can be adapted for writer-dependent application with lighweight operations (a few gradient descent steps). However, one could consider this work as still writer dependent, because despite the fact that a single meta-classifier is learned and stored, it has to be adapted to a particular user (according to the authors with lightweight operations) in order to perform classification. Additionally, the best results obtained with this method were obtained when training and testing over the GPDS-960 dataset obtaining 5.16% and 4.39% of EER using 5 and 12 reference signatures respectively, and a global threshold. The authors also performed cross-dataset validation, training on the GPDS dataset and testing over the MCYT, CEDAR and PUC-PR datasets, concluding that the generalization for other datasets is much worse. Finally, a combined dataset was generated with images from all the mentioned datasets, used during the training phase, showing that results do not improve with this new test. The authors consider that the GPDS dataset dominates training because of being ten times larger than the other datasets used. In summary, the authors conclude that their proposed method "requires a large amount of data and is sensitive to changes in the operating conditions."

Another interesting work is the by Pal et al. (2016), in which a new set of texture-based features are presented, namely Local Binary Patterns (LBP) and Uniform Local Binary Patterns (ULBP) considering non-Latin based signatures. The authors also introduce a new large-scale offline dataset, BHSig260 (listed as Bangla and Hindi in Table 4.1). A Nearest Neighbour classifier is considered in combination with the Euclidean Distance. The mentioned method should be considered as writer-dependent because it tries to solve a one-class classification problem, considering only genuine signatures at the training phase in order to determine if the questioned signature belongs to any of the writer classes, i.e., if it is a genuine signature or not. The training phase was performed with 8 and 12 signatures per class (i.e. per writer), being the last configuration which achieves better results. Extremely similar results were achieve by both LBP and ULBP features, with 33% of Equal Error Rate (EER) for the BHSig260 dataset. Additionally it is worth mentioning that the Bangla part of the BHSig260 dataset obtains worse results than the Hindi subset, so one can logically conclude that the Bangla part of the dataset configures a more challenging problem. Finally, some tests were performed over the GPDS-100 signature dataset (FERRER; ALONSO; TRAVIESO, 2005), obtaining similar results for EER.

2.2.2 Writer-Independent Approaches

The main focus of the present work is to introduce a novel method for handwritten signature verification using offline information and a writer-independent approach. Offline signature verification approaches do not require any helper device and do not modify in any way the protocols established for the manual signature verification in real-life scenarios. In this sense, offline approaches are not invasive, since the writer will perform the signature in the same way they are used to, and for this reason they fit perfectly in most legal and banking processes. On the other hand, offline signature verification is the most difficult problem to deal with (see Chapter 1), and offline approaches perform worse (on average) than online approaches. Finally, we recall that this work deals mainly with skilled forgeries, although some tests were performed with random forgeries during the training phase (Section 4.8).

Most offline signature verification methods take into account the whole signature image as source for feature extraction. For example, in the work presented by Kalera, Srihari and Xu (2004), a set of global, statistical and geometrical/topological features

are extracted directly from images. These kinds of features correspond to what the authors called Gradient, Structural, and Concavity (GSC) features. The gradient features are used to detect local characteristics and to obtain information about stroke shape in short areas. The structural features provide similar information to the gradient features but for longer distances, so they are used to get information about stroke trajectories. Finally, concavity features are used to find stroke relationships across the signature image. Statistical distance distributions are calculated based on the mentioned features and are used for the classification task with a Bayes Classifier (RISH et al., 2001).

Huang and Yan (1997) propose use of geometric features in combination with a neural network classifier, specifically one based on a Multi-Layer Perceptron, using offline information. Artificial genuine and forgery signature samples are generated from a small set of genuine signatures by applying "perturbations" to them. Some of the geometric features extracted are the following: skeleton of the pen trace (obtained by applying a 3×3 operator over a gray-labeled image, pixel is considered if it is a gray-level value local peak), signature outline (corresponding to the pixels with values above a 25% gray-level intensity and have less than 8 neighbors in a 3×3 pixels area), ink distribution, high pressure region feature (where the writer puts more emphasis, it is usually the darker area in the scanned image), directional frontiers, number of constituent parts of the signatures (could be understood as the connected components of the signature), the centroid location and the width and height of each signature constituent parts. A feature alignment process is considered before the training of the neural network. The authors claim to achieve an average correct classification rate of 90% using their artificially generated dataset with test size of 24 genuine signatures and 144 forgery samples.

Ferrer, Alonso and Travieso (2005) present another set of geometric features based on the signature envelope, i.e., the signature shape or outline, and the interior stroke distribution in polar and Cartesian coordinates. The authors tested their proposed features with different classifiers such as hidden Markov Models (HMM), support vector machines (SVM) and Euclidean distance classifiers. Signature outline is obtained by dilating the signature and filling it when necessary, then the outline is represented as a sequence of Cartesian coordinates in counterclockwise orientation beginning from the geometric center of the outline. They achieve their best results when using an HMM but only test on random (general error percentage around 4%) and simple forgeries (general error percentage around 17%).

In the same direction, Kumar, Sharma and Chanda (2012) present a set of features

describing the signatures' shape in terms of the spatial distribution of black pixels around a candidate signature pixel (surroundedness). The set of features also provides texture information through the correlation among signature pixels in the neighborhood of a candidate signature pixel. Surroundedness is calculated with different radii $r = 1, 2, \ldots 11$ centered on the candidate pixel, while the circle with the corresponding radius is calculated using the Chebyshev distance (CANTRELL, 2000). Surroundedness information is summarized using statistical measures like entropy and first, second, and third orders of moment producing a vector of 44 dimensions. CEDAR (KALERA; SRIHARI; XU, 2004) and GPDS300 (VARGAS et al., 2007) signature datasets were tested with two classifiers, namely Multi Layer Perceptron and Support Vector Machine with Radial Basis Function kernal (RBF-SVM), obtaining good accuracy results (91.67% and 86.24% respectively).

Bertolini et al. (2010) propose an offline writer-independent signature verification method which tries to simulate signature strokes using Bezier curves. Based on the bezier curves a set of graphometric features are extracted, such as: density, slant, distribution, and curvature. In other words, the authors try to obtain the same information that an online signature verification approach would provide, but using a static image as input. The method performs in a grid-based manner, defined over the whole signature area, i.e., the signature bounding box. An ensemble of classifiers (64 in total, 16 different grid configurations for each of the four graphometric features just mentioned) is mounted over the set of four forensic document examination characteristics previously mentioned (density, slant, distribution, and curvature) and a genetic algorithm is used to determine which of the base classifiers are the most relevant for the classification task. The method proposed by Bertolini et al. (2010) was tested against a private signature image dataset composed of a total of 100 writers (generated by undergraduate students in four different sessions, collected on A4 white sheet paper and scanned in 600 dpi grayscale), 40, 20, and 40 writers were used for training, validation and testing respectively. To generate genuine sample pairs, four randomly chosen genuine signatures were used per writer which leads to a total of six sample pairs per writer, considering a total of 40 writers for training it leads to 240 genuine pairs. On the other hand, to generate negative samples they take two genuine signatures for the first 36 writers and compare it to two signatures for four different writers chosen randomly which leads to a total of 288 negative samples. The authors show that their proposed ensemble of classifiers based on the four graphometric features are quite efficient and can reduce considerably the acceptance of forgeries, i.e. reduce the False Acceptance Rate (FAR), which is arguably more important than decreasing False Rejection

Rate (FRR) in the context of signature verification systems (i.e., accepting a false signature has arguably worse consequences than rejecting a genuine signature).

Some other works proposed to extract features from signature images using a Convolutional Neural Network, i.e. feature learning from CNNs. For example, the work by Alvarez, Sheffer and Bryant (2016) introduces a completely CNN-based feature extraction method, based on the VGG-16 CNN architecture (SIMONYAN; ZISSERMAN, 2014) which uses common learnable parameters such as fully connected layers and ReLU as activation function. A transfer learning process was applied to the CNN architecture using weights pretrained on ImageNet (DENG et al., 2009). The dataset used comes from the International Conference on Document Analysis and Recognition (ICDAR) 2011 SigComp international signature verification competition (LIWICKI et al., 2011), which contains online and offline information. The offline part contains Dutch (about 25 genuine signatures and 11 forged signatures for 10 writers) and Chinese (about 25 genuine signatures and 30 forged signatures for also 10 writers) signatures. The authors performed writer-dependent and writer-independent tests over the mentioned dataset, getting test accuracies of 94% and 88% for Dutch and Chinese signatures respectively. It is worth to mention that the authors indicate that they split data for each language in 80%, 10% and 10% for training, validation and testing respectively in a random manner, which could lead to examples of signatures from the same writer used in training **and** testing time. When the authors performed tests effectively separating by writer on the training and test data, they show test accuracies of 76% on the Dutch dataset, which is not far better from a dummy classifier accuracy (67%). Finally, writer-dependent experiments are also not shown to be much better (67% for test accuracy).

Another work that applies the concept of feature learning from a CNN is the one presented by Hafemann, Sabourin and Oliveira (2016), which although presents a writer-dependent classification technique, relies on a writer-independent feature learning process. Some preprocessing operations were applied over the images before passing them to the CNN, e.g. background removal (based on Otsu's algorithm (OTSU, 1979)), followed by an image inversion (i.e. white background is mapped to zero values). This work was tested on the GPDS (VARGAS et al., 2007) and PUC-PR datasets (FREITAS et al., 1998). Hafemann, Sabourin and Oliveira (2017) extend the work mentioned before with tests over MCYT and CEDAR datasets, mantaining their approach for writer-independent feature learning and writer-dependent classification, but this time they include skilled forgeries at training time. Additionally, Souza, Oliveira and Sabourin (2018) perform an analysis of

the goodness of the features obtained by Hafemann, Sabourin and Oliveira (2017), but this time using a writer-independent approach. The authors use the Dichotomy transformation, initially proposed by Cha and Srihari (2000), which transforms a multi-class problem into a two-class problem. In particular, the authors calculate feature distances between the same writer samples and categorized as *within autor* class, while feature distances between samples for different writers were also computed and categorized as *between author* class, configuring this way the two classes for classification. An SVM classifier was used based on the RBF kernel, $\gamma = 2^{-11}$ and regularization parameter $C = 1.0$. The authors claim to achieve 4.90% in EER metric using a global threshold, and 1.48% in EER using a user-specific threshold (considering a different optimal threshold for each writer) in the PUC-PR dataset, using 30 samples per writer, outperforming the previous work (HAFEMANN; SABOURIN; OLIVEIRA, 2017) (which uses a writer-dependent approach). The authors also performs tests over the GPDS-160 and GPDS-300 datasets obtaining the best EER results of 7.01% and 7.96% using 12 samples per writer, but in this case not outperforming the writer-dependent technique by Hafemann, Sabourin and Oliveira (2017). Finally, it is worth to mention that the authors also conclude that the number of samples per writer used for training is directly proportional to the reduction of the EER metric.

One of the top-performing offline writer-independent methods is the SigNet method presented by Dey et al. (2017). It uses a Convolutional Siamese Network to extract feature information from signature images, configuring signature image pairs as input for the CNN. Before feeding each image to the network, some preprocessing steps are required, including: resizing the images to a standard size (155×220 pixels) using bilinear interpolation, inverting the pixel values (so that background pixels have value zero), and "normalizing" the images by dividing the pixel values by the standard deviation of all the image pixels in the dataset. The network architecture consists of two twin networks joined by a cost function, which computes a distance metric between the 128-D feature vectors of the two images in a pair. Since SigNet outputs a continuous numerical distance $D(s_1, s_2)$ that measures the dissimilarity between two signature images s_1 and s_2, one has to fix a threshold value τ to generate the binary classifier prediction. That is, if $D(s_1, s_2) \leq \tau$ the classifier predicts that the signatures are genuine (were performed by the same writer), otherwise, if $D(s_1, s_2) > \tau$, the classifier predicts that the pair (i.e., one of the signatures) is a forgery. For performance evaluation of SigNet, the authors compute metrics such as accuracy (which depends on the threshold τ) by selecting the maximum accuracy that is

obtained in the test set over all possible thresholds τ (DEY et al., 2017). According to their results, the proposed technique outperforms the state-of-the-art in datasets with a sufficient number of signatures for training, but its accuracy is negatively affected when signatures vary too much in style and the number of signature samples is low. The authors also perform an experiment training only with unskilled forgeries, demonstrating that the results are worse than training only with skilled forgeries.

The accuracy for SigNet is listed as 100% on the CEDAR dataset, 86.11% on the Bangla dataset and 84.64% on the Hindi dataset (DEY et al., 2017). As shown by the experiments performed in the present work, the 100%-accuracy number is likely caused by an inherent bias in the CEDAR dataset (Section 4.3). Also, the author of the present work was able to recompute the results of Dey et al. (2017), using their source code but removing the dataset bias (see Table 4.3).

Dey et al. (2017) also perform cross-dataset experiments, i.e. training with signatures from one dataset, but testing over another dataset. According to their results, in general, their proposed method does not provide good generalization in the cross-dataset experiments (accuracies dropped dramatically ($> 15\%$) in all the cases except for training on GPDS300 dataset and testing on CEDAR with 94.82% of accuracy). The other cross-dataset validations obtain accuracies ranging from 50% to 69%.

The work by Dutta, Pal and Lladós (2016) also proposes a top-performing writer-independent method for offline signature verification, but this time based only on hand-crafted local features and global statistics. The feature extraction process can be summarized by the following steps:

- Get BRISK feature points (corner detector and scale invariant, so, theoretically this feature is independent of the signature size).
- Delaunay Triangulation is applied over BRISK feature points, connecting two feature points if and only if they are connected by and edge in the triangulation. Their hypothesis is that for genuine signatures most of the edges from the triangulation remain stable, while for forged signatures they could suffer distortions.
- Additionally, histogram of oriented gradient (HOG) descriptors are computed for each BRISK feature point (local descriptors). For points connected by an edge in the triangulation, their HOG descriptors are concatenated (called pairwise descriptors).
- These descriptors go through an encoding process, followed by the computation of weighted-histograms that define global image statistics (weights are defined based on the areas of signature features, from a “water reservoir model”).

- Finally, a kernel function that measures the similarity between two images is defined, and an SVM is trained to perform classification. The SVM hyperparameters are optimized for each cross-validation fold based on the maximum accuracy obtained over a range of possible parameters (DUTTA; PAL; LLADÓS, 2016).

The authors claim to achieve 100% accuracy for the CEDAR dataset (which is likely caused by an inherent bias in the CEDAR dataset (Section 4.3)) while getting 88.79% for GPDS300 dataset.

Hamadene and Chibani (2016) present a one-class writer-independent method that involves the use of the Contourlet Transform for feature generation and the use of feature dissimilarity measures for automatic threshold selection. The main contribution of the proposed method is to require just a few genuine signatures for training. Dissimilarity thresholding is based on the concept of signature stability, considering a signature pair as more stable when their feature dissimilarity is lower. For the feature generation process the Contourlet Transform is obtained by decomposing the image signature at only one resolution level and four main directions, which leads to a feature vector of 16 components. The feature dissimilarity is the Canberra distance because of its relative efficiency in comparison to the Euclidean Distance. Additionally, threshold selection is performed by computing feature dissimilarities between each possible genuine-genuine signature pair per writer. Feature dissimilarities for each possible combination per writer are ordered in a increasing order of dissimilarity value, being then the first and last element set the minimum and maximum dissimilarity value. The decision threshold is selected from this ordered set based on a stability parameter defined as the Half Total Error Rate which is basically the average of FRR and FAR. For the experiments, the authors used CEDAR and GPDS datasets, using 10 writers per dataset (20 writers in total) to define a writer-independent decision threshold. According to their results, using more reference signatures improves the performance, obtaining 2.10% of AER for CEDAR, 18.23% for GPDS and 16.8% for a combined mixed test dataset between CEDAR (45 writers) and GPDS (462 writers).

Some other tests were performed in order to determine if higher resolutions of the Contourlet Transform have a possitive impact on the error rates with no successful results. Additionally, an analysis of the impact of the number of training writers was made, demonstrating that the AER appears to be invariant to the number of training writers. Finally, a cross-dataset validation was performed, training with only one dataset and testing on another, considering the mixed dataset mentioned in the previous paragraph as

an independent dataset. The results show that the average error rate remains almost the same than when mixed training writers were used, also that the unbalancing between FAR and FRR metrics virtually disappear when training on and testing on the mixed dataset. This demonstrates the importance of merging different datasets for writer-independent approaches. Comparing the results of the proposed method over the GPDS dataset with other writer-independent methods shows that it did not achieve comparatively better results at all, but it still has the benefit of using only a few genuine signatures for training.

In summary, offline signature verification is the least invasive approach for the signature verification problem, although offline methods achieve worse results than the online approaches. Additionally, writer-independent approaches have the advantage of training one single model. Ideally, when trained with signatures from a wide range of handwriting styles, they do not have the necessity of retraining at all. Finally, as discussed in this section one can see different signature verification methods trying to find a good (handcrafted) feature descriptor based on the signature shape or graphometric characteristics such as density, slant, or distribution (BERTOLINI et al., 2010). Other works include global statistics for the signature images, stroke signature and curvature. The problem is: it is extremely difficult to determine the set of graphometric characteristics that can reliably describe the handwritten signature style for all persons. In this sense, the use of CNNs to extract relevant features automatically seems to be the best option and proves to obtain good results (DEY et al., 2017). However, apparently such approaches are not sufficient enough to deal with dataset transformations such as rotation and scaling (see Chapter 4). As such, we propose the use of hybrid features: hancrafted in addition to transferred from a CNN. This is done in order to obtain features that better describe signatures (CNN features) and provide robustness against rotation and scaling operations (handcrafted features).

