Meta-learning for fast classifier adaptation to new users of Signature Verification systems

Luiz G. Hafemann, Robert Sabourin, *Member, IEEE,* and Luiz S. Oliveira.

***Abstract*—Offline Handwritten Signature verification presents a challenging Pattern Recognition problem, where only knowledge of the positive class is available for training. While classifiers have access to a few genuine signatures for training, during generalization they also need to discriminate forgeries. This is particularly challenging for *skilled* forgeries, where a forger practices imitating the user's signature, and often is able to create forgeries visually close to the original signatures. Most work in the literature address this issue by training for a surrogate objective: discriminating genuine signatures of a user and random forgeries (signatures from other users). In this work, we propose a solution for this problem based on meta-learning, where there are two levels of learning: a task-level (where a task is to learn a classifier for a given user) and a meta-level (learning across tasks). In particular, the meta-learner guides the adaptation (learning) of a classifier for each user, which is a lightweight operation that only requires genuine signatures. The meta-learning procedure learns what is common for the classification across different users. In a scenario where skilled forgeries from a subset of users are available, the meta-learner can guide classifiers to be discriminative of skilled forgeries even if the classifiers themselves do not use skilled forgeries for learning. Experiments conducted on the GPDS-960 dataset show improved performance compared to Writer-Independent systems, and achieve results comparable to state-of-the-art Writer-Dependent systems in the regime of few samples per user (5 reference signatures).**

***Index Terms*—Meta Learning, Signature Verification, Biometrics**

I. Introduction

Offline Handwritten signature verification remains a challenging problem in the presence of skilled forgeries, where the forger has access to the user's signature and practices imitating it [1]. This problem is particularly challenging since in a practical application scenario we cannot expect to have access to skilled forgeries for every user in the system for training the classifiers.

This problem is mainly addressed in three ways in the literature: (i) training a classifier for each user using a surrogate objective, where the negative samples are genuine signatures from other users (called *random forgeries* in this context) [2], [3], [4] (ii) training a one-class classifier for each user [5]; (iii) training a global, writer-independent classifier [6], [7], [8]. The first alternative (Writer Dependent (WD) classification) optimizes a surrogate objective, which therefore can be sub-optimal. The second alternative (one class Writer Dependent classification) is an appropriate formulation of the problem, but empirical results show that this approach performs worse than the first. A possible reason is that for signature verification tasks we normally have only a small number of samples per user, which makes it hard to estimate the support (or probability density) of the positive class. For instance, recent work considers a feature space in $\mathbb{R}^{2048}$, while the number of signatures from one individual can be as low as 1-5 in practical applications [1], [4]. Lastly, the third alternative (Writer Independent (WI) classification) alleviates the problem of a small number of samples per user by transforming the problem in a binary classification problem: comparing a query signature with a reference (template) signature, where the same classifier is used for all users [9], [7]. However, empirically these approaches also show worse performance than WD classification, at least when the number of signatures available for training (per user) is larger than 1 [4]. We hypothesize that a reason for this gap in performance is that the WI classifiers compare a query signature with a reference signature one at a time, while the WD classifiers are trained with multiple references at the same time, and therefore can better estimate the invariances in a person's signature (intra-class variation).

Considering different approaches, WD classification (alternative (i) above) shows better empirical performance [1]. However, this approach has other shortcomings compared to WI approaches: they require training a classifier for each user, which is not desirable in some scenarios: For instance, when the number of users is very large, and each user do not use the system often - many classifiers are trained but are almost never used. Also, in the cases where features are learned from data (e.g. [4]), if we want to change the feature representation, for instance by training with new data, it is not straightforward to incorporate the new features without re-training all WD classifiers in the system, while a global (WI) classifier would not require any extra step. WI systems also naturally handles the issue of adding more signatures to the reference set.

In this work, we propose to formulate the task as a meta-learning problem, inspired by the work of a Forensics Handwritten Expert: the expert acquires knowledge examining genuine signatures and forgeries from several people along his/her training and work experience. For a new case, along with knowledge of signatures from the individual, this previous experience is also used when analyzing a signature of interest.

The main contributions of this paper are:

L. G. Hafemann and R. Sabourin are with the Laboratoire d'imagerie, de vision et d'intelligence artificielle, École de technologie supérieure, Université du Québec, Montreal, Canada. (e-mail: luiz.gh@mailbox.org, robert.sabourin@etsmtl.ca)

L. S. Oliveira is with the Department of Informatics, Federal University of Parana, Curitiba, Brazil (e-mail:lesoliveira@inf.ufpr.br)

This work was supported by the Fonds de recherche du Québec - Nature et technologies (FRQNT), the CNPq grant #206318/2014-6 and by the grant RGPIN-2015-04490 to Robert Sabourin from the NSERC of Canada.

- We formulate the signature verification task as a meta-learning problem, considering a meta-learner that learns across-tasks (classification for specific individuals), that is subsequently adapted to a particular user in order to make a prediction on a query signature.
- We extend Model Agnostic Meta Learning (MAML) [10], to consider different loss functions during classifier adaptation and meta-learning, to address the issue of partial-knowledge during training.
- The resulting system is as scalable as a WI system (there is a single meta-model), but that is also adaptable for individual users with a lightweight operation (a few gradient descent steps). Additionally, contrary to other work that learns representations to train WD classifiers ([4]), not only the final classification layer is adapted to the new user, but the *feature representation* is also adapted.
- We evaluate the approach in four widely used datasets, achieving results comparable to state-of-the-art on the GPDS-960 dataset. Finally, we discuss the limitations of the approach, most notably the requirement of using data from a large number of users for training, and worse results when transferring the meta-learner to the other datasets. Code to reproduce the experiments can be found at https://github.com/luizgh/sigver.

The paper is organized as follows: section II reviews the related work on signature verification and meta-learning. Section III introduces the formulation of signature verification as a meta-learning problem, and the proposed algorithm. Section IV describes the experimental protocol, and section V presents and discusses our results. Finally, section VI concludes the paper.

II. Related Work

The objective of signature verification systems is to classify a query signature as being genuine (produced by the claimed individual), or a forgery (produced by another person). In the Pattern Recognition community, different forgeries are considered: Random forgeries - in which the forger has no knowledge of the user's signature, and use his signature instead; Simple forgeries - in which the forger knows the person's name, but not their signature; Skilled forgeries - where the forger has access to the user's signature, and practices imitating it. While the problem of distinguishing random and simple forgeries is relatively easy (i.e. low error rates in state-of-the-art classifiers), skilled forgeries still present a significant challenge for classification.

These systems can be broadly categorized as Writer-Dependent (WD, also called User-Dependent) and Writer-Independent (WI, also called User-Independent). For Writer-Dependent classifiers, we consider a dataset for each user $\{x, y\}_{i=1}^{n}$, where x are signatures, and y indicate whether they are genuine signatures from the user ($y = 1$) or random forgeries ($y = 0$) [2], [3], [4]. Some work consider one-class WD classifiers, in which only genuine signatures from the user are used for training (only $y = 1$) [5]. For WI classifiers, there are two main approaches: training a single classifier in a *dissimilarity* space, and *metric learning* approaches. In the first case, the training samples are difference of feature vectors: $|\phi(x_1) - \phi(x_2)|$, with $y = 1$ if both signatures are from the same user, and $y = 0$ otherwise [6], [7]. The metric learning approaches use a siamese network architecture [11], which takes two signatures (x_1, x_2) as input, and outputs a metric (distance) between them.

