MetaHTR: Towards *Writer-Adaptive* Handwritten Text Recognition

Ayan Kumar Bhunia[1] Shuvozit Ghose* Amandeep Kumar* Pinaki Nath Chowdhury[1,2]
Aneeshan Sain[1,2] Yi-Zhe Song[1,2]
[1]SketchX, CVSSP, University of Surrey, United Kingdom.
[2]iFlyTek-Surrey Joint Research Centre on Artificial Intelligence.
{a.bhunia, p.chowdhury, a.sain, y.song}@surrey.ac.uk.
{shuvozit.ghose, kumar.amandeep015}@gmail.com.

Abstract

Handwritten Text Recognition (HTR) remains a challenging problem to date, largely due to the varying writing styles that exist amongst us. Prior works however generally operate with the assumption that there is a limited number of styles, most of which have already been captured by existing datasets. In this paper, we take a completely different perspective – we work on the assumption that there is always a new style that is drastically different, and that we will only have very limited data during testing to perform adaptation. This creates a commercially viable solution – being exposed to the new style, the model has the best shot at adaptation, and the few-sample nature makes it practical to implement. We achieve this via a novel meta-learning framework which exploits additional new-writer data via a support set, and outputs a writer-adapted model via single gradient step update, all during inference (see Figure 1). We discover and leverage on the important insight that there exists few key characters per writer that exhibit relatively larger style discrepancies. For that, we additionally propose to meta-learn instance specific weights for a character-wise cross-entropy loss, which is specifically designed to work with the sequential nature of text data. Our writer-adaptive MetaHTR framework can be easily implemented on the top of most state-of-the-art HTR models. Experiments show an average performance gain of 5-7% can be obtained by observing very few new style data ($\leq$ 16).

1. Introduction

Handwritten Text Recognition (HTR) has been a long-standing research problem in computer vision [6, 35, 47, 29]. As a fundamental means of communication, handwritten text can appear in a variety of forms such as memos, whiteboards, handwritten notes, stylus-input, postal automation, reading aid for visually handicapped, etc [49]. In general, the target of automatic HTR is to transcribe handwritten text to its digital content [40] so that the textual content can be made freely accessible.

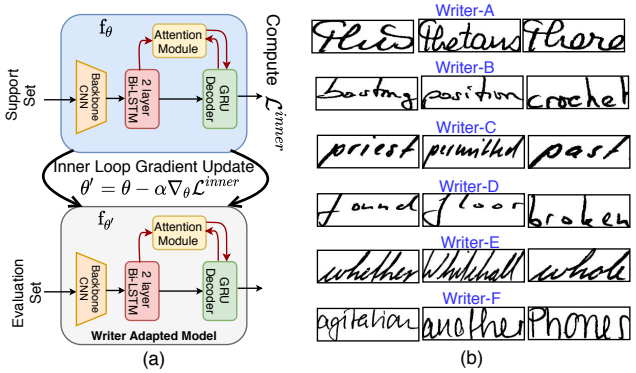


Figure 1. (a) During inference, our MetaHTR framework exploits additional handwritten images of a specific writer through support set, and gives rise to a writer adapted model via single gradient step update. (b) Varying styles across different writers (IAM).

Handwriting recognition is inherently difficult due to its free flowing nature and complex shapes assumed by characters and their combinations [6]. Torn pages, and warped or touching lines also make HTR more challenging. Most importantly however, handwritten texts are diverse across individual handwritten styles where each style can be very unique [13, 23] – while some might prefer an idiosyncratic style of writing certain characters like 'G' and 'Z', others may choose a cursive style with uneven line-spacing.

Modern deep learning based HTR models [28, 40, 26] mostly tackle these challenges by resourcing to a large amount of training data. The hope is that most style variations would have already been captured because of the data volume. Albeit with limited success, it had become apparent that these models tend to over-fit to styles already captured, but generalising poorly to those unseen. This is largely owning to the *uniquely* different styles amongst writers – *there is always a new style that is unobserved, and is drastically different to the already captured* (see Figure 1). The practical implication of this is, e.g., my iPad does

*Interned with SketchX

not recognise my handwriting as well as it does for my 4-year-old. Our ultimate vision is therefore to offer an "adapt to my writing" button, where one is asked to write a specific sentence, so to make recognition performance of my own writing on par with that of my child.

Prior work on resolving the style gap remains very limited. A very recent attempt turns to training using synthetic data, so to help the model to become more accommodating towards new styles [24]. However, synthetic data can hardly mimic all writer-specific styles found in the real-world, especially when the style is very unique. Although domain adaptation and generalisation approaches might sound viable, they generally do not offer satisfactory performance (as shown later in experiments), and require additional training via multiple gradient update steps. The sub-optimal performance can be mostly attributed to the large and often very unique domain gaps new writing styles bring, as opposed to the common dataset biases studied by domain adaption/generalisation.