3 PROPOSED METHOD: VERSIG-R

VerSig-R approaches the signature verification problem in a writer-independent manner using offline information. Since **VerSig-R** is writer-independent, for training and testing phases it receives as input a signature pair, which must be classified as either *genuine* (if both signatures belong to the same writer), or as a *forgery* (if the signatures belong to different writers). In Figure 3.1 a complete overview of **VerSig-R** is presented, showing step by step the feature generation process and the operations performed over the initial raw features to end up with a high dimensional vector, which combines information from both signatures in the signature pair, and is used for the SVM classifier to perform the actual classification task.

As one can observe in Figure 3.1, the signature verification is approached as a binary classification problem (*genuine*, *forgery*), where a high-dimensional feature vector is generated from signature pairs and then used as input to an SVM classifier (which is widely used for binary classification problems, since it classifies information by finding the hyperplane that maximizes class data separation, i.e., the hyperplane with the largest margin between two classes (BOSER; GUYON; VAPNIK, 1992; CHAPELLE; HAFFNER; VAPNIK, 1999)). A set of hybrid features is proposed, which is composed of: handcrafted features obtained by a novel Moving Least-Squares (MLS) strategy (Section 3.1), in addition to features obtained from a pretrained convolutional neural network (CNN) (Section 3.2). It is worth to notice that all the features are calculated in an automated way, with no parameters requiring manual tuning. Additionally, the SVM classifier is trained in a supervised manner (Section 3.3), using default hyperparameter settings.

In the following sections a more detailed description about feature generation process (MLS and CNN features) is presented.

3.1 Moving Least-Squares (MLS) Feature Generation

The proposed handcrafted features are aimed at extracting geometrical and topological characteristics of the signatures, which are good discriminators for signature verification (KALERA; SRIHARI; XU, 2004). Additionally, the secondary goal is to achieve this in a way that is simple and robust. The handcrafted feature generation operates on the idea of quantifying the occurence of pronounced curvatures and stroke-intersections in the signatures, based on the intuitive observation that forgers would need to mimic these

exact characteristics in order to obtain a convincing forged signature. Furthermore, while straight or nearly straight parts of the signature may be easier for forgers to imitate, the exact proportion between curved and straight regions is more difficult to imitate.

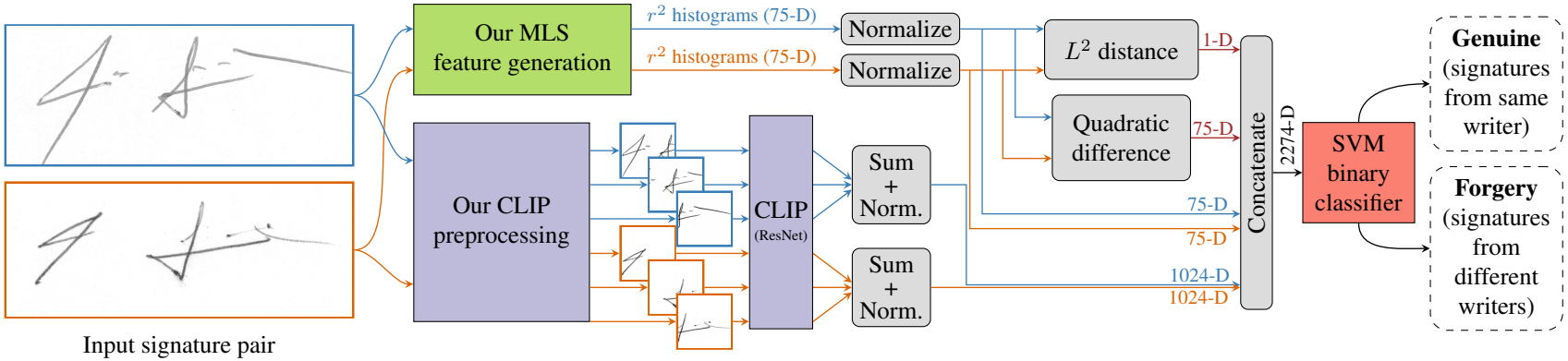
In the following subsections an overview of the variant of the MLS algorithm proposed is presented as well as a step by step description of the algorithm. Additionally, the correlation between MLS algorithm and the r^2 value is discussed in order to finally present a discussion on this handcrafted feature visualization.

3.1.1 MLS Overview

VerSig-R is aimed to quantify the curvature of any part of the signature's stroke by evaluating how well it fits a straight line. More precisely, **VerSig-R** fits a weighted least-squares line to each small neighborhood along the signature's stroke, and evaluate its goodness-of-fit using the r^2 (r-squared) coefficient (DODGE, 2008). Since the signature's stroke locations with high curvature or stroke-intersections are not well described by a straight line, they will be associated with small r^2 values (Figure 3.2c). Furthermore, by computing histograms of all r^2 values obtained along the signature, one is able to describe the distribution between curved and straight regions of the signature (Figure 3.2d).

In general, r^2 values are calculated from the signature's skeleton in small neighbourhoods and then grouped in 5 histograms with a different number of bins, producing the same quantity of histograms (vectors) with their corresponding dimensionality, to finally concatenate them obtaining a final 75-D vector for each image in the signature pair. Operations over this two final vectors like L^2 and quadratic difference are performed to end up with a final 226-D vector which contains information from both signatures in the signature pair.

Figure 3.1: Overview of our offline writer-independent technique for signature verification. Given a pair of signature images (left), our method generates features based on a Moving Least-Squares strategy (Section 3.1), in addition to CNN-transferred features (Section 3.2). The resulting 2274-D feature vector for the pair is fed to a binary SVM classifier, which distinguishes between genuine and forged signature pairs.



Source: Signature images from CEDAR dataset (KALERA; SRIHARI; XU, 2004), diagram created by the author.

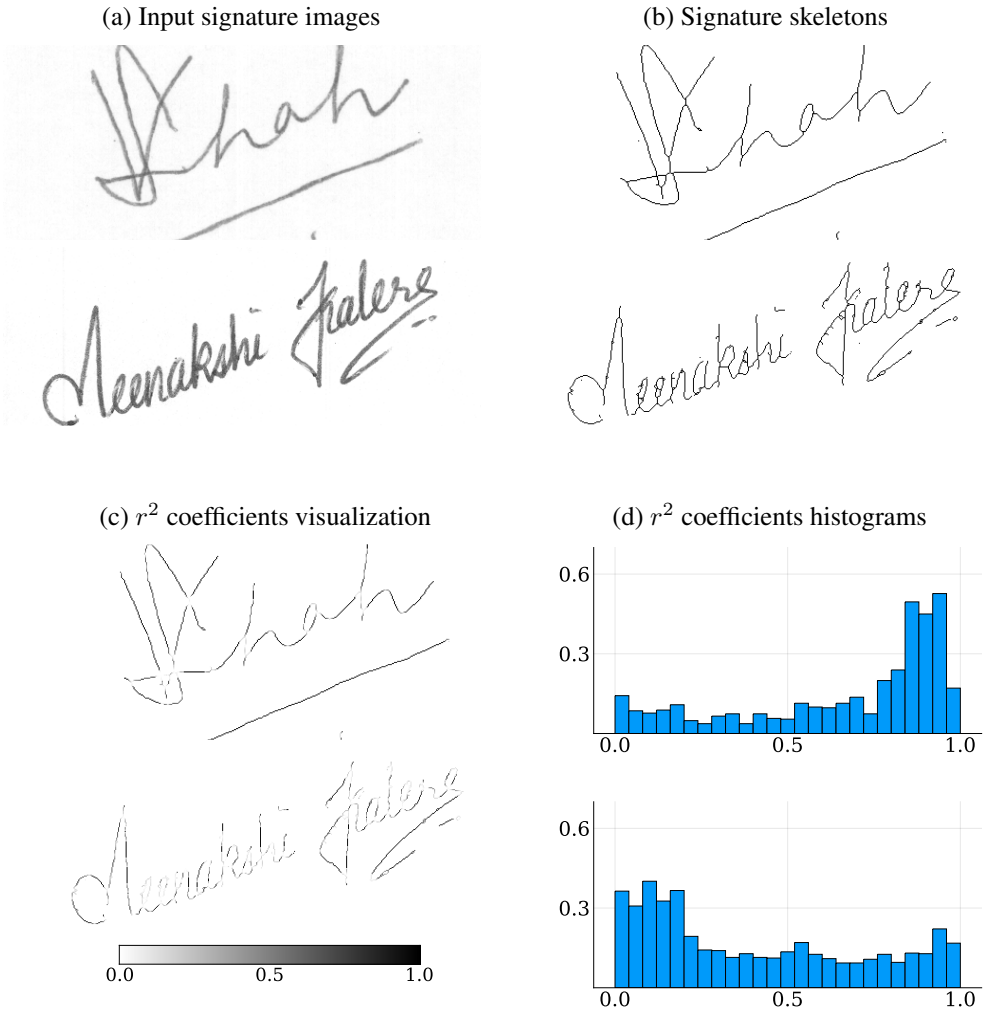
3.1.2 MLS Detailed Algorithm

For each signature image, **VerSig-R** uses the following algorithm to compute a 75-D feature vector based on the idea of r^2 histograms described above:

1. Binarize the input signature image. **VerSig-R** uses Otsu's algorithm (OTSU, 1979), which determines an optimal global binarization threshold from the image's histogram;
2. Compute the signature's skeleton from the binarized image (Figure 3.2b). **VerSig-R** uses the thinning algorithm of Zhang and Suen (1984);
3. For each $L \times L$ pixel neighborhood Ω_p centered at each point p in the signature's skeleton, **VerSig-R** fits a straight line through all skeleton points existing in Ω_p, using a Moving Least-Squares (MLS) strategy (described below in Section 3.1.3). Additionally, **VerSig-R** discards all neighborhoods which contain less than five skeleton points, with $L = 11$ pixels fixed in our implementation (according to the experiments performed by the author, the final result is not sensitive to the exact value of L, likely due to the weighting performed by **VerSig-R** in the least-squares fits);
4. For each neighborhood Ω_p, its associated r^2 goodness-of-fit coefficient is computed from the weighted least-squares residuals (Eq. (3.3), below);
5. Finally, all r^2 values are collected to immediately compute their distributions at several scales, using histograms with 5, 10, 15, 20 and 25 bins (uniformly distributed in the $[0, 1]$ interval, since $r^2 \in [0, 1]$). Figure 3.2d shows the 25-bin version of the r^2 histograms. The author determined experimentally that considering a concatenation of all the histograms achieves better results than considering them individually or considering any subset of them. By concatenating all these histograms, a 75-D feature vector is obtained for the signature image.

Given a signature pair, as shown in Figure 3.1(left), **VerSig-R** computes the 75-D r^2 histograms descriptors (feature vectors) for each image in the pair. These feature vectors are then normalized by dividing them by their L^2 norm (this makes the histograms invariant to the absolute number of pixels in the signature image). Furthermore, **VerSig-R** computes the L^2 distance between the normalized feature vectors $\vec{u}$ and $\vec{v}$ of both images (resulting in a positive scalar $d = \|\vec{u} - \vec{v}\|_{L^2}$), in addition to their quadratic difference (resulting in another 75-D vector $\vec{w}$, whose i-th component is $\vec{w}_i = (\vec{u}_i - \vec{v}_i)^2$). Concatenating all these

Figure 3.2: Illustration of our Moving Least-Squares (MLS) feature generation pipeline, for two example signatures from CEDAR. (a) An input signature is converted to a (b) binary skeleton, followed by (c) MLS goodness-of-fit computation (r^2 coefficients), which detects curved and stroke-intersection signature regions. The resulting (d) r^2 histograms describe the distribution of curved and straight regions of the signature.



Source: (a) Signature images from CEDAR dataset (KALERA; SRIHARI; XU, 2004). (b-d) The author.

quantities as $[\vec{u}, \vec{v}, \vec{w}, d]$ results in the final 226-D handcrafted feature descriptor for the signature pair. This will later be concatenated with the features generated by the neural network backend (Section 3.2).

3.1.3 Moving Least-Squares Fit and r^2

The idea of performing a sequence of least-squares line fits with a "sliding window" is related to the concept of Moving Least-Squares (LANCASTER; SALKAUSKAS, 1981), a technique widely used to reconstruct continuous functions from scattered data (LEVIN, 1998). Of particular importance for **VerSig-R** is the use of weights in the least-squares functionals, which are normally inversely proportional to each data point's distance from the center of the window/neighborhood. This idea was adapted for **VerSig-R** by also considering pixel intensity in the weighting.

Given a particular neighborhood Ω_p and a collection of points $(x_i, y_i) \in \Omega_p \cap \mathcal{S}$ that are also part of the signature's skeleton $\mathcal{S}$ (Figure 3.2b), **VerSig-R** fits a straight line $y = ax + b$ through the points by minimizing the weighted least-squares functional:

$$\mathcal{E}(a, b) = \sum_i w_i^2 \, (y_i - ax_i - b)^2. \tag{3.1}$$

The weights w_i are inversely proportional to the L^2 distance between the data points (x_i, y_i) and the point $p = (x_p, y_p)$ at the center of the neighborhood Ω_p. That is, pixels that are closer to the center of the neighborhood are given more importance (*greater* weights) in the line fit. The weights are also inversely proportional to the intensities $f(x_i, y_i)$ of the pixels. Thus, darker pixels also receive greater weights, as they are more likely to be an important part of the signature's stroke:

$$w_i = \frac{1}{1 + f(x_i, y_i) \, \|(x_i, y_i) - (x_p, y_p)\|_{L^2}}. \tag{3.2}$$

The optimal line parameters $a^\star$ and $b^\star$ that minimize $\mathcal{E}$ are the ones where $\partial\mathcal{E}(a^\star, b^\star)/\partial a = \partial\mathcal{E}(a^\star, b^\star)/\partial b = 0$. One can then compute the goodness-of-fit r^2 value as:

$$r^2 = 1 - \frac{\mathcal{E}(a^\star, b^\star)}{\sum_i w_i^2 (y_i - \bar{y})^2}, \quad \text{where} \quad \bar{y} = \frac{\sum_i w_i y_i}{\sum_i w_i}. \tag{3.3}$$

For lines with slope $|a^\star| > 1$, **VerSig-R** instead fits x as a function of y, with $x = my + c$ (this is done by simply swapping x and y in the equations above). This avoids

numerical problems for lines that are close to vertical (where the slope $|a^\star| \to \infty$).

3.2 CLIP Feature Generation

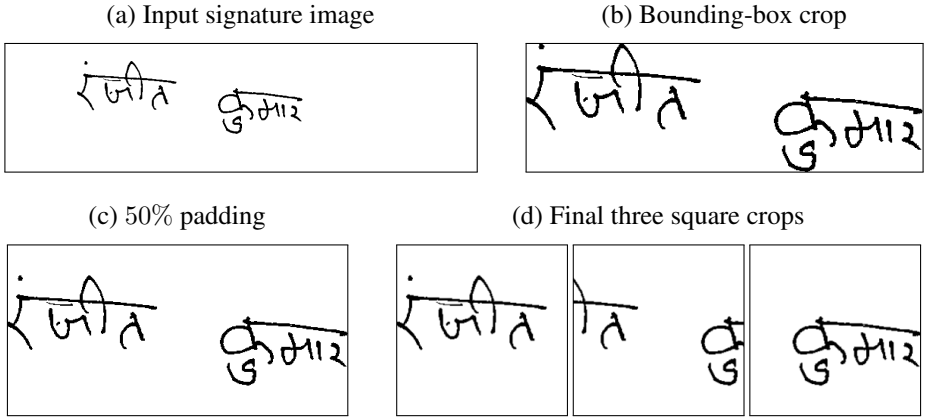
Convolutional Neural Networks (CNNs) are able to automatically learn features for distinguishing between images in classification tasks (LECUN; BENGIO; HINTON, 2015). The discriminative power of CNN features also generalizes well between different problem domains, which is the idea behind *transfer learning* (ZHUANG et al., 2021). This technique allows one to use features learned in a domain containing large amounts of training data, in another domain where training data may be scarcer or harder to obtain.

Based on the transfer learning concept and looking for convolutional neural networks that could generalize well for the signature-verification problem being approched, a set of experiments were performed using different publicly available pretrained Convolutional Neural Networks such as: MobileNetv2 (SANDLER et al., 2018), DenseNet (IANDOLA et al., 2014), ResNet101 (HE et al., 2016), Resnext101 (XIE et al., 2017) and VGG16 (SIMONYAN; ZISSERMAN, 2014). The mentioned CNNs were used (without modifications neither the necessity for retraining) to generate features for each signature image in each signature pair, in order to finally use these features for the classification task. The methodology used for the evaluation of all these models is described in Section 4.2 and the SVM classifier used is described in Section 3.3. None of the just mentioned models exceeded 70% of accuracy on the CEDAR dataset.

Additionally, another set of experiments were performed with the CNN called CLIP (Contrastive Language-Image Pre-training) (RADFORD et al., 2021) using both backends ViT-B and ResNet-50. Both backends perform similarly, but the ResNet-50 backend managed to get better results. The initial tests were performed over CEDAR dataset getting performance slightly over 97% of accuracy. In Table 4.4 one can see averaged (over CEDAR, Bangla and Hindi datasets) results for the unbiased and the other versions of the datasets. Based on the set of experiments performed by the author, one can conclude that the recent CLIP neural network (using the ResNet-50 backend) produces features which generalize well to the signature-verification problem being approached. This Convolutional Neural Network was trained on 400 million images with associated textual captions, with the goal of predicting which caption belongs to which image.