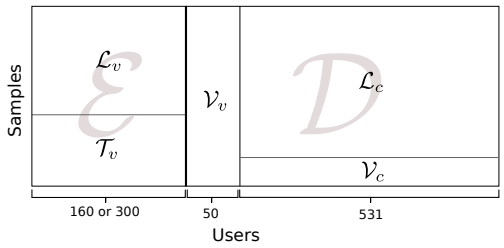


Fig. 1: Common dataset separation for Feature Learning followed by WD classification, on the GPDS dataset. Features are learned in $\mathcal{D}$. Model selection is conducted in $\mathcal{V}_v$. The system is evaluated by training WD classifiers for the exploitation set $\mathcal{E}$. [4]

Recent work on signature verification rely on feature learning methods [12], [4], [13], [14], [8], in which learning is conducted directly from signature pixels, instead of relying on handcrafted feature extractors. In this case, a function $\phi(x)$ is learned to *extract features* from signature images x, by training using a surrogate objective, e.g. dictionary learning [15], [14], or classifying the user that produce the signatures [4]. For instance, the SigNet model [4] is a Convolutional Neural Network trained with the following objective:

$$L = -\sum_{j} y_{ij} \log P(y_j|X_i) \tag{1}$$

Where X_i is a signature and y_i is the user that wrote the signature. Therefore, the network is trained to obtain a representation space where signatures from different people are linearly separable [4]. This feature representation is learned from a Development dataset $\mathcal{D}$, which is then used to extract features and train Writer-Dependent classifiers for a disjoint set of users (exploitation set $\mathcal{E}$) - a diagram of dataset separation is shown in Figure 1. While this approach achieved state-of-the-art verification performance, we note that the feature learning process does not directly optimize for the final objective of the system, which is distinguish genuine signatures and forgeries. This is addressed to some extent in the SigNet-F model, by also classifying whether or not the signature is a forgery. However, in that case, the neuron classifying forgeries does not use a reference signature from the user. While this was shown to be helpful in obtaining a good feature representation, this neuron did not generalize in classifying forgeries for unseen users [4].

Such methods using feature learning followed by WD classification also have other shortcomings: they require training one classifier for each user, which may be an expensive operation (e.g. best results in [12], [4] were reported with an SVM trained with the RBF kernel for each user). If the feature

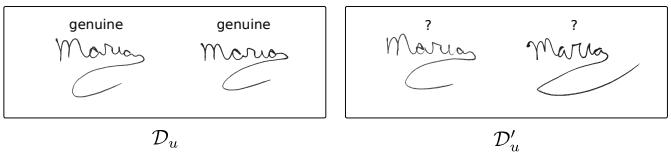


Fig. 2: Illustration of the data available for one task (user). Left: the reference (support) set. Right: query samples.

extractor is updated (e.g. trained with more data), then all classifiers need to be retrained. Also, these systems use a fixed representation for all users, and it is possible that adapting the representation for each user would yield improvements in classification performance.

It is also worth noting that, for WI classification, signature verification systems can be trained jointly (feature extraction and classification) [8]. Despite being jointly trained, such WI systems still perform worse than WD classifiers trained with features learned with surrogate objectives, at least when more than one signature references are used [4]. A possible reason for this gap is the fact that WI systems compare the query signature to each reference individually (or comparing with the centroid of the signatures), which is less powerful than training a classifier for the user, in capturing the invariances of the person's signature.

A. Meta-learning

In a broad sense, meta-learning is concerned with the problem of *learning to learn*, with origins in the 80's and 90's [16], [17]. More recently, algorithms based on meta-learning have achieved state-of-the-art results in tasks such as hyperparameter optimization [18], neural network architecture search [19]), and few-shot learning [20], [10]. Few-shot learning considers a scenario where only a few samples from each class are available for training, which is similar to actual application scenarios in handwritten signature verification.

The goal of these meta-learning approaches for few-shot learning is to train a model that can quickly (i.e. in a few iterations) adapt to a new task using only a few samples. A new task in this context refers, for instance, to classify a new object, for which only a few samples are known. Ravi and Larochelle [20] proposed learning an *optimizer* and *initialization* for the tasks (Meta Nets). They propose using a Long short-term memory (LSTM) model to learn the update rule for adapting the network parameters to a new task. Finn et al [10] proposed a Model Agnostic Meta Learning (MAML) procedure that does not require any extra parameters. This model optimizes the *sensitivity* of the weights, that is, obtain a feature representation that is highly adaptive, such that a single (or a few) gradient descent iterations are sufficient to optimize to new tasks.

III. Proposed Method

In this work we propose a meta-learning approach for signature verification. This formulation considers a meta-learner that guides the adaptation of a classifier for each user. We consider that each user describes a *task*: discriminating between genuine signatures (created by the user) and forgeries. Figure 2 illustrates the data available for one task: we consider a reference (support) dataset that is used for training a classifier that can classify new queries as genuine or forgery.

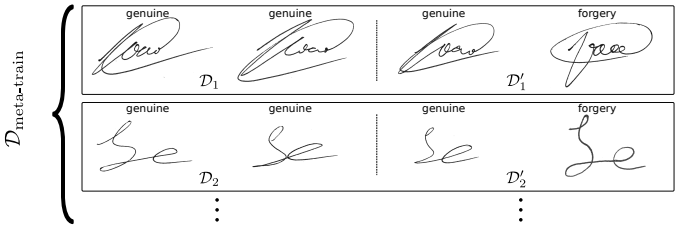


Fig. 3: Example of the meta-learning setup. Each user represents an *episode*, where $\mathcal{D}_u$ is used for classifier adaptation and $\mathcal{D}'_u$ is used for meta-update.

TABLE I: Table of symbols

$\mathcal{T}$	Distribution of tasks (i.e. users)
$\mathcal{T}_u$	Task for user u
$\mathcal{D}_{\text{meta-train}}$	Training set for the meta-learner
$\mathcal{D}_{\text{meta-test}}$	Testing set for the meta-learner
$\mathcal{D}_u$	Samples for weight adaptation for user u
$\mathcal{D}'_u$	Samples for meta-update for user u
G_u	Genuine signatures for user u
S_u	Skilled forgeries for user u
θ	Network parameters
$\theta_k^{(u)}$	Parameters adapted to user u after k descent steps
L	Loss function for weight adaptation
L'	Loss function for meta-update

In a meta-learning setting, we consider that training a classifier for a particular user is guided by a meta-learner, that leverages data from multiple tasks for learning. For this we consider a dataset $\mathcal{D}_{\text{meta-train}}$, and then evaluate the generalization performance on unseen users $\mathcal{D}_{\text{meta-test}}$.

We note that this approach has a direct correspondence to previous work that used feature learning followed by WD classification (section II), and here we make the association between the terminology in the meta-learning research and previous work on Signature Verification. In both cases we use a separate set of users for feature learning ($\mathcal{D}_{\text{meta-train}}$ is analogous to the development set in section II), which is then used for to train and test classifiers on a new set of users ($\mathcal{D}_{\text{meta-test}}$ is analogous to the exploitation set). The key differences of meta-learning is that: (i) The loss optimized for feature learning is directly related to the final objective (separate genuine signatures and forgeries); (ii) training a classifier for a new user is a lightweight process (a few gradient descent iterations); (iii) not only the classifier, but the features are also adapted for each user.

In the next section we formalize the problem of signature verification as a meta-learning task.

A. Problem formulation

We consider that each user describes a task $T_u \in \mathcal{T}$, where the task consists in classifying a signature image as genuine (created by the user) or forgery (not created by the user). A collection of users therefore describes a distribution of tasks $\mathcal{T}$, and the aim of the meta-learner is to explore the structure

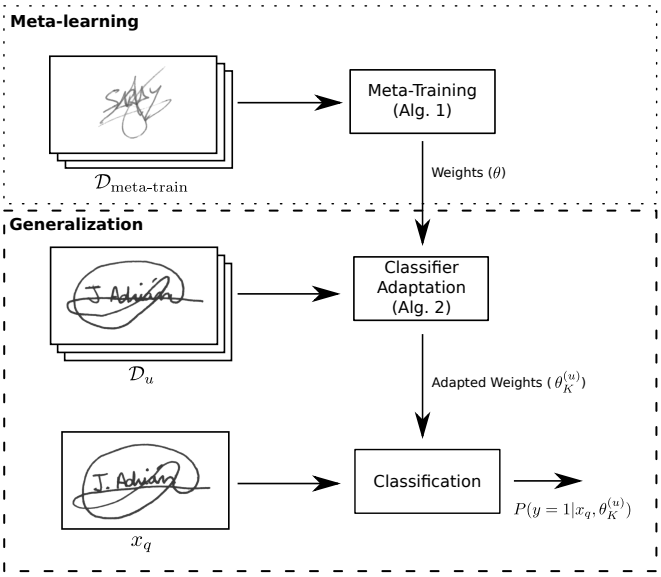


Fig. 4: Overview of the meta-learning system for signature verification.