However, giving the “raw” signature image as input to CLIP CNN is not optimal due to variations in the signatures’ aspect ratios (most signatures have a rectangular shape,

Figure 3.3: Proposed CLIP preprocessing pipeline (Section 3.2).



Source: (a) Bangla dataset (PAL et al., 2016). Images (b-d) Generated by the author based on initial input image (a).

which is not taken into consideration by the original CLIP preprocessing pipeline, which simply considers squared central part of the image). Furthermore, signatures are scanned with varying degrees of padding (empty paper) around the signature (specially in the Hindi and Bangla datasets). We propose the following preprocessing pipeline for CLIP CNN, illustrated in Figure 3.3, which significantly improves classification accuracy (Section 4.5). For each signature image:

1. Crop the image to the bounding-box of the signature. **VerSig-R** does this by removing pixel rows and columns (around the border of the image) whose average pixel values are below the threshold 254 (considering 8-bit encoded grayscale, with pixel values in $[0, 255]$);
2. Pad the image with empty rows/columns (i.e., with white pixels), to guarantee that the smallest dimension is no smaller than 50% the size of the largest dimension. This is done in order to guarantee that the crops generated in the next step (Step 3) cover the whole signature, while also overlapping by $\geq 50\%$;
3. Generate three square crops from the signature image (since CLIP CNN expects square images). If the image has width $>$ height, crops are extracted aligned to the left, center, and right of the image. Otherwise, if width $\leq$ height, the crops are extracted from the top, center, and bottom of the image.

At the end of this preprocessing step, features are generated for each of the crops, using the pretrained CLIP network. It is worth noticing that the number of crops generated was determined empirically during the implementation phase. Additionally, each square crop is resized to 224×224 pixels (operation performed to fulfill CLIP CNN requirements)

and fed to CLIP CNN, resulting in three 1024-D feature vectors (one for each crop). In order to obtain a single feature vector per image, the best way to join the mentioned three vectors was determined empirically. Experiments were performed using simple operations like elementwise sum, mean and max operations, as well as concatenation, all operations followed by a renormalization. The concatenation option significantly increases memory consumption and execution time since the number of CNN features will be 3 times that of the other options. Finally, the experimental results demonstrate that performing three vectors elementwise sum, followed by L^2 normalization, generates the best results generating a single 1024-D feature vector that describes the whole signature.

3.3 Classification with SVM

For each signature pair, both MLS handcrafted features (Section 3.1) and CLIP features (Section 3.2) are concatenated, resulting in a single 2274-D feature vector for each single signature pair (Figure 3.1, right). This feature vector is used as input information to a binary classifier, the two classes involved could be described as: 1) a signature pair composed by two signature images belonging to the same writer (positive class) and 2) a signature pair composed by two signatures images belonging to different writers (negative class).

VerSig-R uses a linear Support Vector Machine (SVM) model to perform the binary classification task, which performs better than other machine-learning classification models for this particular problem according to experiments performed by the author. The implementation of **VerSig-R** in terms of the SVM classifier is based on the `LinearSVC` class of scikit-learn (PEDREGOSA et al., 2011) framework, with default hyperparameters.[1] The training phase for the SVM classifier was performed in a supervised manner, using a training set composed of example signature pairs (combined from many different writers, additionally signature pairs examples from writers used during the training phase were excluded from the testing set) with known genuine/forgery signature pairs labels. Finally, since **VerSig-R** is using a writer-independent approach, it trains a single model for all writers.

[1] The SVM classifier was set to optimize for the primal problem, which is the recommended procedure when the number of training examples is greater than the number of features.

4 EXPERIMENTAL RESULTS

Exhaustive experiments were performed with **VerSig-R** over the CEDAR dataset (KALERA; SRIHARI; XU, 2004) and the Hindi and Bangla datasets composing the BHSig260 dataset (PAL et al., 2016). Experiments were performed over the original versions of the three just mentioned datasets (i.e. images from the datasets without any kind of modifications) as well as their unbiased versions, a more detailed discussion on this topic can be found in Section 4.3. Additionally, in order to analyze the effect of rotations and scalings over the classification performance, three other versions (built on top of the unbiased versions) of the datasets were generated (rotations, scalings, and a combination of both transformations) and were subjected to the same set of experiments (Section 4.4). **VerSig-R** was compared against the best-performing state-of-the-art offline writer-independent methods found in the literature: Dutta, Pal and Lladós (2016), which is based on handcrafted features, and Dey et al. (2017), which is based on a neural network called SigNet. In both cases, the source code provided by the corresponding authors were used to perform experiments. Table 4.3 summarizes the results encountered, which are discussed in a detailed manner in the following subsections.

Additionally, a set of experiments were performed in order to study the effect of the number of reference signature samples used during the training phase over the classification performance. Experiments were performed with 12, 5, 4, 3, and 2 reference samples from positive as well as negative classes. Reducing the number of references samples during the training phase intuitively should have a negative impact over the classification performance of any model, but in the particular case of the signature verification problem, a reduced number of required references samples is desirable because of the lack of reference samples in real-life scenarios. A discussion of this topic and the results encountered in the experiments is included in Section 4.7.

In addition to the experiments mentioned in the previous paragraphs, single experiments were performed over the original versions of MCYT (ORTEGA-GARCIA et al., 2003; FIERREZ-AGUILAR et al., 2004) and GPDS (FERRER; DIAZ-CABRERA; MORALES, 2015; FERRER; DIAZ-CABRERA; MORALES, 2013) datasets. Finally, a group of experiments for cross-dataset validation was performed, i.e. training **VerSig-R** over one dataset and using the trained classifier to test over another dataset.

The remainder of this section is organized as follows: Section 4.1 shows a visual analysis of histograms generated as part of the handcrafted feature generation process, in

order to identify the importance of these features. Section 4.2 introduces the methodology used for the experiments, Section 4.3 discusses and explains details about bias encountered during experiments (mainly in the CEDAR dataset), Section 4.4 presents the analysis of the impact of rotation and/or scalings over the classification performance, Section 4.5 presents an ablation study from using only original CLIP neural network features, the impact of the proposed variations to the CLIP preprocessing pipeline and the final features generated by the method proposed in the present work. Additionally, in Section 4.6 a discussion of the experiments performed over GPDS and MCYT datasets is presented, while also introducing the use of PCA technique for dimensionality reduction of the feature vectors. Section 4.7 evaluates the impact of the number of reference samples used during the training phase while in Section 4.8 the impact of using only random forgeries during training is evaluated. In Section 4.9 a study on the generalization power of **VerSig-R** is presented, i.e., how well a classifier trained on one dataset generalizes when tested over another dataset while in Section 4.10 a brief study of the execution time of the cross validation applied to the datasets being studied in this work is presented. Finally, in Section 4.11 a user study is presented, comparing accuracy results from **VerSig-R** against results for humans performing the classification task.

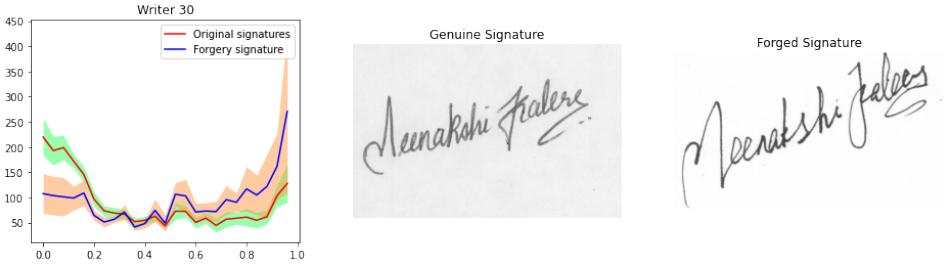
4.1 r^2 Histograms Analysis

The left-hand-size columns of Figures 4.1, 4.2, and 4.3 show mean r^2 histograms computed separately for genuine and forgery signatures samples from a particular writer (the solid blue line with orange shaded area corresponds to forged signatures, and the solid red line with green shaded area corresponds to genuine signatures). Additionally, mean r^2 histograms are represented by the solid lines, and their corresponding three-standard deviations intervals are represented by the corresponding shaded areas. Histograms of 25 bins were computed for each signature image as detailed in Section 3.1.2 and then averaged separately for genuine and forgery references respectively to generate mean r^2 histograms. It is worth noticing that the mean r^2 histograms shown in Figures 4.1 to 4.3 are used only for visual analysis purposes, and their computation is not included in the **VerSig-R** pipeline.

Figures 4.1, 4.2, and 4.3 presents mean histograms for writer 30 (chosen arbitrarily) from the CEDAR, Bangla and Hindi datasets respectively. Additionally, each of the just mentioned figures contains reference genuine and forged signature examples of writer

Figure 4.1: Illustration of mean r^2 histograms generated separately for original and forgery signature references belonging to Writer 30 of the CEDAR dataset. Mean r^2 histograms were generated using 25 bins.

(a) CEDAR BIASED



(b) CEDAR UNBIASED

(c) CEDAR UNBIASED WITH ROTATIONS

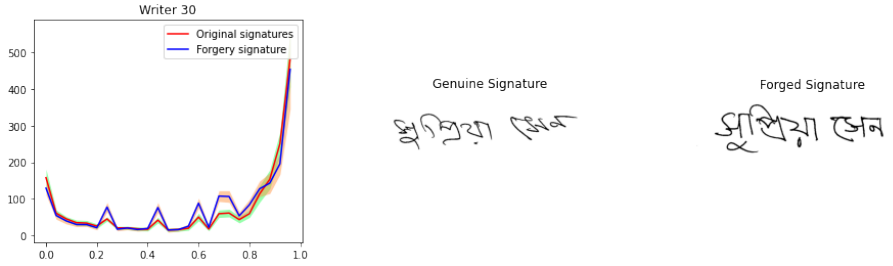
(d) CEDAR UNBIASED WITH SCALES

(e) CEDAR UNBIASED WITH ROTATIONS AND SCALES

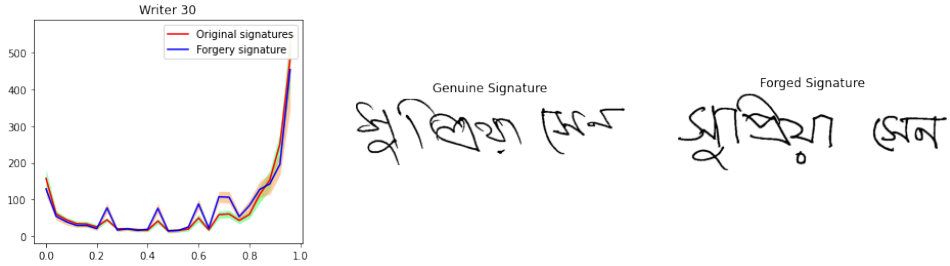
Source: (a) Signature images from CEDAR dataset (KALERA; SRIHARI; XU, 2004). (b-e) Generated by the author based on (a).

Figure 4.2: Illustration of mean r^2 histograms generated separately for original and forgery signature references belonging to Writer 30 of the Bangla dataset. Mean r^2 histograms were generated using 25 bins.

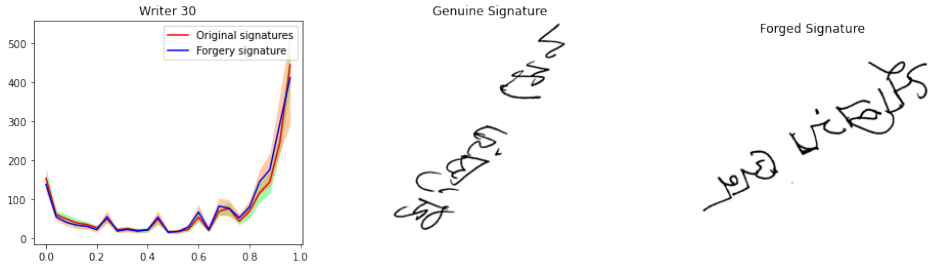
(a) Bangla BIASED



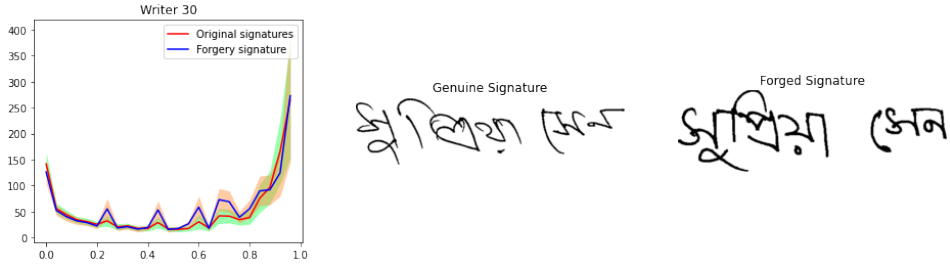
(b) Bangla UNBIASED



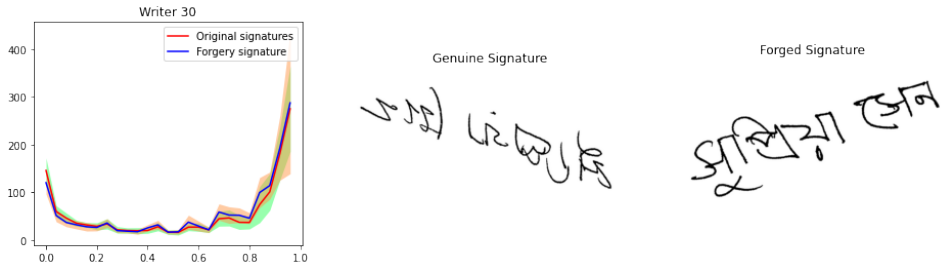
(c) Bangla UNBIASED WITH ROTATIONS



(d) Bangla UNBIASED WITH SCALES



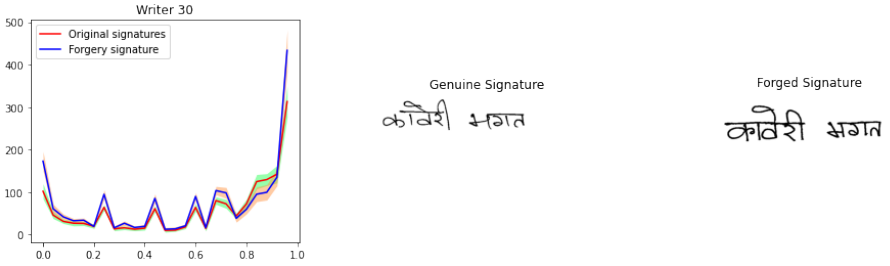
(e) Bangla UNBIASED WITH ROTATIONS AND SCALES



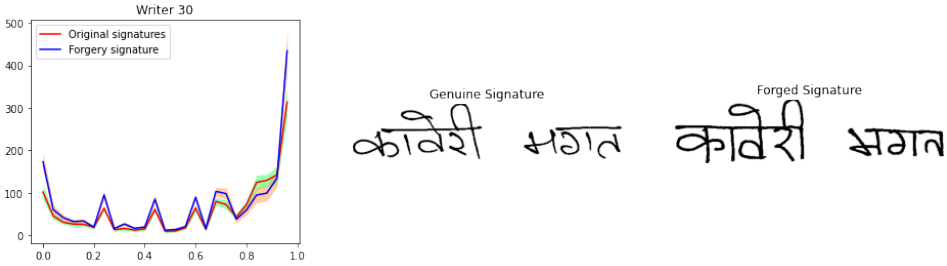
Source: (a) Signature images from Bangla dataset (PAL et al., 2016). (b-e) Generated by the author based on (a).

Figure 4.3: Illustration of mean r^2 histograms generated separately for original and forgery signature references belonging to Writer 30 of the Hindi dataset. Mean r^2 histograms were generated using 25 bins.

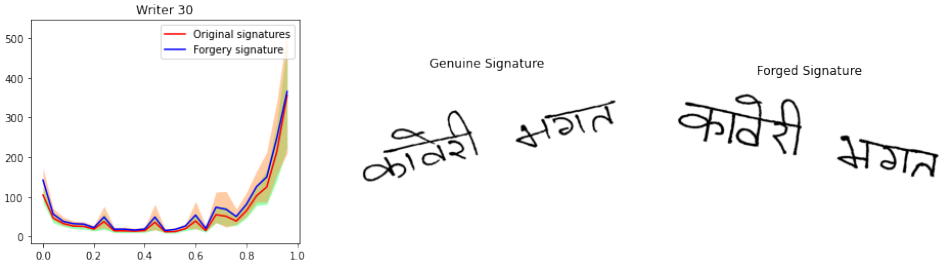
(a) Hindi BIASED



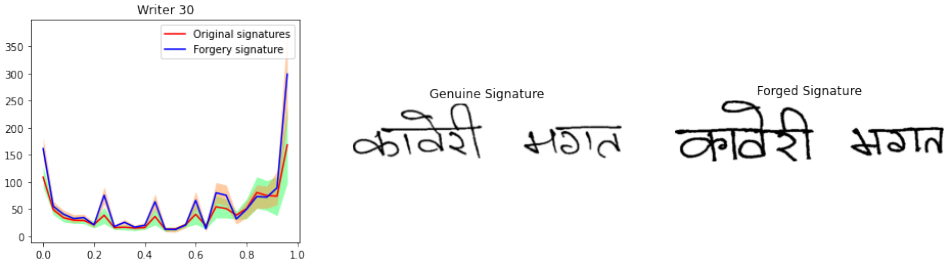
(b) Hindi UNBIASED



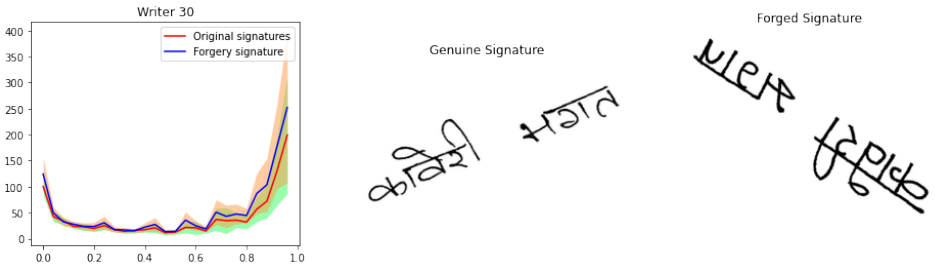
(c) Hindi UNBIASED WITH ROTATIONS



(d) Hindi UNBIASED WITH SCALES



(e) Hindi UNBIASED WITH ROTATIONS AND SCALES



Source: (a) Signature images from Hindi dataset (PAL et al., 2016). (b-e) Generated by the author based on (a).