Algorithm 1 Meta-Training algorithm

Input: M: Meta-batch size
Input: K: Number of gradient descent steps
Input: α, β Learning rates
Output: θ: Meta-learned weights

1: Randomly initialize θ
2: **while** not done **do**
3: Sample a batch of tasks $\{T_u\}_{u=1}^{M} \sim \mathcal{T}$
4: $\theta_{\text{grad}} \leftarrow \vec{0}$
5: **for** $u \leftarrow 1$ to M **do**
6: Sample $\mathcal{D}_u$ ▷ Genuine only
7: $\theta'_0 \leftarrow \theta$
8: **for** $k \leftarrow 1$ to K **do** ▷ Adapt weights to u
9: $\theta'_k \leftarrow \theta'_{k-1} - \alpha \nabla_{\theta'_{k-1}} L(\mathcal{D}_u, \theta'_{k-1})$
10: **end for**
11: Sample $\mathcal{D}'_u$ ▷ Genuine and forgeries
12: $\theta_{\text{grad}} \leftarrow \theta_{\text{grad}} + \frac{1}{M} \nabla_\theta L'(\mathcal{D}'_u, \theta'_K)$
13: **end for**
14: $\theta \leftarrow \theta - \beta \theta_{\text{grad}}$ ▷ Meta-update
15: **end while**

present in this distribution. We consider a dataset $\mathcal{D}_{\text{meta-train}}$ containing tasks from $\mathcal{T}$, that is used for meta-learning. For each user we consider a set $\mathcal{D}_u$, that is used to adapt the classifier, and a set $\mathcal{D}'_u$ that is used for updating the meta-learner. Lastly, to verify the generalization to unseen users, we consider a set $\mathcal{D}_{\text{meta-test}}$, that contains data from a disjoint set of users ($\mathcal{D}_{\text{meta-train}} \cap \mathcal{D}_{\text{meta-test}} = \emptyset$). Figure 3 illustrates the meta-learning setup, and the symbols used in this paper are listed in Table I for clarity.

B. Model Agnostic Meta-Learning for signature verification

In this work we propose an extended version of Model-Agnostic Meta-Learning (MAML) [10], by considering different criteria for classifier adaptation and meta-learning. An overview of the system can be seem in figure 4. We consider a development set for meta-training, that consists in learning the weights θ of a Convolutional Neural Network, that are highly *adaptable* to new tasks. During generalization, for a user u, a reference set $\mathcal{D}_u$ is used to adapt the classifier to this user (using K gradient descent steps) obtaining weights $\theta_K^{(u)}$. This adapted classifier is then used to classify a query image x_q, obtaining $P(y=1|x_q, \theta_K^{(u)})$.

Algorithm 1 describes the full meta-training algorithm. Meta-training is conducted in *episodes* (Figure 3). In each episode, the classifier is adapted to a particular user using $\mathcal{D}_u$ (lines 7 to 10), and the adapted classifier is used to classify the set $\mathcal{D}'_u$. The loss is then back-propagated through all intermediate steps of the classifier adaptation (lines 11 and 12), and is used to update the meta-learner weights θ (line 14). Therefore, instead of having a feature representation that is directly applicable for any user, they are learned to work well for new users *after* K gradient descent steps on the user's signatures. For stability during training, we train on "mini-batches" of episodes, by accumulating the gradients for M episodes before updating θ.

Figure 5 illustrates the classifier adaptation procedure. In this work, we adapt the MAML algorithm to use different loss functions for the classifier adaptation and the final loss (used for the meta-update). In particular, we consider a loss function L that only uses genuine signatures for the classifier adaptation, and a loss function L' that use both genuine signatures and forgeries. Let $\mathcal{D}_u = G_u \cup G_{i \neq u}$ be the training set consisted of genuine signatures from the user (G_u) and random forgeries ($G_{i \neq u}$). We consider the following loss for classifier adaptation:

$$\begin{aligned} L(\mathcal{D}_u, \theta) = &-\frac{1}{|G_u|} \sum_{x:G_u} \log(P(y|x,\theta)) \\ &-\frac{1}{|G_{i\neq u}|} \sum_{x:G_{i\neq u}} \log(P(y|x,\theta)) \end{aligned} \tag{2}$$

where $|G_u|$ and $|G_{i\neq u}|$ are the number of users in the sets, which is used to correct for the imbalance between the two classes.

Let $\mathcal{D'}_u = G'_u \cup G'_{i\neq u} \cup S'_u$ be the a disjoint set of signatures for user u: genuine signatures (G'_u), random forgeries ($G'_{i\neq u}$), and (if available), skilled forgeries S'_u. We define the loss function for meta-update as follows:

$$\begin{aligned} L'(\mathcal{D}'_u, \theta) = &-\frac{1}{|G'_u|} \sum_{x:G'_u} \log(P(y|x,\theta_K^{(u)})) \\ &-\frac{1}{|G'_{i\neq u}|} \sum_{x:G'_{i\neq u}} \log(P(y|x,\theta_K^{(u)})) \\ &-\frac{1}{|S'_u|} \sum_{x:S'_u} \log(P(y|x,\theta_K^{(u)})) \end{aligned} \tag{3}$$

On generalization, for a new user we first adapt the weights to this user using a set of reference signatures $\mathcal{D}_u$, and then classify a new query signature using the adapted weights. Algorithm 2 describes the classifier adaptation to a new user.

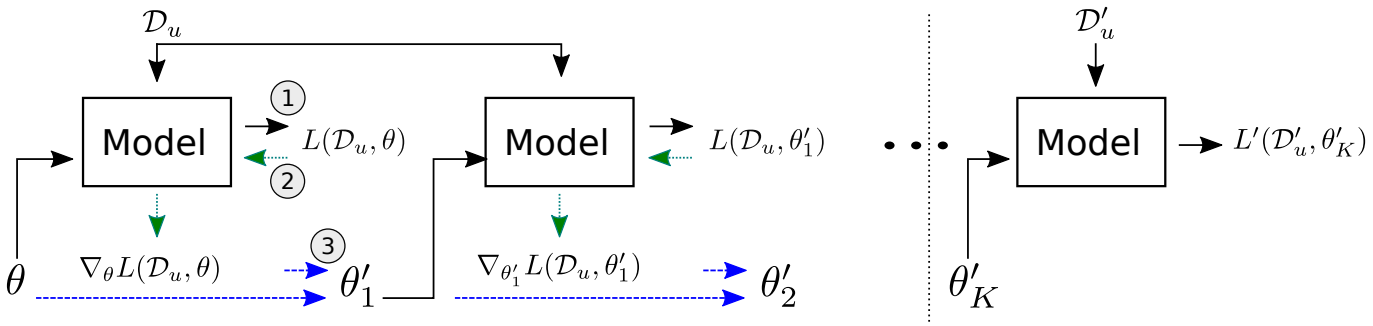


Fig. 5: Illustration of one iteration of meta-training for one task T_u. Starting with parameters θ, the weights are specialized for the task in K gradient descent steps. Each step involves computing the loss (1), back-propagating the loss w.r.t to θ'_{k-1} (2) and updating the weights (3). For the meta-update, the loss L' is backpropagated through the entire chain (from L' back to the initial θ), computing $\nabla_\theta L'(\mathcal{D}'_u, \theta^u_K)$.

Algorithm 2 Classifier adaptation

Input: K: Number of gradient descent steps
Input: α Learning rate
Input: θ Meta-learned weights
Input: $\mathcal{D}_u$ Reference set for user u
Output: θ'_K: Weights adapted to the user after K steps

1: $\theta'_0 \leftarrow \theta$
2: **for** $k \leftarrow 1$ to K **do**
3: $\quad \theta'_k \leftarrow \theta'_{k-1} - \alpha \nabla_{\theta'_{k-1}} L(\mathcal{D}_u, \theta'_{k-1})$
4: **end for**

We note that only the loss function L is used, and therefore only genuine signatures are used when adapting a classifier for a new user.

C. Meta-learning for one-class classification

The approach defined above can also be extended for one-class classification, where the classifier adaptation is done with only genuine signatures from the user of interest. This is easily implemented by considering $\mathcal{D}_u = G_u$. It is worth noting that similarity-based methods and one-class methods that involve feature learning often suffer from the problem of collapsing representations into a point [21]. This is often addressed by adding a penalty in the loss function that requires dissimilar items to be far apart in the feature space. In our formulation, while the user's classifier is only trained with data from one class, we observe that training does not collapse to a single point since the meta-training procedure directly optimizes the performance on separating forgeries in $\mathcal{D}'_u$.

IV. Experimental Protocol

We conducted most experiments on the GPDS-960 dataset [22], that consists of 881 users, with 24 genuine signatures per user and 30 skilled forgeries. We follow the same dataset separation as previous work (figure 1), with users 350-881 as $\mathcal{D}_{\text{meta-train}}$, 300-350 as $\mathcal{D}_{\text{meta-val}}$ and users 0-300 as $\mathcal{D}_{\text{meta-test}}$. We used the same pre-processing method from previous work [12], [4], by removing the background noise using OTSU, centering the images in a canvas of size 952×1360 and resizing them to 170×242.

We analyze the impact of the hyperparameters in the classifier's performance, measured in $\mathcal{D}_{\text{meta-val}}$. We consider the experiments by varying these parameters:

- Number of gradient descent steps in the classifier adaptation: $K \in \{1, 5\}$
- One-class classification vs adaptation using genuine signatures and random forgeries
- Fraction of users with skilled forgeries available for training
- Performance as we vary the number of reference genuine signatures

We compare the results on $\mathcal{D}_{\text{meta-val}}$ with a baseline using feature learning followed WD classification [4]. As in [4], we evaluate each model with repeated random subsampling: we randomly partition the validation set into training ($\mathcal{D}_u$) and testing ($\mathcal{D}'_u$), repeating the experiment 10 times with different partitions. We report the mean and standard deviation of the metrics.

In all experiments, we train the meta-classifier for a total of 100 epochs, considering a meta-batch size $M = 4$. We consider an initial meta-learning rate $\beta = 0.001$, that is reduced (with cosine annealing) to 10^{-5} by the last epoch. We used early stopping, by keeping the meta-learner weights that performed best in the validation set. Following [23], we used Multi-Step Loss Optimization (MSL) for improving training stability. For the first 20 epochs, instead of computing the loss function L' only after K steps (step 12 of algorithm 1), we compute the loss function for all intermediate θ'_k, and consider a weighted average of the losses. In the first epoch the loss using each θ'_k contributes equally to the loss function, and the weights are annealed to give more weight to the last step until iteration 20, after which only the loss function at the final step K contributes to the loss. We found this procedure effective in stabilizing training (measured by the variation in validation accuracy across epochs). We also attempted to use learnable task learning rates (LSLR) described in [23] without success. Empirically, we also noticed that when using only genuine signatures the task learning rate needs to be larger than the case where skilled forgeries are available for training. In our experiments, if the fraction of users with skilled forgeries is less than 10% we used a task learning rate $\alpha = 0.01$, and a

TABLE II: Base architecture used in this work

Layer	Size
Input	1x150x220
Convolution (C1)	32x5x5
Max Pooling	32x5x5
Convolution (C2)	32x5x5
Pooling	32x5x5
Fully Connected (FC3)	1024
Fully Connected (FC4)	256
Fully Connected + Sigmoid	1

learning rate of $\alpha = 0.001$ for the other experiments.

In order to evaluate the transferability of the features to other operating conditions, we conducted experiments on other datasets, (that were collected in different regions, and followed different collection processes): MCYT-75 [24], CEDAR [25] and Brazilian PUC-PR [26]. We conducted two experiments: (i) use the meta-learner trained on GPDS directly for new users of these datasets; (ii) train a meta-learner with data from the four datasets. It is worth noting that, with the exception of GPDS, the datasets are relatively small, with 55, 75 and 60 users for CEDAR, MCYT and Brazilian PUC-PR. We observed that the formulations from this work require a large amount of users for training, and for this reason, we conducted 10-fold cross validation. We divide each dataset in 10 folds (by users), and for each run we consider 1 fold as meta-test, and the remaining folders for meta-training and validation. As in the previous experiments, we further use repeated subsampling for evaluating the adaptation for the new users. In total, for experiment (ii), we train 10 CNN models and perform 10 adaptations for each user. We report the mean error rates over all runs, and the standard deviation across the 10 different adaptations (each based on different train/test splits of the repeated subsampling).

The CNN architecture used in the experiments is listed in table II. We found that using a smaller network, compared to previous work using feature learning followed by WD classification, was successful in the meta-learning setting. This network has a total of 1.4M weights and uses 0.1 GFLOPS for forward propagation, while SigNet [4] has 15.8M weights and uses 0.6 GFLOPS. That is, the CNN used in this work is 10x smaller and 6x times faster.

We evaluate the performance using the following metrics: False Rejection Rate (FRR): the fraction of genuine signatures rejected as forgeries; False Acceptance Rate ($\text{FAR}_{\text{random}}$ and $\text{FAR}_{\text{skilled}}$): the fraction of forgeries accepted as genuine (considering random forgeries and skilled forgeries). We also report the Equal Error Rate (EER): which is the error when FAR = FRR. We considered two forms of calculating the EER: $\text{EER}_{\text{global }\tau}$: using a global decision threshold and $\text{EER}_{\text{user }\tau}$: using user-specific decision thresholds. In both cases, to calculate the Equal Error Rate we only considered skilled forgeries. For FRR and FAR, we report the values with a threshold of 0.5 (i.e. if $p(y = 1|x, \theta'_K) \geq 0.5$ we consider the model predicting x as a genuine signature).

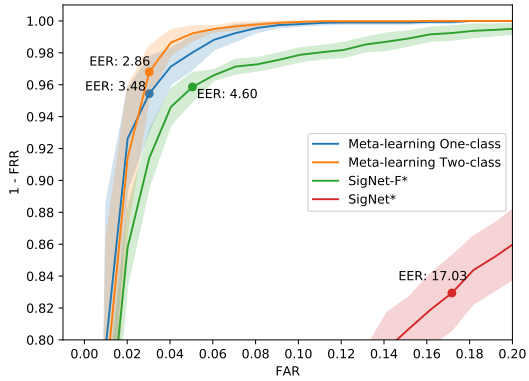


Fig. 6: ROC curves on $\mathcal{D}_{\text{meta-val}}$ comparing the one-class and two-class formulations with the baselines.

V. Results

A. System design

In this section we report the results on $\mathcal{D}_{\text{meta-val}}$ (GPDS users 300-350), considering the experiments defined in section IV. The objective is to evaluate different aspects of the system, such as the number of gradient steps (that trades-off computation complexity and accuracy), as well as investigate the performance of the model in different data scenarios.