30 for each of the dataset versions (biased, unbiased, with rotations, with scales, and the combination of both transformations, as discussed in Sections 4.3 and 4.4) for visual reference.

It is worth to notice that in all cases r^2 histograms for genuine and forged signatures differ from each other in one or more segments of the mean histograms, although the difference sometimes falls inside the three-standard deviations interval. Nonetheless, this fact indicates that the aforementioned MLS handcrafted feature is a good discriminator for this particular classification problem. Additionally, one can notice that histograms (genuine and forged) for the CEDAR dataset, have a totally different structure if one compares them to the histograms corresponding to Bangla and Hindi datasets, and also that Bangla and Hindi datasets have nearly the same mean histogram structure. This is perfectly reasonable if one considers that the CEDAR dataset is composed by Western-style signatures, while Bangla and Hindi corresponds to South-Asian styles signatures.

In particular, if one compares mean histograms in Figures 4.1a and 4.1b, which correspond to the original version of the CEDAR dataset and its unbiased version respectively, one can see that the mentioned histograms are nearly the same. Since the proposed MLS hancrafted feature generation technique is based only on the signature skeleton, this process is not affected (or is affected minimally) by the signature image background. This is particularly useful for real-life scenarios, when the digitalization process could introduced undesired backgrounds, and in contrast to other proposed techniques that use the whole image for feature extraction (Section 4.3), the final features are not affected by the just mentioned backgrounds.

Additionally, if one compares the mean histograms from the unbiased version of the datasets and the version subjected to scales, in Figures 4.1, 4.2, and 4.3, one can notice that the histograms have nearly the same structure but the standard deviation suffers an increment in the scaled version with respect to the unbiased version. Also, both histograms are moved down vertically, i.e., have a smaller count of values for each bin. This fact is easily explained due to the fact that when the image is downscaled, some information needs to be discarded or aggregated, so, the signature skeleton is composed now by a smaller amount of pixels, and this also introduces a larger standard deviation (because of the lack of skeleton points to represent it accurately). This is why in the proposed approach the histograms are normalized by their L^2 norms, making them invariant to the absolute number of pixels in the signature image. Since the difference between the mean histograms for the unbiased and scaled versions of the datasets are minimal, one can say

that the proposed MLS hancrafted feature generation technique provides scale invariant properties to the global feature generation process.

In the same direction, comparing mean histograms from the unbiased versions of the datasets and the version subjected to rotations in Figures 4.1, 4.2, and 4.3, one can notice that the rotation transformation reduces the gap between the mean histograms for genuine and forged signatures, and also slightly increments the standard deviations for both histograms. In other words, it makes it more difficult to discriminate between the positive and negative classes (considering only the proposed MLS hancrafted feature generation technique). This negative impact together with the increment of the standard deviation introduced by the scaling of a signature image makes the rotated and scaled versions of the datasets the more challenging case for this particular classification problem. The inferences just mentioned, are validated by the ablation study performed for **VerSig-R** (see Table 4.4)

One interesting fact to notice is that the CEDAR dataset presents, for all of its versions, clearly separated mean histograms for genuine and forged signatures. This fact means that genuine and forgery classes are easier to discriminate. Also, the standard deviation for genuine and forged classes is greater in the CEDAR dataset than the Bangla or Hindi datasets, specially in the forgery class. Despite the fact that the forgery class for the CEDAR dataset has a significative standard deviation (significant intra-class variability), from Figure 4.1 one can notice that the greater interclass variability for this dataset allows an easier classification process than its counterparts Bangla and Hindi. In simple terms, forgeries for the CEDAR dataset are not so accurate and additionally are different between them, while in the case of Bangla and Hindi datasets, the forgeries are more accurate than CEDAR dataset, making the classification over these datasets a more challenging problem.

4.2 Experimental Methodology

In order to perform a fair comparison between the three methods being evaluated, it is necessary to implement a methodology or framework in such a way that it could be applied in the same way for all signature datasets and classification methods. The hardware specifications used to perform all the experiments shown here was the following: RTX 2080 Ti GPU, Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz (16 threads). In the subsequent paragraphs, the methodology used for all the experiments performed in the present work is explained in detail.

Each dataset is composed of a collection of writers, each writer having a certain

Table 4.1: Statistics of the datasets used in the experiments performed in the present work.

		(per writer)				
Dataset	Writers	Genuine (G)	Forgery (F)	G-G	G-F	G-G + G-F (Balanced)
CEDAR	55	24	24	276	576	552
Bangla	100	24	30	276	720	552
Hindi	160	24	30	276	720	552
GPDS	4000	24	30	276	720	552
MCYT	75	15	15	105	225	210

number of genuine signature images (G) and *skilled-forged* signature images (F). By pairing signatures from the same writer, one can generate examples of genuine-genuine signature pairs (G-G), and genuine-forgery signature pairs (G-F). Table 4.1 summarizes these numbers for the CEDAR, Bangla, Hindi, MCYT and GPDS datasets, while Table 4.2 summarizes these numbers when using a a smaller number of references than is available (Section 4.7). It is worth to mention that, since the present work considers signature verification as binary classification problem, G-G and G-F signature pairs are treated as unique pairs, independently of the order of the signatures belonging to the pair, i.e. the pairs X-Y and Y-X, for specific reference samples X and Y, are considered equivalent.

Additionally, the methodology includes dataset balancing because of the binary classification model proposed, otherwise the model could be dominated by the predominant class (the G-F class in this case). To get a balanced dataset, one must use the same quantity of G-G and G-F pairs during training. Since the number of unique combinations of G-G pairs is always less than unique G-F pairs for any dataset with the same or greater number of forgery (F) reference samples than genuine (G) reference samples for each writer (which is the case for all the datasets considered in this work), a balanced dataset is obtained by randomly sampling count(G-G) pairs from all possible G-F pairs. For example, in the case of the CEDAR dataset, all the 276 G-G pairs and 276 randomly choosen G-F pairs *per writer* will be used to constitute a balanced dataset. Although Bangla, Hindi and GPDS datasets have more G-F pairs than the CEDAR dataset, they finally will get the same number of signature pairs per writer when generating the corresponding balanced dataset, because they have the same quantity of G-G pairs than the CEDAR dataset. Finally, in the case of MCYT dataset, only 105 G-G and G-F pairs will be used totaling 210 signature pairs per writer in a balanced dataset.

For all methods being evaluated, *10-fold* cross validation was performed. Data splitting into training and test sets for each fold was performed in a *by writer* manner, i.e. signature pairs belonging to the same writer must be either all on the training set or all on

Table 4.2: Number of reference samples per writer based on number of genuine reference samples for the experiments in Section 4.7.

Genuine (G)	Forgery (F)	G-G	G-F	G-G + G-F (Balanced)
24	24	276	576	552
12	12	66	144	132
5	5	10	25	20
4	4	6	16	12
3	3	3	9	6
2	2	1	4	2

the test set, ensuring this way that test data is completely unknown by the SVM classifier at the training phase. Metrics are evaluated for each fold independently, but averaged across the 10 folds for simplicity, as one can see in the evaluation tables, mean and standard deviation for each metric evaluated are reported. The metrics considered for techniques comparison were: Accuracy, F1-score, precision, recall, ROC curve and the Equal Error Rate (EER). The EER metric is included, because it is widely used to compare signature verification techniques (MOHAMMED et al., 2015). EER is defined as the value at the intersection of the False Acceptance Rate (FAR) and False Rejection Rate (FRR) curves (which are generated by varying the discrimination threshold). Lower EER means better classification performance. According to Mohammed et al. (MOHAMMED et al., 2015), EER for offline signature verification systems ranges from 10% to 30% (versus 2% to 5% for online systems).

The source code of SigNet (DEY et al., 2017) does not implement cross validation for evaluation of their proposed technique. Because of that, the same cross validation mechanism described in the previous paragraph was implemented by the author. It is worth to mention that since the SigNet method is a Convolutional Neural Network, it separates a portion of the training data for validation, in order to always save the best obtained model state across all training epochs.

Additionally, it is worth to remind that, as mentioned in Chapter 2, the SigNet method selects the discrimination threshold that maximizes its accuracy *over the test set*, possibly using a different threshold in each different fold. In contrast, the proposed technique and Dutta et al.'s method use SVM classifiers with a fixed threshold defined by the particular implementation over the *training* set (normally the threshold used is 0). Therefore, the EER metric is the most suitable when comparing experimental results against the SigNet method, since EER is not affected by the test-set threshold selection of SigNet (the names of the metrics that are affected by this selection are marked with an

asterisk in the result tables of the present work, see Table 4.3).

In the case of the implementation of Dutta et al. (DUTTA; PAL; LLADÓS, 2016), it requires large amounts of memory for signature features computation, because of that, their source code provides a parameter which controls which percentage of the dataset is used for training and testing in each fold. In the experiments performed using this method, the default value of 30% (hard-coded in their source code) was used for CEDAR. In order to end up with similar number of examples in the test set for all datasets, percentages of 15% for Bangla and 10% for Hindi datasets were selected. The dataset percentages choosen lead to 828, 828 and 884 signature pairs for CEDAR, Bangla and Hindi respectively, for the testing set in each fold (considering all writers in the set).

In Table 4.3, the best results in each group are marked in bold, and the results which are statistically equivalent to the best result (according to Welch's *t*-test with $p < 0.05$) are highlighted in green.

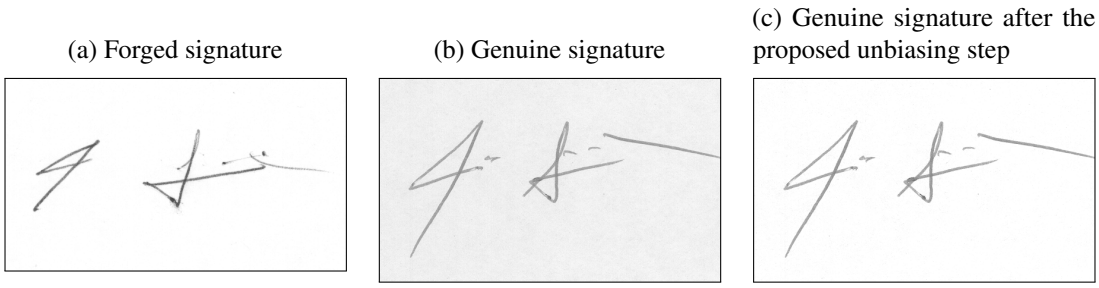
Additionally, a set of experiments was performed over the GPDS dataset (FERRER; DIAZ-CABRERA; MORALES, 2015; FERRER; DIAZ-CABRERA; MORALES, 2013) which contains information from 4000 synthetic individuals (Table 4.1), i.e. synthetically generated signature samples for 4000 writers according to the technique described by Ferrer et al. (FERRER; DIAZ-CABRERA; MORALES, 2015; FERRER; DIAZ-CABRERA; MORALES, 2013). This dataset is referred to as GPDS-4000. Additionally, one experiment was performed over the MCYT dataset (ORTEGA-GARCIA et al., 2003; FIERREZ-AGUILAR et al., 2004), the results obtained from this set of experiments are reported in Table 4.5.

Finally, a set of experiments for cross-dataset validation was performed. The idea behind this kind of experiments is to validate how well a classifier trained with one dataset generalizes when used to evaluate writers for another dataset, especially when trained and test over datasets with different type of handwriting. Associated results can be found in Tables 4.11, 4.12, 4.13 and 4.14.

4.3 Evaluation on "Unbiased" Datasets

When performing experiments over the CEDAR dataset, the author noticed that genuine signature images for all the writers in the dataset have an extremely different background when compared against their corresponding forged signature images (Figures 4.4a and 4.4b). More precisely, genuine signature images have light-gray background, in

Figure 4.4: Example signatures from CEDAR dataset, illustrating the significantly different background color between the (a) genuine and (b) forged signature images. This is a source of bias for Machine Learning algorithms.



Source: (a-b) CEDAR dataset (KALERA; SRIHARI; XU, 2004), (c) Image generated by the author based on image (b).

constrast with forged signature images with completely white backgrounds. This pattern is repeated for all writers in CEDAR, and is a significant source of bias for Machine Learning algorithms, which could "cheat" by differentiating between the two classes just based on the background color. In particular, the author hypothesizes that the 100%-accuracy listed in the SigNet paper for CEDAR is a consequence of this bias.

To prove the hypothesis introduced in the previous paragraph, a histogram transformation was applied to all genuine signatures in CEDAR in order to remove the background color discrepancy (i.e., make all images have a white background). This is illustrated in Figure 4.4c. The histogram transformation was applied using the following equation:

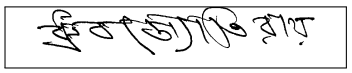
$$\text{new\_pixel\_color} = \text{clamp01}\left(1.23 * (\text{original\_pixel\_color} - 0.4) + 0.35\right), \qquad (4.1)$$

where the $\text{clamp01}(x)$ operator returns the value x clamped to the $[0, 1]$ range. From this "unbiasing" procedure the author defines a new version of the CEDAR dataset which was named C-Unbiased, and the original CEDAR dataset was renamed as C-Biased.

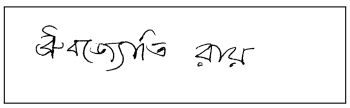
As shown in Table 4.3, the accuracy for SigNet drops from 100% in C-Biased to 79.51% in C-Unbiased, supporting the hypothesis that SigNet is heavily influenced just by the difference in the background color between the genuine and forged signature reference samples (i.e., the classifier was not differentiating between the signatures, just their backgrounds). The method of Dutta et al., on the other hand, is not corrupted by this bias, instead *increasing* its accuracy from 90.93% in C-Biased to 92.87% in C-Unbiased (likely the white-background images work better for their BRISK feature extraction). Finally, **VerSig-R** also includes CNN features and thus is sensitive to this type of bias, but proves to be significantly more robust (versus SigNet): its accuracy drops just 2.15

Figure 4.5: Example signatures from Bangla dataset, illustrating the different signature position between (a) genuine and (b) forged signature images.

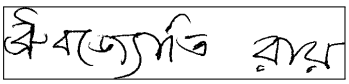
(a) Forged signature



(b) Genuine signature



(c) Genuine signature after the proposed unbiasing step



(a-b) Bangla dataset (PAL et al., 2016). (c) Image generated by the author based on Image (b).

percentage points, from 99.57% in C-Biased to 97.42% in C-Unbiased. Overall, **VerSig-R** significantly outperforms both SigNet and the technique of Dutta et al. in C-Unbiased, achieving an EER of just 1.95% (and thus FAR $=$ FRR $= 1.95\%$).

Following this bias analysis on the CEDAR dataset, the author analyzed both the Bangla and Hindi datasets also looking for possible sources of bias. The author found that all the genuine image signatures for all the Bangla and Hindi writers are not centered. In fact, all the genuine signature images, for all the writers have the signature positioned at the left side, as shown in Figure 4.5b. To test whether this could introduce bias, a trimming operation was applied over all images in both datasets, cropping empty regions around the signature images (Figure 4.5c). Similarly to the renaming perform over the CEDAR dataset, the author thus defined B-Unbiased and B-Biased datasets based on Bangla, and H-Unbiased and H-Biased datasets based on Hindi dataset. As shown in Table 4.3, the centering of the signatures did not prove to be a significant source of bias.

4.4 Evaluation of Rotation and Scale Invariance

In real-world scenarios, signatures can be found in different orientations and sizes. Therefore, it is important for signature-verification methods to perform well in situations where the signatures being compared are subjected to differences in rotation and scale. To evaluate this property, the author propose the following variations of the previously-proposed X-Unbiased datasets (for $X \in \{\text{C, B, H}\}$, respectively CEDAR, Bangla, and Hindi):

- X-**UR** = X-Unbiased + **Rotations**, where each image in the original X-Unbiased dataset has been subjected to a different random rotation between 0 and 360 degrees;
- X-**US** = X-Unbiased + **Scalings**, where each image in the original X-Unbiased dataset has been subjected to a different random downscaling between 50% and

100% of its original size;

- *X*-**URS** = *X*-Unbiased + **Rotations** + **Scalings**, which combines both random rotations and downscalings (with different random parameters versus UR and US datasets).

Table 4.3 summarizes the results encountered by the experiments performed in the present work, grouped by dataset. As one can see, **VerSig-R** is significantly more robust against Rotations (UR datasets), with an average loss in accuracy of just 2.2 pp (percentage points) when replacing Unbiased by UR data. In comparison, the next-best method, SigNet, suffers a 8.4 pp loss in accuracy, while Dutta et al. loses 14.8 pp. Furthermore, **VerSig-R** method is adequately robust against Scalings (US datasets), with an average loss in accuracy of 2.2 pp (versus 0.9 pp for SigNet and 9.9 pp for Dutta et al.). Finally, the **VerSig-R** method is the most robust against combined Rotations and Scalings (URS datasets), with an average loss in accuracy of just 4.5 pp, versus 8.9 pp for SigNet and 22.7 for Dutta et al.