In a first experiment we consider the results of the one-class formulation and the two-class formulation as we vary the number of Random Forgeries used for classifier adaptation (#RF). For this experiment, use used 5 genuine signatures for classifier adaptation, and $K = 5$ gradient descent steps; for meta-training we considered that skilled forgeries were available on $\mathcal{D}_{\text{meta-train}}$ (users 350-881). Note that for validation, no skilled forgeries were used for training. Table III reports the results of these experiments. We observe similar verification performance on the two formulations. Note that the formulation using random forgeries is more computationally expensive, as the classifier adaptation involves a larger batch of images (e.g. computing the loss for one-class uses 5 images, while for two-class with #RF=30 uses 35 images). We also compare with a method using feature learning followed by WD classification [4]. The entries denoted *SigNet** used the same approach proposed in [4], but using the CNN architecture defined for this work (table II). We note that the meta-learning formulation performed much better, while being a simpler model (single model for all users). A comparison with the SigNet CNN architecture from [4] is conducted in section V-B, where we compare to the state-of-the-art. Figure 6 presents ROC curves for the one-class formulation and the two-class formulation with #RF=20, as well as the two baselines. We consider average ROC curves over the 10 folds, where the solid line indicates the mean FRR for a given value of FAR, and the shaded area indicates one standard-deviation. Again, we see improved performance compared to the baselines.

Figure 7 shows the results on verification performance as we vary the number of gradient descent steps K. For each value of K, we meta-trained a network and evaluate its performance on $\mathcal{D}_{\text{meta-val}}$. We observed improved performance

TABLE III: Performance on $\mathcal{D}_{\text{meta-val}}$ with one-class and two-class formulations

Type	#Gen	#RF	FRR	$\text{FAR}_{\text{random}}$	$\text{FAR}_{\text{skilled}}$	$\text{EER}_{\text{global }\tau}$	$\text{EER}_{\text{user }\tau}$
SigNet* + WD	5	7434	10.48 (±2.24)	0.03 (±0.01)	24.67 (±0.99)	17.03 (±1.06)	13.17 (±0.94)
SigNet-F* + WD	5	7434	18.08 (±1.49)	0.16 (±0.04)	1.55 (±0.22)	4.6 (±0.59)	3.08 (±0.38)
Meta-learning One-class	5	-	2.54 (±0.61)	2.74 (±0.93)	4.24 (±0.93)	3.48 (±0.57)	1.69 (±0.43)
Meta-learning Two-class	5	5	2.82 (±0.59)	1.98 (±0.55)	4.18 (±0.48)	3.8 (±0.48)	2.04 (±0.41)
	5	10	5.1 (±0.99)	1.94 (±0.27)	2.66 (±0.76)	3.56 (±0.57)	1.85 (±0.57)
	5	20	2.84 (±0.97)	1.98 (±0.48)	3.1 (±0.83)	2.86 (±0.59)	1.78 (±0.27)
	5	30	2.62 (±0.82)	2.48 (±0.39)	3.46 (±0.62)	3.17 (±0.37)	1.4 (±0.48)

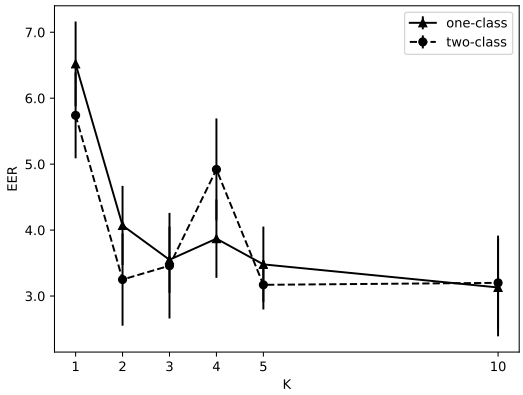


Fig. 7: Performance on $\mathcal{D}_{\text{meta-val}}$ as we vary the number of update steps K.

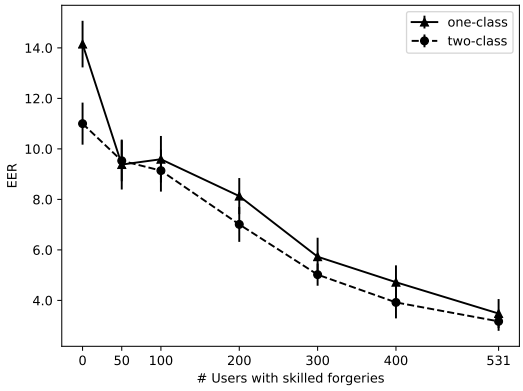
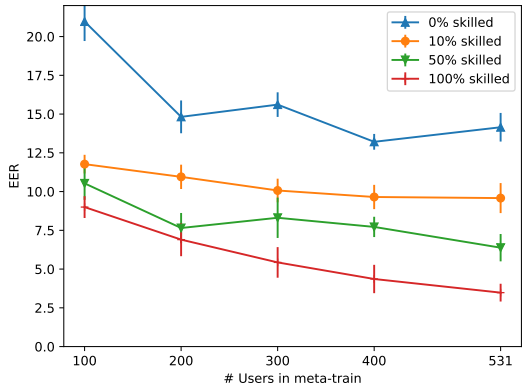
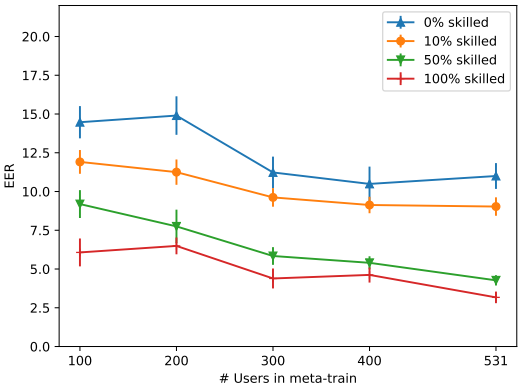


Fig. 8: Performance on $\mathcal{D}_{\text{meta-val}}$ as we vary the number of users in $\mathcal{D}_{\text{meta-train}}$ for which skilled forgeries are available.



(a) One-class



(b) Two-class

Fig. 9: Performance on $\mathcal{D}_{\text{meta-val}}$ as we vary the number of users available for meta-training. (a): one-class formulation; (b) two-class formulation.

with larger number of steps, but with diminishing returns. We note a high variance of the errors in these experiments, and therefore we cannot determine a particular K as being optimal. As we increase the number of steps, we also increase the computational cost. If we consider that forward propagation and backward propagation have similar cost, the classifier adaptation for a new user takes $2K$ the time for a single forward pass. A higher K also requires more memory (in the order of $2K$) during meta-training, since the whole update sequence needs to be stored in memory in order to compute the gradient for meta-update (as can be seen in figure 5).

In figures 8 and 9 we analyze the impact in performance as we vary the size of the $\mathcal{D}_{\text{meta-train}}$ set. As noted in section III-B, if skilled forgeries from a subset of users are available, we can incorporate them into the meta-update loss function L'. In this experiment we considered that $\mathcal{D}_{\text{meta-train}}$ contains all 531 users, and vary the number of users for which skilled forgeries are available. For each case, we build a dataset consisting of genuine signatures for all users and skilled forgeries for the selected users, and trained a model. Figure 8 shows the performance as we vary the number of users for which skilled forgeries as available. We re-iterate that we evaluate the performance on a disjoint set of users ($\mathcal{D}_{\text{meta-val}}$) for which only genuine signatures are used. We observed that

TABLE IV: Comparison with state-of-the art on the GPDS dataset (errors in %)

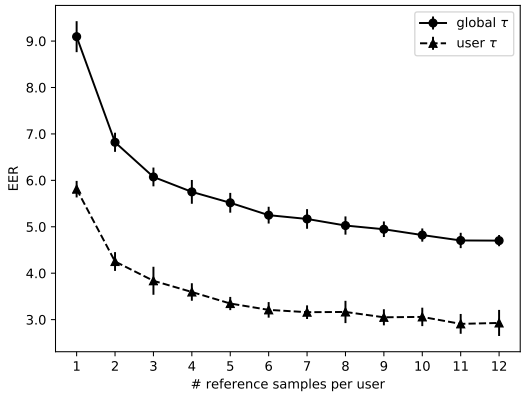
Reference	Type	Dataset	#samples per user	Features	EER
Hu and Chen [27]	WI	GPDS-150	10	LBP, GLCM, HOG	7.66
Guerbai et al [5]	WD	GPDS-160	12	Curvelet transform	15.07
Serdouk et al [28]	WD	GPDS-100	16	GLBP, LRF	12.52
Yilmaz [3]	WD	GPDS-160	5	LBP, HOG, SIFT	7.98
Yilmaz [3]	WD	GPDS-160	12	LBP, HOG, SIFT	6.97
Soleimani et al [29]	WI	GPDS-300	10	LBP	20.94
Hafemann et al [4]	WD	GPDS-300	5	SigNet-F (global τ)	5.25 ($\pm$0.15)
Hafemann et al [4]	WD	GPDS-300	5	SigNet-F (user τ)	2.42 ($\pm$0.24)
Hafemann et al [4]	WD	GPDS-300	12	SigNet-F (global τ)	3.74 ($\pm$0.15)
Hafemann et al [4]	WD	GPDS-300	12	SigNet-F (user τ)	1.69 ($\pm$0.18)
Souza et al [30]	WI	GPDS-300	5	SigNet (global τ)	9.05 ($\pm$0.34)
Souza et al [30]	WI	GPDS-300	5	SigNet (user τ)	4.40 ($\pm$0.34)
Souza et al [30]	WI	GPDS-300	12	SigNet (global τ)	7.96 ($\pm$0.26)
Souza et al [30]	WI	GPDS-300	12	SigNet (user τ)	3.34 ($\pm$0.22)
Present work	WI/WD	GPDS-300	5	MAML one-class (global τ)	5.52 ($\pm$0.20)
Present work	WI/WD	GPDS-300	5	MAML one-class (user τ)	3.35 ($\pm$0.13)
Present work	WI/WD	GPDS-300	5	MAML two-class (global τ)	5.16 ($\pm$0.19)
Present work	WI/WD	GPDS-300	5	MAML two-class (user τ)	2.94 ($\pm$0.20)
Present work	WI/WD	GPDS-300	12	MAML one-class (global τ)	4.70 ($\pm$0.11)
Present work	WI/WD	GPDS-300	12	MAML one-class (user τ)	2.93 ($\pm$0.27)
Present work	WI/WD	GPDS-300	12	MAML two-class (global τ)	4.39 ($\pm$0.18)
Present work	WI/WD	GPDS-300	12	MAML two-class (user τ)	2.68 ($\pm$0.17)

the meta-learning formulation of the problem is well suited to incorporating information from skilled forgeries (when it is available), and this generalizes well to unseen users, for which we only have genuine signatures. However, we observed that the performance is not very good when there are only genuine signatures for meta-training: the one-class formulation achieves 14.15% EER when only genuine signatures are available, and 3.48% EER when skilled forgeries are available for all 531 users in meta-training.

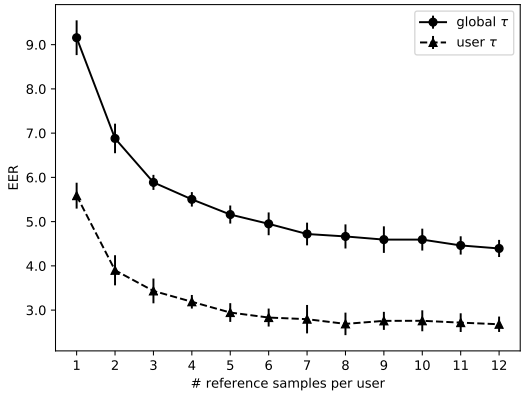
In figure 9, we evaluate the performance of the system as we vary the number of users in $\mathcal{D}_{\text{meta-train}}$. We also consider 4 levels of availability of skilled forgeries in the meta-training set: 0% (genuine only), 10%, 50% and 100%, where the percentages refer to the number of users for which skilled forgeries are available (e.g. 10% with 100 users means that forgeries for 10 users are considered, where the remaining 90 users have only genuine signatures). For a given number of users and skilled forgery percentage, we construct a dataset with randomly selected users (taken from the 531 users in the development set), with genuine signatures from all the selected users, and skilled forgeries for a fraction of the users. We then use this dataset for meta-training a model, and evaluate its performance on $\mathcal{D}_{\text{meta-val}}$. We observed improved performance both as more users are available for meta-training, as well as when more knowledge of skilled forgeries is available. Most surprisingly, we observed that for the two-class formulation, a classifier trained with 100 users with 100% forgeries (i.e. forgeries for every user in meta-train) performed better than a model trained with 531 users with forgeries for only 100 users (comparing figures 9b and 8): 6.07% EER vs 9.14% EER. We re-iterate that this measures the performance on discriminate genuine signatures and skilled forgeries, and the model that has access to more users (with the same amount of users with skilled forgeries) has better performance on discriminating random forgeries, since its optimization consisted mostly of this problem.

B. Comparison with the state-of-the-art

We now compare our results with the state-of-the-art in the GPDS-300 dataset. For these comparisons, we considered a model trained with the one-class formulation, and a model trained with the two-class formulation, with $r = 30$ forgeries. In both cases, we used the whole dataset $\mathcal{D}_{\text{meta-train}}$ for training the meta-classifier, and used 5 genuine signatures for classifier adaptation, with $k = 5$ updates. While training was conducted with 5 reference signatures, we evaluate the performance of the system with different number of references.



(a) One-class



(b) Two-class

Fig. 10: Performance on GPDS-300 as we vary the number reference signatures available for each user. (a): one-class formulation; (b) two-class formulation.

Table IV compares our results with the state-of-the-art. We observe an improved performance compared to other WI systems, achieving 5.16% EER (global τ) with 5 reference signatures, compared to 9.05% from [30]. Comparing to WD systems, we observed similar performance in some scenarios (5 reference signatures), and worse results otherwise. With 12 reference signatures, the proposed system obtained 4.39% EER (global τ), compared to 3.74 for the WD system [4]. However, the proposed system is more scalable, as a single model is stored for all users.

Figure 10 shows the performance on GPDS-300 as we vary the number of reference samples available for each user. As commonly observed in WD systems (e.g. [4]), the performance greatly improves as more reference samples are available for training: For the one-class formulation, performance with a single reference is 9.09% EER (global τ) and 5.81% EER (user τ). With 12 references, we obtain 4.70% EER (global τ) and 2.93% EER (user τ).

TABLE V: Transfer performance to the other datasets

Target Dataset	Training Dataset	EER (global)	EER (user)
MCYT	GPDS	15.48 (± 1.00)	12.54 (± 1.86)
	All	15.37 (± 0.97)	12.77 (± 0.46)
CEDAR	GPDS	15.98 (± 1.09)	12.07 (± 1.01)
	All	10.69 (± 1.76)	8.02 (± 1.22)
Brazilian	GPDS	8.05 (± 0.95)	4.83 (± 1.07)
	All	8.55 (±0.55)	6.7 (± 0.87)

TABLE VI: Comparison with the state-of-the-art in MCYT

Reference	Type	# Samples	Features	EER
Wen et al.[31]	WD	5	RPF	15.02
Vargas et al.[32]	WD	5	LBP	11.9
Vargas et al.[32]	WD	10	LBP	7.08
Ooi et al[33]	WD	5	DRT + PCA	13.86
Ooi et al[33]	WD	10	DRT + PCA	9.87
Soleimani et al.[29]	WD	5	HOG	13.44
Soleimani et al.[29]	WD	10	HOG	9.86
Hafemann et al.[4]	WD	5	SigNet (user τ)	3.58 (± 0.54)
Hafemann et al.[4]	WD	10	SigNet (user τ)	2.87 (± 0.42)
Present Work	WI/WD	5	MAML one-class (global τ)	15.37(± 0.97)
Present Work	WI/WD	5	MAML one-class (user τ)	12.77(± 0.46)
Present Work	WI/WD	10	MAML one-class (global τ)	14.50(± 0.77)
Present Work	WI/WD	10	MAML one-class (user τ)	12.44(± 0.97)

C. Transfer to other datasets

We now consider results on three other datasets: MCYT, CEDAR and the Brazilian PUC-PR. Table V shows the performance in two scenarios: (i) meta-learner trained only in GPDS, with its generalization to new operating conditions and (ii) meta-learned trained on all four datasets (using 10-fold cross validation, as described in section IV). While the method generalized well to unseen GPDS users, we see that the generalization performance to other datasets is much worse. Furthermore, we notice that even when training with a subset of users from all datasets, the performance does not improve for all datasets. A possible explanation is that the GPDS dataset is still much larger (10 times larger than the others) and dominates training. Overall, this suggests that the proposed method requires a large amount of data from the target application, and is sensitive to changes in operating conditions. Finally, tables VI, VII and VIII compares de results with the state-of-the-art on MCYT, CEDAR and Brazilian PUC-PR, respectively.