Table 4.3: Classification metrics for the methods evaluated in the experiments performed in the present work. Best results in each group are marked in bold, and results which are statistically equivalent to the best result are highlighted in green (Welch's t-test). Metric names marked with an asterisk (*) are affected by SigNet's test-set threshold selection procedure (see Section 4.2).

Dataset	Method	EER (%) (lower is better)	Acc.* (%) (higher is better)	F1-Score* (%)	Precision* (%)	Recall* (%)	ROC AUC
C-Unbiased	**VerSig-R**	$\mathbf{1.95 \pm 1.71}$	$\mathbf{97.42 \pm 1.95}$	$\mathbf{97.43 \pm 1.97}$	$\mathbf{97.08 \pm 3.03}$	$\mathbf{97.92 \pm 3.40}$	$\mathbf{99.75 \pm 0.44}$
C-Unbiased	SigNet	21.73 ± 9.17	79.51 ± 8.40	80.25 ± 7.74	78.90 ± 10.10	82.75 ± 9.83	85.43 ± 9.39
C-Unbiased	Dutta et al.	6.67 ± 3.58	92.87 ± 3.39	92.80 ± 3.48	93.53 ± 4.65	92.49 ± 6.13	97.84 ± 1.89
C-Unbiased	*Human*		79.20 ± 5.22				
C-UR	**VerSig-R**	$\mathbf{2.82 \pm 1.56}$	$\mathbf{96.71 \pm 1.95}$	$\mathbf{96.68 \pm 2.02}$	$\mathbf{97.02 \pm 2.43}$	$\mathbf{96.48 \pm 3.74}$	$\mathbf{99.52 \pm 0.49}$
C-UR	SigNet	25.76 ± 6.18	75.83 ± 6.62	78.00 ± 6.08	71.65 ± 5.78	85.84 ± 7.98	81.31 ± 6.78
C-UR	Dutta et al.	20.07 ± 3.82	79.18 ± 4.45	77.69 ± 7.03	82.64 ± 4.87	74.83 ± 12.46	87.85 ± 4.17
C-US	**VerSig-R**	$\mathbf{5.13 \pm 2.90}$	$\mathbf{94.17 \pm 2.84}$	$\mathbf{94.09 \pm 2.97}$	$\mathbf{94.75 \pm 3.39}$	$\mathbf{93.70 \pm 5.30}$	$\mathbf{98.71 \pm 1.38}$
C-US	SigNet	25.76 ± 9.23	75.05 ± 9.15	76.46 ± 7.46	73.97 ± 10.05	79.76 ± 6.45	80.19 ± 10.54
C-US	Dutta et al.	23.68 ± 4.02	75.76 ± 4.77	73.94 ± 7.25	78.63 ± 2.66	70.87 ± 12.44	84.74 ± 4.50
C-URS	**VerSig-R**	$\mathbf{6.26 \pm 2.16}$	$\mathbf{93.39 \pm 2.09}$	$\mathbf{93.38 \pm 2.16}$	$\mathbf{93.24 \pm 2.34}$	$\mathbf{93.62 \pm 3.65}$	$\mathbf{98.34 \pm 1.24}$
C-URS	SigNet	27.10 ± 4.95	74.67 ± 5.30	76.92 ± 5.46	70.64 ± 4.91	84.96 ± 8.69	79.82 ± 5.09
C-URS	Dutta et al.	34.23 ± 2.31	64.95 ± 2.49	59.39 ± 5.27	70.47 ± 3.31	52.03 ± 8.35	71.42 ± 3.36
C-URS	*Human*		78.20 ± 4.66				

B-Unbiased	**VerSig-R**	$\mathbf{15.87 \pm 4.02}$	83.76 ± 3.68	83.57 ± 4.38	83.87 ± 2.36	83.62 ± 7.70	$\mathbf{92.30 \pm 3.32}$
B-Unbiased	SigNet	16.46 ± 4.20	$\mathbf{84.57 \pm 3.82}$	$\mathbf{85.14 \pm 3.67}$	82.31 ± 4.70	$\mathbf{88.45 \pm 5.14}$	91.20 ± 3.71
B-Unbiased	Dutta et al.	16.16 ± 3.36	83.55 ± 3.22	83.29 ± 3.44	$\mathbf{84.83 \pm 5.03}$	82.39 ± 6.60	91.90 ± 2.78
B-UR	**VerSig-R**	$\mathbf{17.66 \pm 2.62}$	$\mathbf{81.98 \pm 2.62}$	$\mathbf{81.80 \pm 2.65}$	$\mathbf{82.93 \pm 4.47}$	81.05 ± 4.81	$\mathbf{90.80 \pm 2.35}$
B-UR	SigNet	21.05 ± 4.37	79.62 ± 4.23	80.43 ± 3.88	77.51 ± 4.42	$\mathbf{83.67 \pm 3.98}$	86.94 ± 4.20
B-UR	Dutta et al.	32.74 ± 3.49	66.82 ± 3.60	65.12 ± 5.26	68.46 ± 3.76	62.68 ± 8.64	72.96 ± 4.10
B-US	**VerSig-R**	16.96 ± 4.01	82.90 ± 3.92	82.69 ± 4.45	83.40 ± 3.95	82.34 ± 7.21	90.91 ± 3.46
B-US	SigNet	$\mathbf{15.35 \pm 3.66}$	$\mathbf{85.45 \pm 3.78}$	$\mathbf{85.82 \pm 3.59}$	$\mathbf{84.03 \pm 5.06}$	$\mathbf{87.96 \pm 4.68}$	$\mathbf{91.76 \pm 3.13}$
B-US	Dutta et al.	22.15 ± 4.24	77.79 ± 4.06	77.10 ± 4.84	79.42 ± 4.75	75.39 ± 7.79	86.12 ± 4.24
B-URS	**VerSig-R**	$\mathbf{19.24 \pm 2.78}$	$\mathbf{80.65 \pm 2.96}$	$\mathbf{80.44 \pm 3.01}$	$\mathbf{81.60 \pm 4.61}$	79.71 ± 5.28	$\mathbf{89.36 \pm 2.90}$
B-URS	SigNet	21.78 ± 3.65	79.24 ± 3.46	80.39 ± 3.31	76.29 ± 3.85	$\mathbf{85.16 \pm 4.90}$	86.17 ± 3.68
B-URS	Dutta et al.	35.99 ± 3.14	63.76 ± 3.12	62.39 ± 5.27	64.60 ± 2.85	60.94 ± 8.99	69.03 ± 3.91
H-Unbiased	**VerSig-R**	16.39 ± 3.47	83.59 ± 3.37	83.53 ± 3.46	83.85 ± 3.94	83.36 ± 4.69	91.61 ± 3.31
H-Unbiased	SigNet	$\mathbf{15.22 \pm 4.02}$	$\mathbf{85.14 \pm 3.90}$	$\mathbf{85.33 \pm 3.88}$	$\mathbf{84.35 \pm 4.41}$	$\mathbf{86.48 \pm 4.63}$	$\mathbf{92.74 \pm 3.03}$
H-Unbiased	Dutta et al.	19.36 ± 2.67	80.69 ± 2.57	80.96 ± 2.61	79.83 ± 2.69	82.19 ± 3.37	88.94 ± 2.54
H-UR	**VerSig-R**	$\mathbf{20.64 \pm 4.23}$	$\mathbf{79.39 \pm 4.21}$	$\mathbf{79.21 \pm 4.26}$	$\mathbf{80.02 \pm 4.86}$	$\mathbf{78.55 \pm 4.89}$	$\mathbf{87.30 \pm 4.25}$
H-UR	SigNet	31.85 ± 3.30	68.55 ± 3.38	70.02 ± 3.53	66.84 ± 2.86	73.59 ± 4.83	74.42 ± 4.07
H-UR	Dutta et al.	33.36 ± 2.65	66.56 ± 2.23	65.34 ± 3.09	67.71 ± 1.90	63.26 ± 4.79	73.20 ± 2.86

H-US	**VerSig-R**	18.87 ± 3.73	81.03 ± 3.84	80.96 ± 4.04	81.19 ± 4.03	80.87 ± 5.26	88.88 ± 3.92
H-US	SigNet	$\mathbf{14.41 \pm 3.57}$	$\mathbf{86.06 \pm 3.26}$	$\mathbf{86.33 \pm 2.90}$	$\mathbf{85.26 \pm 4.83}$	$\mathbf{87.59 \pm 2.38}$	$\mathbf{92.89 \pm 2.92}$
H-US	Dutta et al.	26.15 ± 2.34	73.71 ± 2.40	73.26 ± 2.49	74.71 ± 3.57	72.13 ± 4.43	81.99 ± 2.75
H-URS	**VerSig-R**	$\mathbf{22.54 \pm 4.22}$	$\mathbf{77.28 \pm 4.23}$	$\mathbf{77.06 \pm 4.25}$	$\mathbf{77.87 \pm 4.64}$	$\mathbf{76.36 \pm 4.76}$	$\mathbf{84.99 \pm 4.55}$
H-URS	SigNet	31.77 ± 4.33	68.70 ± 4.29	70.13 ± 4.66	67.05 ± 3.91	73.91 ± 7.44	74.34 ± 5.10
H-URS	Dutta et al.	39.66 ± 2.23	60.23 ± 1.86	59.19 ± 2.70	60.77 ± 1.97	57.85 ± 4.40	64.14 ± 2.53
C-Biased	**VerSig-R**	0.20 ± 0.26	99.67 ± 0.34	99.67 ± 0.34	99.55 ± 0.66	99.79 ± 0.31	$\mathbf{100.00 \pm 0.00}$
C-Biased	SigNet	$\mathbf{0.00 \pm 0.00}$	$\mathbf{100.00 \pm 0.00}$	$\mathbf{100.00 \pm 0.00}$	$\mathbf{100.00 \pm 0.00}$	$\mathbf{100.00 \pm 0.00}$	$\mathbf{100.00 \pm 0.00}$
C-Biased	Dutta et al.	8.32 ± 4.36	90.93 ± 4.52	90.66 ± 5.04	92.41 ± 6.12	90.07 ± 9.89	96.44 ± 3.43
B-Biased	**VerSig-R**	14.98 ± 3.31	84.80 ± 3.17	84.67 ± 3.49	85.17 ± 3.32	84.45 ± 6.00	$\mathbf{93.10 \pm 2.57}$
B-Biased	SigNet	$\mathbf{14.30 \pm 3.99}$	$\mathbf{86.46 \pm 3.85}$	$\mathbf{86.56 \pm 3.62}$	$\mathbf{86.37 \pm 5.36}$	$\mathbf{87.04 \pm 4.59}$	92.48 ± 3.56
B-Biased	Dutta et al.	17.25 ± 3.58	82.74 ± 3.56	82.39 ± 4.11	83.90 ± 4.20	81.45 ± 7.40	91.07 ± 3.10
H-Biased	**VerSig-R**	16.03 ± 3.58	83.95 ± 3.61	83.85 ± 3.78	$\mathbf{84.29 \pm 3.91}$	83.57 ± 5.31	91.59 ± 3.65
H-Biased	SigNet	$\mathbf{14.93 \pm 2.87}$	$\mathbf{85.73 \pm 2.66}$	$\mathbf{86.26 \pm 2.54}$	83.32 ± 3.40	$\mathbf{89.52 \pm 3.09}$	$\mathbf{93.25 \pm 1.85}$
H-Biased	Dutta et al.	18.72 ± 2.62	81.19 ± 2.04	81.40 ± 2.07	80.55 ± 2.63	82.38 ± 3.28	89.52 ± 1.98

4.5 Ablation Study

Table 4.4 presents an analysis of the proposed set of features and their individual performance, averaged over the CEDAR, Bangla and Hindi datasets for simplicity. Note how the proposed CLIP feature extraction, which uses the preprocessing pipeline described in Section 3.2, performs better in the signature-verification context than the original CLIP pipeline (which simply crops the input image to a square, discarding information from the signatures, which are often rectangular in shape). Furthermore, combining the learned CLIP features with the generated MLS features (**VerSig-R**), results in further increase in performance (higher accuracy and lower EER), while also reducing the standard deviation between different folds of the cross validation experiment.

It is also worth to notice that the improvement in performance from the features obtained using the original CLIP preprocessing pipeline and **VerSig-R**, to the final proposed features is considerably better in the unbiased version of the datasets, and also the only rotated and only scaled versions of the datasets. In the rotated and scaled version of the datasets there is only a small improvement when including the proposed MLS features with the proposed CLIP features.

Overall, Table 4.4 shows that the proposed preprocessing pipeline for CLIP feature generation and its combination with the proposed MLS handcrafted features have a positive impact over the metrics in general.

Table 4.4: Ablation study (averaged over CEDAR, Bangla and Hindi datasets). **VerSig-R** = Proposed MLS + CLIP Features. The best result in each group is highlighted in bold. UR = Unbiased + Rotations; US = Unbiased + Scales; URS = Unbiased + Rotations + Scales.

Dataset	Method	EER (%) (lower is better)	Accuracy* (%) (higher is better)
Unbiased	**VerSig-R**	$\mathbf{11.40 \pm 3.07}$	$\mathbf{88.26 \pm 3.00}$
Unbiased	Our MLS Features	17.19 ± 4.87	82.71 ± 4.79
Unbiased	Our CLIP Features	14.60 ± 4.06	85.28 ± 4.04
Unbiased	Original CLIP	19.03 ± 3.35	80.59 ± 3.69
UR	**VerSig-R**	$\mathbf{13.71 \pm 2.80}$	$\mathbf{86.03 \pm 2.93}$
UR	Our MLS Features	22.88 ± 4.39	76.98 ± 4.62
UR	Our CLIP Features	14.99 ± 2.86	84.74 ± 2.84
UR	Original CLIP	19.68 ± 3.26	80.11 ± 3.26
US	**VerSig-R**	$\mathbf{13.65 \pm 3.55}$	$\mathbf{86.03 \pm 3.53}$
US	Our MLS Features	31.31 ± 3.09	68.72 ± 3.17
US	Our CLIP Features	15.18 ± 4.03	84.60 ± 3.95
US	Original CLIP	21.36 ± 4.28	78.27 ± 4.34
URS	**VerSig-R**	$\mathbf{16.01 \pm 3.05}$	$\mathbf{83.77 \pm 3.09}$
URS	Our MLS Features	35.00 ± 3.82	64.97 ± 4.01
URS	Our CLIP Features	16.19 ± 3.02	83.64 ± 3.05
URS	Original CLIP	21.48 ± 3.31	78.11 ± 3.37

Table 4.5: Classification metrics for the MCYT and GPDS datasets. Metric names marked with an asterisk (*) are affected by SigNet's test-set threshold selection procedure (see Section 4.2). ** SigNet and Dutta et al. results shown in the table were extracted from (DEY et al., 2017), no experiments were reproduced.

Dataset	Method	EER (%) (lower is better)	Acc.* (%) (higher is better)	F1-Score* (%)	Precision* (%)	Recall* (%)	ROC AUC
GPDS-200	**VerSig-R**	32.14 ± 2.94	67.57 ± 2.79	66.98 ± 2.84	68.31 ± 3.33	65.86 ± 3.87	74.16 ± 3.62
GPDS-500	**VerSig-R**	30.99 ± 1.59	69.09 ± 1.62	69.37 ± 1.79	68.75 ± 1.51	70.03 ± 2.52	75.71 ± 1.95
GPDS-1000	**VerSig-R**	30.17 ± 0.97	69.97 ± 0.93	70.58 ± 1.02	69.18 ± 0.96	72.05 ± 1.69	76.75 ± 1.08
GPDS-1000	**VerSig-R with PCA**	30.10 ± 0.95	70.04 ± 1.0	70.64 ± 1.23	69.23 ± 0.91	72.14 ± 2.17	76.81 ± 1.05
GPDS-4000	**VerSig-R with PCA**	31.04 ± 0.59	69.25 ± 0.58	70.24 ± 0.80	68.05 ± 0.56	72.59 ± 1.64	75.66 ± 0.78
GPDS-4000	**SigNet****	22.24	77.76	–	–	–	–
GPDS-4000	**Dutta et al.****	27.98	73.67	–	–	–	–
MCYT	**VerSig-R**	30.16 ± 4.26	69.27 ± 4.33	68.14 ± 4.88	71.37 ± 6.83	66.21 ± 8.56	76.59 ± 5.35

4.6 GPDS and MCYT dataset results

This section discusses a set of experiments over the GPDS-4000 (FERRER; DIAZ-CABRERA; MORALES, 2015; FERRER; DIAZ-CABRERA; MORALES, 2013) dataset and the MCYT (ORTEGA-GARCIA et al., 2003; FIERREZ-AGUILAR et al., 2004) dataset with the same methodology used for CEDAR and BHSig260 datasets, described in Section 4.2. Experiments were performed over 3 sub-datasets of the GPDS-4000 dataset considering only the 200, 500 and 1000 first writers of the whole GPDS-4000 dataset (which we call GPDS-200, GPDS-500 and GPDS-1000, respectively). Results for these experiments can be encountered in the Table 4.5. In the case of the whole dataset, a dimensionality reduction process was needed because the workstation where experiments were executed could not manage the amount of memory required to perform this particular experiment (GPDS-4000 has 4000 writers with 552 signature pairs per writer (balanced), resulting in a total of 2,208,000 signatures).