TABLE VII: Comparison with the state-of-the-art in CEDAR

Reference	Type	# Samples	Features	AER/EER
Chen and Srihari[34]	WD	16	Graph Matching	7.9
Kumar et al.[35]	WI	1	morphology	11.81
Kumar et al.[6]	WI	1	Surroundness	8.33
Bharathi and Shekar[36]	WD	12	Chain code	7.84
Guerbai et al.[5]	WD	4	Curvelet transform	8.7
Guerbai et al.[5]	WD	8	Curvelet transform	7.83
Guerbai et al.[5]	WD	12	Curvelet transform	5.6
Hafemann et al.[4]	WD	4	SigNet (SVM)	5.87 (± 0.73)
Hafemann et al.[4]	WD	8	SigNet (SVM)	5.03 (± 0.75)
Present Work	WI/WD	4	MAML one-class (global τ)	11.06(± 1.12)
Present Work	WI/WD	4	MAML one-class (user τ)	8.27(± 1.45)
Present Work	WI/WD	8	MAML one-class (global τ)	10.21(± 1.21)
Present Work	WI/WD	8	MAML one-class (user τ)	7.07(± 1.08)

TABLE VIII: Comparison with the state-of-the-art on the Brazilian PUC-PR dataset (errors in %)

Reference	Type	#samples	Features	$\text{AER}_{\text{genuine + skilled}}$/EER
Bertolini et al. [37]	WI	15	Graphometric	8.32
Batista et al. [38]	WD	30	Pixel density	10.5
Rivard et al. [9]	WI	15	ESC + DPDF	11.08
Eskander et al. [7]	WD	30	ESC + DPDF	10.67
Hafemann et al.[4]	WD	5	SigNet (user τ)	2.92 (± 0.44)
Hafemann et al.[4]	WD	15	SigNet (user τ)	2.07 (± 0.63)
Souza et al.[30]	WI	5	SigNet (global τ)	5.95 (± 0.68)
Souza et al.[30]	WI	5	SigNet (user τ)	2.58 (± 0.72)
Souza et al.[30]	WI	15	SigNet (global τ)	5.13 (± 0.23)
Souza et al.[30]	WI	15	SigNet (user τ)	1.70 (± 0.40)
Present Work	WI/WD	5	MAML one-class (global τ)	8.55 (± 0.55)
Present Work	WI/WD	5	MAML one-class (user τ)	6.70(± 0.87)
Present Work	WI/WD	15	MAML one-class (global τ)	6.93(± 0.73)
Present Work	WI/WD	15	MAML one-class (user τ)	5.74(± 0.84)

It is worth noting that the meta-learning does generalize to *new users* of the GPDS dataset, as verified in sections V-A and V-B, since we evalute in a $\mathcal{D}_{\text{meta-test}}$ that contains a disjoint set of users that was used to train the meta-learner. What we observed, however, is that this meta-learned does not transfer well to other datasets. This has been observed more recent work with meta-learning [39], that shows that although these models perform well for new classes of the same distribution (e.g. same dataset), the performance deteriorates when evaluating on new datasets (i.e. a shift in the task-distribution). This is still an active area of research in meta-learning.

VI. Conclusion

In this paper we proposed to formulate Signature Verification as a meta-learning problem, where each user defines a task. This formulation enables directly optimizing for the objective (separating genuine signatures and forgeries) even when forgeries are not available for all users. The resulting system is scalable and yet adaptable for individual users: a single meta-classifier is learned and stored, and for the verification of a given signature, the classifier is adapted to the claimed user and subsequently used for verification. The proposed method is also able to naturally incorporate new reference signatures for a user, and enable adapting the representation as more training data is available. The drawbacks of this solution are twofold: increased computational cost and worse transferability to new conditions. The method is $2K$ slower, when using K updates for the classification adaptation, although it allows the option to trade storage and computational cost - the adapted weights for a given user can be stored for faster classification.

In our experiments with the GPDS-960 dataset, the proposed method obtains better results than WI systems in the literature, and approach the performance of WD systems, especially when few samples are available for training. With 5 reference signatures, the proposed method obtains 5.16% EER (using a global threshold), compared to 9.05% of a WI system and 5.25% of a WD system. For a larger number of references the WD system still performs better, but the gap in performance is greatly reduced. Considering 12 reference signatures, the method obtains 4.39% EER (with a global threshold), vs 3.74% for the WD system, while being more scalable (single meta-classifier) Our experiments transferring the meta-learner to other datasets show reduced performance, highlighting the need for better adaptation to new conditions, which will be explored in future work. Future work also includes considering a dynamic scenario, where the meta-classifier is updated as new training data is available.

References

[1] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Offline handwritten signature verification — Literature review," in *2017 Seventh Interna-*

tional Conference on Image Processing Theory, Tools and Applications (IPTA), Nov. 2017, pp. 1–8.

[2] J. Vargas, C. Travieso, J. Alonso, and M. Ferrer, "Off-line Signature Verification Based on Gray Level Information Using Wavelet Transform and Texture Features," in *2010 International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Nov. 2010, pp. 587–592.

[3] M. B. Yılmaz and B. Yanıkoğlu, "Score level fusion of classifiers in off-line signature verification," *Information Fusion*, vol. 32, Part B, pp. 109–119, Nov. 2016.

[4] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Learning features for offline handwritten signature verification using deep convolutional neural networks," *Pattern Recognition*, vol. 70, pp. 163–176, Oct. 2017.

[5] Y. Guerbai, Y. Chibani, and B. Hadjadji, "The effective use of the one-class SVM classifier for handwritten signature verification based on writer-independent parameters," *Pattern Recognition*, vol. 48, no. 1, pp. 103–113, Jan. 2015.

[6] R. Kumar, J. D. Sharma, and B. Chanda, "Writer-independent off-line signature verification using surroundedness feature," *Pattern Recognition Letters*, vol. 33, no. 3, pp. 301–308, Feb. 2012.

[7] G. Eskander, R. Sabourin, and E. Granger, "Hybrid writer-independent-writer-dependent offline signature verification system," *IET Biometrics*, vol. 2, no. 4, pp. 169–181, Dec. 2013.

[8] H. Rantzsch, H. Yang, and C. Meinel, "Signature embedding: Writer independent offline signature verification with deep metric learning," in *Advances in Visual Computing*, ser. Lecture Notes in Computer Science. Springer International Publishing, pp. 616–625, DOI: 10.1007/978-3-319-50832-0_60.

[9] D. Rivard, E. Granger, and R. Sabourin, "Multi-feature extraction and selection in writer-independent off-line signature verification," *International Journal on Document Analysis and Recognition*, vol. 16, no. 1, pp. 83–103, 2013.

[10] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70. International Convention Centre, Sydney, Australia: PMLR, 06–11 Aug 2017, pp. 1126–1135.

[11] J. Bromley, I. Guyon, Y. LeCun, E. Siickinger, and R. Shah, "Signature Verification using a "Siamese" Time Delay Neural Network," 1994.