A dimensionality reduction was performed after the feature generation using the well-known concept of Principal Component Analysis (PCA) (WOLD; ESBENSEN; GELADI, 1987). The idea behind using dimensionality reduction is to reduce the size (in memory) of the final feature vector for each image and in this way reduce the amount of memory required to perform the experiment. For this particular case, the number of principal components was set to 128 (this number was defined arbitrarily), the dimensionality reduction was applied after obtaining the final feature vector (2048 features from CLIP and 226 features from handcrafted MLD $=$ 2274-D), so the feature generation process was not modified at all, just considering the 128-D most representative feature dimensions according to the PCA algorithm. The amount of memory used to store the features were reduced by approximately 17 times for each image: with PCA applied the complete CEDAR dataset (2640 images) has a size of $\sim$1.3MB, while without PCA it has a size of $\sim$23MB.

Results obtained with the additional PCA dimensionality reduction over the GPDS-4000 and GPDS-1000 datasets are shown in Table 4.5. The differences in metrics for GPDS-1000 between both trained classifiers, with and without PCA, are minimal, showing that the dimensionality reduction step likely does not significantly impact the result shown for GPDS-4000 as well. Additionally, results for GPDS-4000 dataset obtained by Dey et al. (2017) and Dutta, Pal and Lladós (2016), as reported by Dey et al. (2017), are shown in the same table. Please consider that this experiments were not reproduced by the author. Finally, the results obtained by **VerSig-R** over the MCYT dataset are reported in the last

Table 4.6: General results for **VerSig-R** method after applying a dimensionality reduction to the final feature vectors from 2274-D to 128-D using Principal Component Analysis.

Dataset	EER (%) (lower is better)	Accuracy* (%) (higher is better)	Exec. Time (in seconds)
C-Biased	$\mathbf{0.16 \pm 0.25}$	$\mathbf{99.42 \pm 0.73}$	5.12
C-Unbiased	$\mathbf{2.71 \pm 1.82}$	$\mathbf{97.00 \pm 1.89}$	6.09
C-UR	$\mathbf{3.62 \pm 1.41}$	$\mathbf{96.08 \pm 1.60}$	8.22
C-URS	$\mathbf{7.58 \pm 2.07}$	$\mathbf{92.33 \pm 2.03}$	9.89
C-US	$\mathbf{6.94 \pm 2.91}$	$\mathbf{92.68 \pm 3.00}$	7.87
B-Biased	$\mathbf{15.35 \pm 4.71}$	$\mathbf{84.33 \pm 4.64}$	20.47
B-Unbiased	$\mathbf{17.03 \pm 4.91}$	$\mathbf{82.85 \pm 4.73}$	21.21
B-UR	$\mathbf{17.81 \pm 5.19}$	$\mathbf{82.03 \pm 5.20}$	25.82
B-URS	$\mathbf{19.36 \pm 5.44}$	$\mathbf{80.55 \pm 5.48}$	27.82
B-US	$\mathbf{18.94 \pm 5.59}$	$\mathbf{80.85 \pm 5.31}$	25.05
H-Biased	$\mathbf{16.75 \pm 3.07}$	$\mathbf{83.04 \pm 2.90}$	39.48
H-Unbiased	$\mathbf{17.23 \pm 3.12}$	$\mathbf{82.46 \pm 3.03}$	37.95
H-UR	$\mathbf{20.97 \pm 3.04}$	$\mathbf{79.01 \pm 3.14}$	49.75
H-URS	$\mathbf{23.85 \pm 3.14}$	$\mathbf{76.13 \pm 3.23}$	57.72
H-US	$\mathbf{19.83 \pm 3.33}$	$\mathbf{79.95 \pm 3.10}$	47.56

row of the same table.

From the results shown in Table 4.5, one can notice that for GPDS-4000, **VerSig-R** is not as well-performant as in the case of CEDAR dataset for example. GPDS-4000 dataset is an extremely difficult dataset since looking at the references samples, it shows an extremely higher intraclass variability in genuine and forged samples (different sizes, signature's stroke width, signature shapes, etc.), also the author considers that being synthetically generated signatures, they could not replicate the curvatures and stroke intersections in the way a human would do, affecting the performance of **VerSig-R**.

Based on the results obtained using PCA dimensionality reduction technique over the GPDS-1000 (1000 first writers of GPDS-4000), the author applies the same dimensionality reduction process for CEDAR and BHSig260 datasets. Results shown in Table 4.6 demonstrate that the performance loss is extremely low, but in contrast, it offers an extremely faster execution time (see Section 4.10) and a 10 times reduction in size for the final feature vector. It is worth noticing that dimensionality reduction with PCA is not part of the main **VerSig-R** pipeline, but it is an optional step, especially useful when memory resources are limited.

4.7 Impact of reference samples on classification performance

The number of reference samples using during the training phase is an extremely important variable to consider because normally a large number of both genuine and forgery signature images are not available. In this sense, a method which achieves similar or better performance than another using a smaller amount of reference samples during the training phase should be considered better. Also in this direction, writer-independent approaches are preferable, since it does not need a lot of reference samples per writer to generate a well-performant classification model.

In Tables 4.7, 4.8, and 4.9, one can find an analysis about the impact of the number of references samples used during the training phase into the two main metrics considered in this work: Accuracy and Equal Error Rate (EER) for the features proposed in the present research work. Additionally, a validation step was performed over the signature samples not used during cross-validation training, using a classifier trained on all the corresponding reference samples, and evaluated on the remaining samples. This metric is presented in the mentioned tables as Val. (%). It is worth to notice that the fewer the reference samples, the worse the metrics, but down to 4 references (even 3 references in some cases) the Val. metrics could still be considered acceptable. Furthermore, with a very reduced number of reference samples, the possibility of overfitting during training increases, reducing dramatically the classifier's performance.

The mentioned tables present information for CEDAR, Bangla and Hindi datasets respectively for each of their versions.

4.8 Impact of using only random forgeries during training

In the same direction that the number of reference samples used during training, the author performed some experiments to analyze the impact of using only random forgeries during the training phase. As indicated in Chapter 2, signature pairs including a random forgery are usually generated using *(i)* one genuine signature sample from a writer, and *(ii)* a genuine signature sample from another writer, to act as a forgery signature for the first writer. In this way, one does not need skilled forgeries at all to train the writer independent classifier.

The author generated new signature pairs for writer X by randomly chosing another writer Y and using only Y's genuine signatures to act as the forged signatures for writer

Table 4.7: Study of the impact of the number of reference samples used during training with **VerSig-R** (MLS + CLIP features) for the CEDAR dataset. Accuracy and EER and execution time metrics are calculated from cross validation during training, Validation accuracy (Val.) is calculated from evaluation of the test set.

Dataset	#References per writer	Accuracy* (%) (higher is better)	EER (%) (lower is better)	Val. (%) (accuracy)	Exec. Time (in seconds)
Biased	All=24	$\mathbf{99.76 \pm 0.27}$	$\mathbf{0.07 \pm 0.12}$	-	39.03
Biased	12	$\mathbf{99.47 \pm 0.73}$	$\mathbf{0.22 \pm 0.43}$	100.00	14.64
Biased	5	$\mathbf{98.43 \pm 1.73}$	$\mathbf{0.63 \pm 1.09}$	99.61	4.10
Biased	4	$\mathbf{97.89 \pm 2.67}$	$\mathbf{2.03 \pm 2.25}$	99.31	2.81
Biased	3	$\mathbf{95.89 \pm 3.45}$	$\mathbf{3.72 \pm 4.42}$	98.37	1.48
Biased	2	$\mathbf{93.50 \pm 6.21}$	$\mathbf{0.83 \pm 2.50}$	93.22	0.54
Unbiased	All=24	$\mathbf{97.81 \pm 1.96}$	$\mathbf{1.67 \pm 1.33}$	-	50.46
Unbiased	12	$\mathbf{96.98 \pm 2.09}$	$\mathbf{2.34 \pm 1.98}$	98.62	17.75
Unbiased	5	$\mathbf{93.53 \pm 3.39}$	$\mathbf{4.77 \pm 2.73}$	97.23	4.05
Unbiased	4	$\mathbf{93.06 \pm 4.27}$	$\mathbf{4.36 \pm 2.68}$	96.07	2.48
Unbiased	3	$\mathbf{91.50 \pm 4.46}$	$\mathbf{8.72 \pm 5.31}$	93.63	1.39
Unbiased	2	$\mathbf{100.00 \pm 0.00}$	$\mathbf{0.00 \pm 0.00}$	80.09	0.49
UR	All=24	$\mathbf{97.11 \pm 1.83}$	$\mathbf{2.49 \pm 1.59}$	-	74.94
UR	12	$\mathbf{94.10 \pm 3.19}$	$\mathbf{5.68 \pm 3.63}$	96.68	25.77
UR	5	$\mathbf{90.30 \pm 4.04}$	$\mathbf{8.07 \pm 3.34}$	92.00	5.05
UR	4	$\mathbf{89.67 \pm 3.93}$	$\mathbf{6.81 \pm 4.26}$	89.92	3.13
UR	3	$\mathbf{85.94 \pm 7.41}$	$\mathbf{12.44 \pm 8.97}$	86.23	1.69
UR	2	$\mathbf{97.17 \pm 4.35}$	$\mathbf{0.00 \pm 0.00}$	72.51	0.57
URS	All=24	$\mathbf{94.28 \pm 1.62}$	$\mathbf{5.59 \pm 1.75}$	-	92.46
URS	12	$\mathbf{90.18 \pm 3.33}$	$\mathbf{9.24 \pm 3.07}$	94.18	33.99
URS	5	$\mathbf{88.52 \pm 3.72}$	$\mathbf{11.30 \pm 2.76}$	88.52	6.17
URS	4	$\mathbf{87.81 \pm 4.59}$	$\mathbf{9.97 \pm 6.27}$	85.97	3.73
URS	3	$\mathbf{84.44 \pm 7.65}$	$\mathbf{16.11 \pm 9.60}$	81.19	1.80
URS	2	$\mathbf{94.33 \pm 4.67}$	$\mathbf{2.00 \pm 6.00}$	66.85	0.61
US	All=24	$\mathbf{94.49 \pm 2.59}$	$\mathbf{4.88 \pm 2.66}$	-	69.40
US	12	$\mathbf{92.36 \pm 3.99}$	$\mathbf{7.33 \pm 3.67}$	96.79	23.60
US	5	$\mathbf{89.77 \pm 5.39}$	$\mathbf{10.67 \pm 3.81}$	93.31	4.89
US	4	$\mathbf{89.64 \pm 4.13}$	$\mathbf{11.25 \pm 6.14}$	91.69	2.84
US	3	$\mathbf{86.56 \pm 6.12}$	$\mathbf{11.22 \pm 7.83}$	85.44	1.52
US	2	$\mathbf{98.00 \pm 4.00}$	$\mathbf{0.00 \pm 0.00}$	70.49	0.52

Table 4.8: Study of the impact of the number of reference samples used during training with **VerSig-R** (MLS + CLIP features) for the Bangla dataset. Accuracy and EER and execution time metrics are calculated from cross validation during training, Validation accuracy (Val.) is calculated from evaluation of the test set.

Dataset	#References per writer	Accuracy* (%) (higher is better)	EER (%) (lower is better)	Val. (%) (accuracy)	Exec. Time (in seconds)
Biased	All=24	$\mathbf{84.60 \pm 3.33}$	$\mathbf{14.85 \pm 3.58}$	-	205.09
Biased	12	$\mathbf{85.11 \pm 3.49}$	$\mathbf{14.61 \pm 3.52}$	89.28	56.17
Biased	5	$\mathbf{83.52 \pm 4.01}$	$\mathbf{16.17 \pm 3.76}$	86.10	10.29
Biased	4	$\mathbf{82.56 \pm 3.93}$	$\mathbf{17.34 \pm 4.23}$	85.08	6.83
Biased	3	$\mathbf{79.59 \pm 4.58}$	$\mathbf{21.24 \pm 4.39}$	82.98	3.18
Biased	2	$\mathbf{73.28 \pm 5.27}$	$\mathbf{26.17 \pm 8.10}$	78.61	1.17
Unbiased	All=24	$\mathbf{83.07 \pm 3.66}$	$\mathbf{16.42 \pm 3.93}$	-	214.97
Unbiased	12	$\mathbf{82.67 \pm 4.15}$	$\mathbf{16.49 \pm 4.11}$	88.21	57.59
Unbiased	5	$\mathbf{82.98 \pm 3.19}$	$\mathbf{16.57 \pm 3.41}$	85.19	10.09
Unbiased	4	$\mathbf{83.23 \pm 3.94}$	$\mathbf{16.62 \pm 4.35}$	84.51	6.39
Unbiased	3	$\mathbf{79.89 \pm 5.10}$	$\mathbf{19.94 \pm 4.91}$	82.40	3.16
Unbiased	2	$\mathbf{76.78 \pm 6.72}$	$\mathbf{23.72 \pm 8.90}$	78.38	1.17
UR	All=24	$\mathbf{82.50 \pm 2.96}$	$\mathbf{17.48 \pm 3.05}$	-	254.31
UR	12	$\mathbf{81.28 \pm 3.76}$	$\mathbf{18.50 \pm 3.47}$	80.40	66.24
UR	5	$\mathbf{83.29 \pm 4.30}$	$\mathbf{16.63 \pm 3.99}$	76.52	11.06
UR	4	$\mathbf{82.57 \pm 4.37}$	$\mathbf{17.94 \pm 5.14}$	76.60	6.78
UR	3	$\mathbf{83.72 \pm 6.18}$	$\mathbf{16.80 \pm 4.47}$	72.88	2.98
UR	2	$\mathbf{95.39 \pm 5.35}$	$\mathbf{5.11 \pm 5.59}$	62.28	0.94
URS	All=24	$\mathbf{82.07 \pm 3.16}$	$\mathbf{17.82 \pm 2.98}$	-	273.26
URS	12	$\mathbf{79.59 \pm 5.41}$	$\mathbf{20.13 \pm 5.32}$	77.71	70.54
URS	5	$\mathbf{83.03 \pm 3.39}$	$\mathbf{16.66 \pm 3.50}$	73.95	10.95
URS	4	$\mathbf{86.23 \pm 4.78}$	$\mathbf{14.51 \pm 5.20}$	67.65	6.34
URS	3	$\mathbf{86.06 \pm 3.47}$	$\mathbf{12.94 \pm 4.16}$	59.89	2.77
URS	2	$\mathbf{92.39 \pm 5.19}$	$\mathbf{7.11 \pm 5.17}$	54.97	0.97
US	All=24	$\mathbf{81.64 \pm 3.98}$	$\mathbf{18.18 \pm 4.19}$	-	244.41
US	12	$\mathbf{81.01 \pm 4.19}$	$\mathbf{18.75 \pm 4.31}$	86.25	64.39
US	5	$\mathbf{79.09 \pm 4.78}$	$\mathbf{20.61 \pm 4.96}$	82.36	12.04
US	4	$\mathbf{76.73 \pm 4.24}$	$\mathbf{22.01 \pm 3.92}$	81.46	7.26
US	3	$\mathbf{74.98 \pm 5.85}$	$\mathbf{24.19 \pm 5.16}$	77.96	3.46
US	2	$\mathbf{69.89 \pm 8.70}$	$\mathbf{30.11 \pm 11.43}$	73.61	1.16

Table 4.9: Study of the impact of the number of reference samples used during training with **VerSig-R** (MLS + CLIP features) for the Hindi dataset. Accuracy and EER and execution time metrics are calculated from cross validation during training, Validation accuracy (Val.) is calculated from evaluation of the test set.

Dataset	#References per writer	Accuracy* (%) (higher is better)	EER (%) (lower is better)	Val. (%) (accuracy)	Exec. Time (in seconds)
Biased	All=24	$\mathbf{83.69 \pm 3.97}$	$\mathbf{16.46 \pm 3.84}$	-	377.59
Biased	12	$\mathbf{83.04 \pm 4.38}$	$\mathbf{17.07 \pm 4.47}$	87.92	98.72
Biased	5	$\mathbf{81.23 \pm 4.87}$	$\mathbf{18.72 \pm 4.62}$	84.21	17.29
Biased	4	$\mathbf{81.20 \pm 4.76}$	$\mathbf{18.70 \pm 4.30}$	83.37	10.50
Biased	3	$\mathbf{78.34 \pm 4.80}$	$\mathbf{21.24 \pm 4.65}$	80.56	5.23
Biased	2	$\mathbf{78.69 \pm 6.10}$	$\mathbf{22.85 \pm 7.17}$	76.33	1.74
Unbiased	All=24	$\mathbf{83.89 \pm 3.99}$	$\mathbf{16.06 \pm 4.07}$	-	371.50
Unbiased	12	$\mathbf{83.09 \pm 4.35}$	$\mathbf{16.85 \pm 4.35}$	87.32	96.82
Unbiased	5	$\mathbf{80.98 \pm 4.65}$	$\mathbf{18.61 \pm 4.09}$	83.89	16.79
Unbiased	4	$\mathbf{81.65 \pm 4.15}$	$\mathbf{17.96 \pm 5.11}$	82.80	11.00
Unbiased	3	$\mathbf{80.53 \pm 5.01}$	$\mathbf{19.88 \pm 5.01}$	80.56	5.17
Unbiased	2	$\mathbf{77.04 \pm 7.53}$	$\mathbf{24.21 \pm 9.25}$	76.87	1.90
UR	All=24	$\mathbf{79.96 \pm 4.49}$	$\mathbf{20.01 \pm 4.39}$	-	485.87
UR	12	$\mathbf{78.90 \pm 4.59}$	$\mathbf{21.10 \pm 4.52}$	76.69	118.62
UR	5	$\mathbf{81.30 \pm 3.03}$	$\mathbf{18.73 \pm 3.08}$	70.92	18.36
UR	4	$\mathbf{81.03 \pm 4.15}$	$\mathbf{19.03 \pm 3.83}$	70.22	11.61
UR	3	$\mathbf{81.87 \pm 5.23}$	$\mathbf{18.62 \pm 5.48}$	65.66	5.63
UR	2	$\mathbf{86.23 \pm 7.90}$	$\mathbf{13.77 \pm 9.98}$	56.53	1.80
URS	All=24	$\mathbf{77.49 \pm 4.28}$	$\mathbf{22.34 \pm 4.21}$	-	548.85
URS	12	$\mathbf{78.93 \pm 4.28}$	$\mathbf{21.24 \pm 4.35}$	71.16	124.40
URS	5	$\mathbf{82.09 \pm 3.91}$	$\mathbf{18.16 \pm 4.17}$	67.20	18.69
URS	4	$\mathbf{82.07 \pm 3.22}$	$\mathbf{17.15 \pm 2.55}$	63.73	11.47
URS	3	$\mathbf{81.70 \pm 4.62}$	$\mathbf{19.25 \pm 5.32}$	62.70	5.74
URS	2	$\mathbf{87.44 \pm 6.68}$	$\mathbf{10.06 \pm 8.48}$	54.15	1.58
US	All=24	$\mathbf{80.99 \pm 4.48}$	$\mathbf{18.91 \pm 4.35}$	-	459.23
US	12	$\mathbf{79.52 \pm 5.17}$	$\mathbf{20.38 \pm 5.12}$	84.40	116.16
US	5	$\mathbf{75.41 \pm 6.04}$	$\mathbf{24.46 \pm 5.80}$	80.01	19.60
US	4	$\mathbf{74.41 \pm 6.24}$	$\mathbf{25.26 \pm 6.55}$	77.54	12.02
US	3	$\mathbf{72.88 \pm 5.92}$	$\mathbf{26.90 \pm 6.06}$	75.50	5.80
US	2	$\mathbf{69.21 \pm 6.24}$	$\mathbf{33.60 \pm 5.89}$	67.39	1.85

X. These signature pairs can be considered as a completely new dataset, having the same number of writers as well as the same number of signature pairs than the original dataset (since one simply replaced the skilled forgeries by random forgeries). In this sense, one can apply the **VerSig-R** method over this new dataset directly, considering the same methodology (see Section 4.2). A cross validation was applied in a series of experiments obtaining the results shown in Table 4.10. A single classifier model was generated for each dataset version based on the complete dataset reference samples (balanced dataset). Finally, the classifiers generated based on these datasets of *random forgeries* were also evaluated using the corresponding *skilled-forgery* datasets, which represent a more challenging situation than random forgeries. The results of this evaluation is also included in Table 4.10.

As see in Table 4.10, for the Unbiased datasets, the classifiers trained on random forgeries achieve Accuracy values above 60% when testing with random forgeries, and above 70% when testing with skilled forgeries, which the the most challenging of the two situations. It is unclear why the classifiers perform better in the skilled-forgeries case vs the random-forgeries case, despite being trained on random forgeries. One possible explanation could be that the handcrafted features measure the amount of curvature of the signature, and the curvature of two signatures from different writers (random forgeries) are not necessarily different, specially when both have the same handwriting style (same dataset). Nonetheless, training the classifier on random forgeries and using the classifier on skilled forgeries represents a common real-world situation, and as such these represent good results. One interesting experiment, which is planned as a future work, could be to analyze the impact of using a combination of random forgeries and a small number of skilled forgeries during the training phase. Finally, the behaviour observed on the metrics in Table 4.3 is repeated in Table 4.10, where one sees a performance reduction when considering dataset transformations such as scalings and rotations.

4.9 Cross-dataset validation

This experiment was considered by the author based on the cross-dataset validation experiments reported by Dey et al. (DEY et al., 2017). The main idea is to validate how well a classifier trained over a specific dataset generalizes when it is tested over another dataset. To perform this experiment the author used the trained classifiers for the three main datasets used in this work (CEDAR and BHSig260 (Bangla and Hindi)), in their

Table 4.10: Study of the impact of training only with *random forgeries*, while evaluating with random and skilled forgeries, over the Accuracy metric of **VerSig-R** (MLS + CLIP features) for all datasets and their transformations. Datasets: C = CEDAR, B = Bangla, H = Hindi. Transformations: UR = Unbiased + Rotations; US = Unbiased + Scales; URS = Unbiased + Rotations + Scales. Hardware specifications: RTX 2080 Ti GPU, Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz (16 threads)

Dataset (with **random** forgeries)	Accuracy (%) (on **random** forgeries)	Accuracy (%) (on **skilled** forgeries)	Exec. Time (in seconds)
C-Biased	$\mathbf{58.99 \pm 5.39}$	**75**	86.45
C-Unbiased	$\mathbf{61.77 \pm 5.26}$	**77**	55.91
C-UR	$\mathbf{51.51 \pm 3.11}$	**69**	65.27
C-US	$\mathbf{57.82 \pm 4.80}$	**71**	66.76
C-URS	$\mathbf{48.27 \pm 2.35}$	**65**	67.39
B-Biased	$\mathbf{62.66 \pm 3.79}$	**74**	109.58
B-Unbiased	$\mathbf{62.83 \pm 4.70}$	**72**	109.13
B-UR	$\mathbf{55.36 \pm 2.70}$	**66**	120.88
B-US	$\mathbf{46.20 \pm 2.28}$	**62**	125.58
B-URS	$\mathbf{49.09 \pm 2.08}$	**63**	127.57
H-Biased	$\mathbf{68.20 \pm 3.26}$	**72**	182.21
H-Unbiased	$\mathbf{68.59 \pm 3.09}$	**71**	186.40
H-UR	$\mathbf{54.27 \pm 3.05}$	**61**	205.02
H-US	$\mathbf{52.92 \pm 2.92}$	**64**	206.07
H-URS	$\mathbf{48.37 \pm 2.25}$	**57**	213.95

Table 4.11: Cross-dataset validation Accuracy of the proposed technique for the unbiased version of the datasets.

—	CEDAR	Bangla	Hindi
CEDAR	97.42	50.24	50.09
Bangla	52.74	83.76	68.76
Hindi	54.90	68.24	83.59

Table 4.12: Cross-dataset validation Accuracy of the proposed technique for the unbiased rotated (UR) version of the datasets.

—	CEDAR-UR	Bangla-UR	Hindi-UR
CEDAR-UR	96.71	50.05	50.17
Bangla-UR	52.90	81.98	64.02
Hindi-UR	51.70	70.81	79.39

unbiased versions. The same process was followed for the other versions of the datasets to analyze the results when transformed dataset are used (rotations and scales). Results obtained can be encountered in Tables 4.11, 4.12, 4.13 and 4.14

As one can expect, when training on CEDAR dataset and testing on Bangla/Hindi datasets the classifier performs poorly, and in the opposite case the results are just slightly better. Also, when training on Bangla and testing on Hindi the results are significantly better than testing with CEDAR, and in the opposite case the results are even better (training on Hindi and testing on Bangla). This could be explained in two statements:

- Datasets with a larger number of writers (not necessarily larger number of reference samples per writer) normally generalize better that one with fewer writers;
- Datasets normally generalize well on the same type of writing, CEDAR (Western style) and Hindi/Bangla (Asian style) are very different kind of writing styles.

Table 4.13: Cross-dataset validation Accuracy of the proposed technique for the unbiased scaled (US) version of the datasets.

—	CEDAR-US	Bangla-US	Hindi-US
CEDAR-US	94.17	50.22	49.98
Bangla-US	52.20	82.90	63.46
Hindi-US	52.45	66.19	81.03

Table 4.14: Cross-dataset validation Accuracy of the proposed technique for the unbiased rotated and scaled (URS) version of the datasets.

—	CEDAR-URS	Bangla-URS	Hindi-URS
CEDAR-URS	93.39	49.03	50.56
Bangla-URS	51.35	80.65	65.50
Hindi-URS	50.57	69.35	77.28

Table 4.15: Time (in minutes and seconds – MM:SS) for feature generation of **VerSig-R**, measured in the unbiased version of the datasets. CLIP features are generated on a RTX 2080 Ti GPU (with pre- and post-processing on the CPU), and MLS features are generated on a Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz, using 16 threads.

Dataset	# signature pairs	Avg. resolution	CLIP features	MLS features	Total time
CEDAR	30.360	544×350	00:21	06:47	07:08
Bangla	55.200	982×282	00:27	10:52	11:19
Hindi	88.320	989×282	00:57	22:26	23:23

4.10 Implementation Details and Execution Time

Since in our experiments we use signature pairs, one signature image can appear several times composing different signature pairs. In this sense, the author computes the features for each signature only once, and then simply reuses the feature vector when needed. CLIP feature generation for all crops takes a few seconds to complete (when computed with a GPU; the GPU used for this research work was an NVIDIA GeForce RTX 2080 Ti), but after that some post-processing process is applied to generate the final CLIP features (see Section 3.2). In our Python implementation, the whole process takes about 7 minutes for CEDAR-based datasets, about 11 minutes for Bangla-based datasets and about 23 minutes in the case of Hindi-based datasets. These timings are detailed in Table 4.15. Finally, applying PCA does not represent a considerable increase on time, since it takes approximately 1 second to be applied to a single dataset (this is valid for all datasets considered in this work) and even offers time reduction in other parts of the **VerSig-R** method (see Tables 4.6 and 4.16).

Execution times reported in the tables in the present research work are summarized in Table 4.16. These represent the time computed for the 10-fold cross-validation step, i.e. after the feature generation, and include training and testing time for all folds. As one can observe and as one can expect, execution time for cross validation is directly proportional to the number of reference signatures per-writer used during the training phase (values of R shown on the table). Additionally, one can notice that using random forgeries during the training phase leads to a significantly shorter execution time for the Bangla and Hindi datasets, in other words, it took less execution time for the SVM classifier to converge. Additionally, when using PCA (reducing the feature dimensionality from 2274-D to 128-D), execution time is reduced up to 10 times (see execution times for B-Unbiased for example).

Table 4.16: Execution time of **VerSig-R** (in seconds) for 10-fold cross-validation, including training and testing time, for CEDAR and BHSig260 and their transformations. The number R represents the number of reference signatures per-writer used during training. Datasets: C = CEDAR, B = Bangla, H = Hindi. Transformations: UR = Unbiased + Rotations; US = Unbiased + Scales; URS = Unbiased + Rotations + Scales.

Dataset	$R = 24$	$R = 24$ with PCA	$R = 12$	$R = 5$	$R = 24$ random forgeries
C-Biased	39.03	5.12	14.64	4.10	86.45
C-Unbiased	50.46	6.09	17.75	4.05	55.91
C-UR	74.94	8.22	25.77	5.05	65.27
C-URS	92.46	9.89	33.99	6.17	67.39
C-US	69.40	7.87	23.60	4.89	66.76
B-Biased	205.0	20.48	56.17	10.29	109.58
B-Unbiased	214.9	21.22	57.59	10.09	109.13
B-UR	254.3	25.82	66.24	11.06	120.88
B-URS	273.2	27.83	70.54	10.95	127.57
B-US	244.4	25.05	64.39	12.04	125.58
H-Biased	377.5	39.49	98.72	17.29	182.21
H-Unbiased	371.5	37.95	96.82	16.79	186.40
H-UR	485.8	49.76	118.62	18.36	205.02
H-URS	548.8	57.73	124.40	18.69	213.95
H-US	459.2	47.56	116.16	19.60	206.07

4.11 User Study with Human Subjects

The author designed a small user study to measure the expected human accuracy in the signature verification task. A pair of signature images is presented (side by side) to the user, who is asked to determine if the signatures are genuine (written by the same writer) or if one of them is a forgery (signatures are written by different writers). A total of five users were recruited for the mentioned study, with no previous experience in visual signature verification neither relation to the present research work. Each user saw a total of 200 signature pairs, being 100 from the C-Unbiased and 100 from the C-URS dataset (in this order). Within each dataset, the author separated a balanced set of 50 genuine and 50 forged signature pairs, which were presented to the users in a random order. All users saw the same set of images, and no time limited was imposed to them although the average time for test completion was about 25 minutes. The author built a single graphical interface for the human tests which allow the user to indicate their classification through keys 0 (to indicate a forgery) and 1 (to indicate genuine signatures), then the application automatically presents the next signature pair avoiding waste of time and maintaining user focus in the classification task.

As seen in Table 4.3 (rows in light blue), the average human accuracy was $\sim 79\%$, and the users did not suffer any significant performance loss due to rotations and scalings of the images (C-URS dataset). This reinforces the importance for the classification methods to exhibit rotation and scale invariance properties. The individual accuracies for the users were $\{82\%, 86\%, 80\%, 74\%, 74\%\}$ for C-Unbiased and $\{81\%, 81\%, 80\%, 79\%, 70\%\}$ for C-URS. Although a user study with just five participants could not be considered as statistically significant, this small user study allows the author to infer that signature verification is not an easy task for (untrained) humans, and state-of-the-art algorithms are able to significantly surpass human classification performance as well as being extremely faster than humans.

5 CONCLUSION

The present work introduced a new offline writer-independent signature verification method, based on a combination of Moving Least-Squares handcrafted features and features transferred from the CLIP convolutional neural network. **VerSig-R** significantly outperforms state-of-the-art techniques on the CEDAR dataset (Western-style handwriting), while simultaneously obtaining good results on the Bangla and Hindi datasets both belonging to BHSig260 large dataset. The worse performance numbers on the BHSig260 dataset when compared with the results for the CEDAR dataset could be explained by the different handwriting style for these datasets. In particular, the BHSig260 signatures have considerably minor portions of curved regions when compared to the signatures on the CEDAR dataset. Finally, **VerSig-R** exhibits some degree of rotation and scale invariance, being the best-performing technique across all datasets when the signature images are subjected to random changes in rotation and scale.

Additionally, this research work presented a discussion of unintended bias in the datasets (mainly on the CEDAR dataset), suggesting an approach to remove such bias and demonstrating that effectively removing such bias produces a decrease in the SigNet (DEY et al., 2017) proposed technique metrics, supported by a wide range of experiments. Another source of bias were investigated and discussed over the Bangla and Hindi datasets, although they did not affect considerably the metrics. In the same direction, new dataset versions are proposed in the present research work, applying transformations, such as rotations and scalings, to the original unbiased versions of the datasets.

It is clear from experimental results shown in Tables 4.11, 4.12, 4.13 and 4.14 that in order to get an offline writer-independent classifier that generalizes well for most of the handwriting styles (Western, Asian, Arabic, etc) the dataset needs to be more heterogeneus with respect to the training dataset. One interesting experiment (planned as future work) is to build a large dataset containing information from CEDAR, BHSig260, MCYT and GPDS and use it to train a single writer-independent classifier in order to analyze how well it generalizes.

Additionally, results reported in Tables 4.7, 4.8, 4.9 show that **VerSig-R** behaves surprisingly well using fewer number of references per writer during the training phase, i.e. the impact of decreasing the number of references during training is relatively low (although with only two references the validation accuracy decrease dramatically, especially for the rotated versions of the datasets). This fact is highly desirable for the specific case

of signature verification problem since normally a limited number of reference samples per writer are available in real-life scenarios. In the same direction, experiments using only random forgeries during the training phase were performed (Section 4.8) demonstrating that random forgeries could be a good initial point for building an ensemble of classifiers that helps with the classification task, and giving some directions to the author for future work and making the model more adaptable to real-life scenarios.

In addition to this, a general evaluation of execution time is performed for the whole cross-validation process as well as the two parts of the feature generation process (CLIP and MLS handcrafted features). Although classifiers could be trained and be ready to use in a few hours in the worst cases (and in a few minutes in the best cases), saving the features to disk for reproducibility is still demanding (some gigabytes of space in disk is required to save even the smallest dataset features (CEDAR)).

Finally, to the best of the author's knowledge, the first user study to obtain a reference measure of human performance on the signature verification task on the CEDAR dataset was introduced in the present research work. The results could allow us to infer that the human accuracy for CEDAR unbiased dataset and the CEDAR-URS versions are on average 79% for both datasets, demonstrating that signature verification is not an easy task, even for humans, and that **VerSig-R** probably outperforms the human performance.

The contributions of this thesis were presented at the SIBGRAPI 2021 conference and published as an article in the SIBGRAPI full paper proceedings (Main Track) (PACHAS; GASTAL, 2021).

6 FUTURE WORK

6.1 Training with Random Forgeries

One interesting experiment to perform is to analyze a new procedure for generating random forgeries during training. Instead of choosing a single writer to act as the "forgery source" for another writer (as discussed in Section 4.8), the idea would be to use the genuine signatures from *all* other writers to create signature pairs for a writer. The challenge with this approach is the significantly larger dataset that would be generated and would be required to use during training.

Additionally, as mentioned in Section 4.8, a pending experiment for the author, is to analyze the metrics obtained when we use a combination of random forgeries and a minimum number of skilled or simple forgeries. These future works have the objective to reduce the necessity of skilled forgeries for training the classifiers in order to introduce better generalization and the adaptability required for real-life scenarios.

6.2 Dataset Improvements

As one can see in Table 4.3 and Table 4.4, rotated versions of the datasets are the most difficult problem to solve. This could be explained by the lack of references samples, since the generated rotations were applied in a unique randomly chosen degree for each reference sample in the initial dataset (one rotation per signature). A more complete approach to deal with this problem is to generate more rotation reference samples, ideally one rotation for every single degree belonging to the interval $[0°, 360°[$, this way every rotation of signatures could be considered during the training phase and could lead to an even better performance and generalization for **VerSig-R**.