[12] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Writer-independent feature learning for offline signature verification using convolutional neural networks," in *Neural Networks, The 2016 International Joint Conference on*, 2016.

[13] L. G. Hafemann, L. S. Oliveira, and R. Sabourin, "Fixed-sized representation learning from offline handwritten signatures of different sizes," *International Journal on Document Analysis and Recognition (IJDAR)*, pp. 1–14, 2018.

[14] E. N. Zois, M. Papagiannopoulou, D. Tsourounis, and G. Economou, "Hierarchical Dictionary Learning and Sparse Coding for Static Signature Verification," p. 11.

[15] E. N. Zois, I. Theodorakopoulos, D. Tsourounis, and G. Economou, "Parsimonious Coding and Verification of Offline Handwritten Signatures," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jul. 2017, pp. 636–645.

[16] J. Schmidhuber, "Evolutionary principles in self-referential learning," PhD Thesis, Technische Universität München, München, 1987.

[17] Y. Bengio, S. Bengio, and J. Cloutier, "Learning a synaptic learning rule," in *IJCNN-91-Seattle International Joint Conference on Neural Networks*, vol. ii, Jul. 1991, pp. 969 vol.2–.

[18] D. Maclaurin, D. Duvenaud, and R. Adams, "Gradient-based hyperparameter optimization through reversible learning," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 2113–2122.

[19] B. Baker, O. Gupta, N. Naik, and R. Raskar, "Designing Neural Network Architectures using Reinforcement Learning," in *International Conference on Learning Representations*, 2017.

[20] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," 2016.

[21] P. Perera and V. M. Patel, "Learning Deep Features for One-Class Classification," *arXiv:1801.05365 [cs]*, Jan. 2018.

[22] J. Vargas, M. Ferrer, C. Travieso, and J. Alonso, "Off-line Handwritten Signature GPDS-960 Corpus," in *Document Analysis and Recognition, 9th International Conference on*, vol. 2, Sep. 2007, pp. 764–768.

[23] A. Antoniou, H. Edwards, and A. Storkey, "How to train your MAML," in *International Conference on Learning Representations*, 2019.

[24] J. Ortega-Garcia, J. Fierrez-Aguilar, D. Simon, J. Gonzalez, M. Faundez-Zanuy, V. Espinosa, A. Satue, I. Hernaez, J.-J. Igarza, C. Vivaracho, and others, "MCYT baseline corpus: a bimodal biometric database," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 150, no. 6, pp. 395–401, 2003.

[25] M. K. Kalera, S. Srihari, and A. Xu, "Offline signature verification and identification using distance statistics," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 18, no. 07, pp. 1339–1360, Nov. 2004.

[26] C. Freitas, M. Morita, L. Oliveira, E. Justino, A. Yacoubi, E. Lethelier, F. Bortolozzi, and R. Sabourin, "Bases de dados de cheques bancarios brasileiros," in *XXVI Conferencia Latinoamericana de Informatica*, 2000.

[27] J. Hu and Y. Chen, "Offline Signature Verification Using Real Adaboost Classifier Combination of Pseudo-dynamic Features," in *Document Analysis and Recognition, 12th International Conference on*, Aug. 2013, pp. 1345–1349.

[28] Y. Serdouk, H. Nemmour, and Y. Chibani, "New gradient features for off-line handwritten signature verification," in *2015 International Symposium on Innovations in Intelligent SysTems and Applications (INISTA)*, Sep. 2015, pp. 1–4.

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

[30] V. L. F. Souza, A. L. I. Oliveira, and R. Sabourin, "A Writer-Independent Approach for Offline Signature Verification using Deep Convolutional Neural Networks Features," in *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*, Oct. 2018, pp. 212–217.

[31] J. Wen, B. Fang, Y. Y. Tang, and T. Zhang, "Model-based signature verification with rotation invariant features," *Pattern Recognition*, vol. 42, no. 7, pp. 1458–1466, Jul. 2009.

[32] J. F. Vargas, M. A. Ferrer, C. M. Travieso, and J. B. Alonso, "Off-line signature verification based on grey level information using texture features," *Pattern Recognition*, vol. 44, no. 2, pp. 375–385, Feb. 2011.

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

[34] S. Chen and S. Srihari, "A New Off-line Signature Verification Method based on Graph," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 2, 2006, pp. 869–872.

[35] R. Kumar, L. Kundu, B. Chanda, and J. D. Sharma, "A Writer-independent Off-line Signature Verification System Based on Signature Morphology," in *Proceedings of the First International Conference on Intelligent Interactive Technologies and Multimedia*, ser. IITM '10. New York, NY, USA: ACM, 2010, pp. 261–265.

[36] R. Bharathi and B. Shekar, "Off-line signature verification based on chain code histogram and Support Vector Machine," in *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Aug. 2013, pp. 2063–2068.

[37] D. Bertolini, L. S. Oliveira, E. Justino, and R. Sabourin, "Reducing forgeries in writer-independent off-line signature verification through ensemble of classifiers," *Pattern Recognition*, vol. 43, no. 1, pp. 387–396, Jan. 2010.

[38] L. Batista, E. Granger, and R. Sabourin, "Dynamic selection of generative–discriminative ensembles for off-line signature verification," *Pattern Recognition*, vol. 45, no. 4, pp. 1326–1340, Apr. 2012.

[39] E. Triantafillou, T. Zhu, V. Dumoulin, P. Lamblin, K. Xu, R. Goroshin, C. Gelada, K. Swersky, P.-A. Manzagol, and H. Larochelle, "Meta-dataset: A dataset of datasets for learning to learn from few examples," *arXiv preprint arXiv:1903.03096*, 2019.

Luiz G. Hafemann received his B.S. degree in Computer Science in 2008 and his M.Sc. degree in Informatics in 2014, both from the Federal University of Paraná, Curitiba, PR, Brazil. He received his Ph.D. degree in Systems Engineering in 2019 from the École de Technologie Supérieure, Université du Québec, in Montreal, QC, Canada. He is currently a researcher at Sportlogiq, applying computer vision models for sports analytics. His current interests include meta-learning, adversarial machine learning and group activity recognition.

Luiz S. Oliveira received his B.S. degree in Computer Science from Unicenp, Curitiba, PR, Brazil, the M.Sc. degree in electrical engineering and industrial informatics from the Centro Federal de Educação Tecnológica do Paraná (CEFET-PR), Curitiba, PR, Brazil, and Ph.D. degree in Computer Science from École de Technologie Supérieure, Université du Québec in 1995, 1998 and 2003, respectively. From 2004 to 2009 he was professor of the Computer Science Department at Pontifical Catholic University of Paraná, Curitiba, PR, Brazil. In 2009, he joined the Federal University of Paraná, Curitiba, PR, Brazil, where he is professor of the Department of Informatics and head of the Graduate Program in Computer Science. His current interests include Pattern Recognition, Machine Learning, Image Analysis, and Evolutionary Computation.

Robert Sabourin joined the physics department of the Montreal University in 1977 where his main contribution was the design and the implementation of a microprocessor-based fine tracking system combined with a low-light level CCD detector. In 1983, he joined the staff of the École de Technologie Supérieure, Université du Québec, in Montreal, where he co-founded the Dept. of Automated Manufacturing Engineering where he is currently Full Professor and teaches Pattern Recognition, Evolutionary Algorithms, Neural Networks and Fuzzy Systems. In 1992, he joined also the Computer Science Department of the Pontificia Universidade Católica do Paraná (Curitiba, Brazil). Since 1996, he is a senior member of the Centre for Pattern Recognition and Machine Intelligence (CENPARMI, Concordia University). Since 2012, he is the Research Chair holder specializing in Adaptive Surveillance Systems in Dynamic Environments. Dr. Sabourin is the author (and co-author) of more than 450 scientific publications including journals and conference proceedings. His research interests are in the areas of adaptive biometric systems, adaptive classification systems in dynamic environments, dynamic classifier selection and evolutionary computation.