In the same direction, the author proposes to introduce data augmentation techniques, which is an (arguably) different approach than generating totally synthetic signatures (see (FERRER; DIAZ-CABRERA; MORALES, 2015; FERRER; DIAZ-CABRERA; MORALES, 2013) for synthetic signature generation applied to generate GPDS-4000). The proposal is to use data augmentation techniques to generate slightly transformed reference samples in the datasets, maintaining the inherent characteristics of a handwritten signature, and this way increment the number of samples used during training which at the same time could lead to a better performing classifier. This remains to be explored,

however, since signature images distorted by geometric augmentation transformations could lead to worse results, and colorspace transformations would probably not be very useful (signatures are most commonly dark ink on white paper).

Finally, according to the results obtained in the present research work, the most 'diverse' the training reference samples the best results any signature verification method will obtain. In the cross-dataset validation (Section 4.9) one can notice that in general, datasets with different handwriting style do not generalize well when testing over a dataset with a different handwriting style. Then, one reasonable experiment consists in combining datasets with different handwriting styles, in order to improve generalization across datasets. This could be done with CEDAR and BHSig260 datasets in order to validate the previous mentioned hypothesis.

6.3 Signature Identification

Signature identification consists on identifying the position of any signature in an static image, in order to extract it to perform signature verification. In the present work the author only approaches the signature verification problem, but in real-life scenarios signature identification also needs to be approached in order to make the signature verification scalable across an organization, for example. Some works exist on the literature that deal with the mentioned problem, and future work for the author consists on evaluating these works and if necessary proposing a new signature identification method which works well with the already proposed signature verification method.

6.4 Feature Generation

Features generated by **VerSig-R** behave well for CEDAR and BHSig260 datasets, achieving good accuracies and reducing the Equal Error Rate (EER). However, **VerSig-R** generates feature vectors of 2274-D per image, so, giving a pair of signature images, the actual dimensionality of the signature pair is 4548-D, making it extremely challenging to manage this information in memory for using when training the classifiers. As shown in Section 4.6, PCA dimensionality reduction manages to reduce the feature dimensionality without sacrificing metric performance. Future work for the author is to make more experiments in this direction as well as to evaluate ways to optimize the feature generation

process to include fewer but representative feature description of the signature pairs, maybe by concatenating images in the signature pair and calculating CLIP and MLS handcrafted features from only one bigger image.

REFERENCES

ALVAREZ, G.; SHEFFER, B.; BRYANT, M. Offline signature verification with convolutional neural networks. **Tech. Report, Stanford Univ.**, 2016.

BERTOLINI, D.; OLIVEIRA, L. S.; JUSTINO, E.; SABOURIN, R. Reducing forgeries in writer-independent off-line signature verification through ensemble of classifiers. **Pattern Recognition**, Elsevier, v. 43, n. 1, p. 387–396, 2010.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: **Proceedings of the fifth annual workshop on Computational learning theory**. [S.l.: s.n.], 1992. p. 144–152.

CANTRELL, C. D. **Modern mathematical methods for physicists and engineers**. [S.l.]: Cambridge University Press, 2000.

CHA, S.-H.; SRIHARI, S. N. Writer identification: statistical analysis and dichotomizer. In: SPRINGER. **Joint IAPR international workshops on statistical techniques in pattern recognition (SPR) and structural and syntactic pattern recognition (SSPR)**. [S.l.], 2000. p. 123–132.

CHAPELLE, O.; HAFFNER, P.; VAPNIK, V. N. Support vector machines for histogram-based image classification. **IEEE transactions on Neural Networks**, IEEE, v. 10, n. 5, p. 1055–1064, 1999.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. Imagenet: A large-scale hierarchical image database. In: IEEE. **2009 IEEE conference on computer vision and pattern recognition**. [S.l.], 2009. p. 248–255.

DEY, S.; DUTTA, A.; TOLEDO, J. I.; GHOSH, S. K.; LLADÓS, J.; PAL, U. SigNet: Convolutional siamese network for writer independent offline signature verification. **arXiv:1707.02131**, 2017.

DIAZ, M.; FERRER, M. A.; IMPEDOVO, D.; MALIK, M. I.; PIRLO, G.; PLAMONDON, R. A perspective analysis of handwritten signature technology. **Acm Computing Surveys (Csur)**, ACM New York, NY, USA, v. 51, n. 6, p. 1–39, 2019.

DODGE, Y. **The Concise Encyclopedia of Statistics**. [S.l.]: Springer, 2008. ISBN 9780387317427.

DUTTA, A.; PAL, U.; LLADÓS, J. Compact correlated features for writer independent signature verification. In: IEEE. **Intl. Conference on Pattern Recognition**. [S.l.], 2016. p. 3422–3427.

FERRER, M. A.; ALONSO, J. B.; TRAVIESO, C. M. Offline geometric parameters for automatic signature verification using fixed-point arithmetic. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, v. 27, n. 6, p. 993–997, 2005.

FERRER, M. A.; DIAZ-CABRERA, M.; MORALES, A. Synthetic Off-Line Signature Image Generation. In: **6th IAPR International Conference on Biometrics**. [S.l.: s.n.], 2013. p. 1–7. 4–7 June 2013, Madrid. doi: 10.1109/ICB.2013.6612969.

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**, IEEE, v. 37, n. 3, p. 667–680, 2015.

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: SPRINGER LNCS-3087. **Workshop on Biometric Authentication**. [S.l.], 2004. p. 295–306.

FREITAS, C.; BORTOLOZZI, F.; SABOURIN, R.; FACON, J. Bases de dados de cheques bancarios brasileiros. 1998.

HADNAGY, C. **Social engineering: The art of human hacking**. [S.l.]: John Wiley & Sons, 2010.

HAFEMANN, L. G.; SABOURIN, R.; OLIVEIRA, L. S. Writer-independent feature learning for offline signature verification using deep convolutional neural networks. In: IEEE. **Intl. Joint Conference on Neural Networks**. [S.l.], 2016. p. 2576–2583.

HAFEMANN, L. G.; SABOURIN, R.; OLIVEIRA, L. S. Learning features for offline handwritten signature verification using deep convolutional neural networks. **Pattern Recognition**, Elsevier, v. 70, p. 163–176, 2017.

HAFEMANN, L. G.; SABOURIN, R.; OLIVEIRA, L. S. Characterizing and evaluating adversarial examples for offline handwritten signature verification. **IEEE Transactions on Information Forensics and Security**, v. 14, n. 8, p. 2153–2166, 2019.

HAFEMANN, L. G.; SABOURIN, R.; OLIVEIRA, L. S. Meta-learning for fast classifier adaptation to new users of signature verification systems. **IEEE Transactions on Information Forensics and Security**, IEEE, v. 15, p. 1735–1745, 2019.

HAMADENE, A.; CHIBANI, Y. One-class writer-independent offline signature verification using feature dissimilarity thresholding. **IEEE Transactions on Information Forensics and Security**, IEEE, v. 11, n. 6, p. 1226–1238, 2016.

HAMEED, M. M.; AHMAD, R.; KIAH, M. L. M.; MURTAZA, G. Machine learning-based offline signature verification systems: A systematic review. **Signal Processing: Image Communication**, Elsevier, p. 116139, 2021.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 770–778.

HUANG, K.; YAN, H. Off-line signature verification based on geometric feature extraction and neural network classification. **Pattern Recognition**, Elsevier, v. 30, n. 1, p. 9–17, 1997.

IANDOLA, F.; MOSKEWICZ, M.; KARAYEV, S.; GIRSHICK, R.; DARRELL, T.; KEUTZER, K. Densenet: Implementing efficient convnet descriptor pyramids. **arXiv preprint arXiv:1404.1869**, 2014.

KALERA, M. K.; SRIHARI, S.; XU, A. Offline signature verification and identification using distance statistics. **Intl. Journal of Pattern Recog. and Artificial Intelligence**, World Scientific, v. 18, n. 07, p. 1339–1360, 2004.

KUMAR, A.; BHATIA, K. A survey on offline handwritten signature verification system using writer dependent and independent approaches. In: IEEE. **International Conference on Advances in Computing, Communication, & Automation**. [S.l.], 2016. p. 1–6.

KUMAR, R.; SHARMA, J.; CHANDA, B. Writer-independent off-line signature verification using surroundedness feature. **Pattern Recognition Letters**, Elsevier, v. 33, n. 3, p. 301–308, 2012.

LANCASTER, P.; SALKAUSKAS, K. Surfaces generated by moving least squares methods. **Mathematics of computation**, v. 37, n. 155, p. 141–158, 1981.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep Learning. **Nature**, Nature Publishing Group, v. 521, n. 7553, p. 436–444, may 2015. ISSN 1476-4687.

LEVIN, D. The approximation power of moving least-squares. **Mathematics of computation**, v. 67, n. 224, p. 1517–1531, 1998.

LIWICKI, M.; MALIK, M. I.; HEUVEL, C. E. V. D.; CHEN, X.; BERGER, C.; STOEL, R.; BLUMENSTEIN, M.; FOUND, B. Signature verification competition for online and offline skilled forgeries (sigcomp2011). In: IEEE. **2011 International conference on document analysis and recognition**. [S.l.], 2011. p. 1480–1484.

MOHAMMED, R. A.; NABI, R. M.; SARDASHT, M.; MAHMOOD, R.; NABI, R. M. State-of-the-art in handwritten signature verification system. In: IEEE. **Intl. Conference on Computational Science and Computational Intelligence**. [S.l.], 2015. p. 519–525.

MUNICH, M. E.; PERONA, P. Visual identification by signature tracking. **IEEE TPAMI**, IEEE, v. 25, n. 2, p. 200–217, 2003.

OLIVEIRA, L. S.; JUSTINO, E.; SABOURIN, R. Off-line signature verification using writer-independent approach. In: IEEE. **Intl. Joint Conference on Neural Networks**. [S.l.], 2007. p. 2539–2544.

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. **IEE Proceedings-Vision, Image and Signal Processing**, IET, v. 150, n. 6, p. 395–401, December 2003.

OTSU, N. A threshold selection method from gray-level histograms. **IEEE Trans. on Systems, Man, and Cybernetics**, IEEE, v. 9, p. 62–66, 1979.

PACHAS, F. E. H.; GASTAL, E. S. L. An offline writer-independent signature verification method with robustness against scalings and rotations. In: **2021 34th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. [s.n.], 2021. p. 322–329. Available from Internet: <https://doi.org/10.1109/SIBGRAPI54419.2021.00051>.

PAL, S.; ALAEI, A.; PAL, U.; BLUMENSTEIN, M. Performance of an off-line signature verification method based on texture features on a large indic-script signature dataset. In: IEEE. **12th IAPR workshop on document analysis systems**. [S.l.], 2016. p. 72–77.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

RABINER, L. Fundamentals of speech recognition. **Fundamentals of speech recognition**, PTR Prentice Hall, 1993.

RADFORD, A.; KIM, J. W.; HALLACY, C.; RAMESH, A.; GOH, G.; AGARWAL, S.; SASTRY, G.; ASKELL, A.; MISHKIN, P.; CLARK, J. et al. Learning transferable visual models from natural language supervision. **preprint arXiv:2103.00020**, 2021.

RISH, I. et al. An empirical study of the naive bayes classifier. In: **IJCAI 2001 workshop on empirical methods in artificial intelligence**. [S.l.: s.n.], 2001. v. 3, n. 22, p. 41–46.

SAKOE, H.; CHIBA, S. Dynamic programming algorithm optimization for spoken word recognition. **IEEE transactions on acoustics, speech, and signal processing**, IEEE, v. 26, n. 1, p. 43–49, 1978.

SANDLER, M.; HOWARD, A.; ZHU, M.; ZHMOGINOV, A.; CHEN, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2018. p. 4510–4520.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. **arXiv preprint arXiv:1409.1556**, 2014.

SOLEYMANPOUR, E.; RAJAE, B.; POURREZA, H. R. Offline handwritten signature identification and verification using contourlet transform and support vector machine. In: IEEE. **2010 6th Iranian Conference on Machine Vision and Image Processing**. [S.l.], 2010. p. 1–6.

SOUZA, V. L.; OLIVEIRA, A. L.; SABOURIN, R. A writer-independent approach for offline signature verification using deep convolutional neural networks features. In: IEEE. **Brazilian Conference on Intelligent Systems**. [S.l.], 2018. p. 212–217.

VARGAS, F.; FERRER, M.; TRAVIESO, C.; ALONSO, J. Off-line handwritten signature gpds-960 corpus. In: IEEE. **Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)**. [S.l.], 2007. v. 2, p. 764–768.

WOLD, S.; ESBENSEN, K.; GELADI, P. Principal Component Analysis. **Chemometrics and Intelligent Laboratory Systems**, Elsevier, v. 2, n. 1-3, p. 37–52, 1987.

XIE, S.; GIRSHICK, R.; DOLLÁR, P.; TU, Z.; HE, K. Aggregated residual transformations for deep neural networks. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2017. p. 1492–1500.

ZHANG, T.; SUEN, C. Y. A fast parallel algorithm for thinning digital patterns. **Commun. of the ACM**, ACM New York, NY, USA, v. 27, n. 3, p. 236–239, 1984.

ZHUANG, F.; QI, Z.; DUAN, K.; XI, D.; ZHU, Y.; ZHU, H.; XIONG, H.; HE, Q. A comprehensive survey on transfer learning. **Proceedings of the IEEE**, v. 109, n. 1, p. 43–76, 2021.

APPENDIX A — RESUMO EXTENDIDO

As assinaturas manuscritas são ainda um dos métodos de autenticação biometrica mais usados em processos legais, administrativos e bancários. O presente trabalho de mestrado procura contribuir com a automatização da verificação de assinaturas manuscritas com um enfoque *writer independent*, i.e. construindo um modelo de classificação só, que permita classificar pares de assinaturas como verdadeiras (se as duas assinaturas pertencem a mesma pessoa) o falsas (se uma das assinaturas foi feita por alguém tentando imitar a assinatura da pessoa de interesse) ao invés de criar um modelo de classificação específico para cada pessoa. Adicionalmente, o presente trabalho está focado na verificação offline, i.e. usando só imagens estáticas das assinaturas (digitalizadas usando um scanner após a assinatura for escrita), as quais não contém informação dinamica referente ao processo de escrita da assinatura como inclinação e pressão aplicada sobre a caneta, seqüência de tempo de assinatura, entre outros.

Um novo framework para extração de features é proposto baseado num conjunto de features obtidas com uma rede neural convolucional (CNN) pre-treinada chamada CLIP e um outro conjunto de *handcrafted features* com base no conceito de *Moving Least Squares*. No caso das features obtidas da CNN o autor propõe um novo pipeline para o cálculo das features que gera cortes das imagens e processa elas indepentemente para logo fazer um proceso de junção desses features pela média deles elemento a elemento. Dessa forma, o novo pipeline consegue extrair features da imagem completa independentemente da forma dela (retangular o quadrada) ou sua orientação (horizontal ou vertical).

As *handcrafted features* propostas pelo autor são geradas com base na ideia intuitiva de que curvas acentuadas e / ou interseções de traços são as partes mais difíceis de imitar quando se deseja forjar uma assinatura. As *handcrafted features* são obtidas após de um processo de binarização da imagem seguido por um processo de *skeletonization* sobre a imagem binarizada, o que visa obter uma representação de um pixel de largura da assinatura. Pequenas vizinhanças do skeleton da assinatura são ajustados a *weighted least-square lines*, o coeficiente r^2 é usado para medir a qualidade do ajuste (curvas acentuadas e interseções de traços estarão asociados con valores pequenos do coeficiente r^2). Finalmente, os valores do coeficiente r^2 obtidos para cada vizinhança são convertidos a vetores de features usando histogramas, os quais são invariantes a rotações e escalamentos.

As features obtidas a partir da CNN tem uma dimensionalidade de 1024D por cada imagem, considerando um par de imagens, finalmente é obtido um vetor de 2048D. No caso

das *handcrafted features*, histogramas com dimensionalidade de 75D são geradas a partir da junção de diferentes vetores gerados usando diferentes numeros de bins (5,10,15,20,25). A diferencia quadrática dos vetores de features (histogramas) de cada par de assinaturas é calculada obtendo um outro vetor de 75 dimensões, finalmente a distância L^2 desses mesmos vetores é calculada (1 dimensão só). No final as *handcrafted features* ficam com 226 dimensões que juntadas com as 2048 dimensões das features da CNN nos dá um total de 2274 dimensões.

As features finais de cada par de assinaturas são passadas para um classificador SVM (Support Vector Machine0) com o conjunto de hiperparâmetros padrão. Adicionalmente, o presente trabalho nota um possível bias num dataset de assinaturas amplamente utilizado na literatura e propõe modificações aos datasets atuais para gerar versões deles que incluem rotações e escalamentos ou a combinação deles (casos da vida real). Experimentos abrangentes sobre a resistência do método proposto ás mencionadas modificações foram feitos, assim como uma discussão do impato do número de exemplos usado para o treinamento do modelo nas métricas e uma avaliação do poder de geralização do modelo proposto quando o treinamento é realizado num dataset e o teste é realizado em outro diferente. Um *ablation study* é também realizado, para mostrar o impato de cada uma das features utilizadas (CNN e *handcrafted*) nos resultados e as métricas obtidas.

Os resultados mostram que o método proposto consegue melhorar o estado da arte num dos datasets (western style) usados enquanto mantem uma perfomance similar a outros métodos nos outros datasets (asian-style). Adicionalmente, o método proposto é resistente a rotações e escalamentos em contraste com outros métodos, o autor considera que as handcrafted features fornecem a propriedade de invariabilidade por causa de que as mencionadas featuras estão compostas por histogramas (invariantes a rotações e escalamentos).