In this paper, we turn to a meta-learning formulation, which not only yields performances that are of potential commercial value (from 81.3% to 89.2% Word Recognition Accuracy), but also offers quick adaption (with just a single gradient update) using very few samples ($\leq$16). The general motivation behind meta-learning [14, 31, 43] matches ours very well – absorbing information from related tasks and generalise onto unseen ones, by performing quick adaptation using a small set of examples *during testing*. However, getting it to work with HTR has its own challenges, and to our knowledge has not been tackled before in the literature. The main challenges come from the inherent character sequence recognition nature of HTR, which is different to conventional meta-learning whose objective is mostly few-shot classification [14, 42]. Furthermore, we importantly discovers that there also exists character-level style discrepancies, which when unaccounted for would trigger significant performance drop (see Section 3.4).

To address these specific challenges, we first introduce a character-wise cross-entropy loss to our meta-learning framework. This albeit being a simple change, is crucial in light of the sequence recognition nature of our problem. We further guide the adaptation by introducing *instance-specific weights* on top of the character-wise loss, instead of treating all characters equally by simply averaging [40, 28]. Modelling such character-specific weight is however non-trivial, as no fixed weight labels exist to supervise the learning process. Consequently, we let the model *learning-to-learn* instance-specific weights for character-wise cross-entropy loss during the adaptation step. Our final model, MetaHTR, is therefore a meta-learning pipeline for writer-adaptive HTR where the model itself adaptively weights different characters to prioritise learning from more discrepant characters. That is, during inference, our MetaHTR framework exploits few additional handwritten images of a specific writer through a support set, and gives rise to a writer-adapted model via a single gradient update step (see Figure 1). Our meta-learning design can be coupled with any state-of-the-art HTR model, and empirical investigation shows that model agnostic meta-learning (MAML) pipeline [14] provides a legitimate choice to design our MetaHTR framework upon.

Contributions of this paper can be summarised as follows: (1) We introduce for the first time, the problem of writer-adaptive HTR, where the model adapts to new writing styles with only very few samples during inference, (2) We introduce a meta-learning framework to tackle this new problem, by introducing learnable instance-wise weights for a character-specific loss specifically designed for HTR. (3) We confirm that our framework consistently improves upon even the most recent state-of-the-art methods.

2. Related Works

Text Recognition: Connectionist Temporal Classification (CTC) layer [17] made end-to-end sequence discriminative learning possible. Subsequently, CTC module was replaced by attention-based decoding mechanism [25, 40] that encapsulates language modeling, weakly supervised character detection and character recognition under a single model. This involves a rectification network to handle irregular text, followed by the final text recognition network. Needless to say attentional decoder became the state-of-the-art paradigm for text recognition for both scene text [28, 50, 48, 51] and handwriting [6, 29, 47, 52]. Different incremental propositions have been made in this context, such as designing multi-directional convolutional feature extractor [10], improving attention [9, 26] and stacking multiple BLSTM layers for better context modelling [28].

Besides word recognition accuracy, some works have focused on improving performance in low data regime, by designing adversarial feature deformation module [6], learning optimal augmentation strategy [29], and learning from synthetic training data via domain adaptation [52]. In this work, we introduce a new dimension of handwritten text recognition where model could be adapted during inference based on few handwritten word samples of the new writer in order to cope up with writer specific handwriting style.

Dealing with Writer Specificity: Writer identification [21, 20] has been a long standing problem in the handwriting analysis community. Furthermore, the phenomenon of writer specific nature of handwriting is accepted in forensic science [8, 21], and handwritten signature [19] is used as an authentication medium in various official and banking sectors. Few shot writer-specific handwriting generation started in the online handwritten data [1] with coordinates, and has further been realised for offline handwritten images [23]. Although the idea of style specific adap-

tation [34, 15, 12] has been introduced two decades ago, it has been limited to online handwritten characters [34], handcrafted feature normalisation [34], few pre-defined handwritten styles (not user specific) and fixed lexicon-vocabulary [12]. Nevertheless, there has been no work encasing the full potential of end-to-end trainable deep models for writer adaptive lexicon free offline handwritten word recognition. Adaptation could be done without any increase of model parameters –leading to cost-effective deployment.

Meta-learning: Meta-Learning aims to train a model on a series of related tasks, such that it learns the unseen task with only a few training samples [14]. One way is to learn optimal initialization, such that it quickly adapts to new tasks with a few data [42, 45]. Various meta-learning algorithms can be broadly categorised in three groups. While memory network based methods [32] learn across the task knowledge and aim to generalise to the unseen task, metric-based methods [42] aim to model a metric space in which learning is efficient with only a few samples. The earlier two approaches are mostly architecture specific, and have been employed for few-shot classification problem. In contrast, there has been a significant attention towards using optimization based meta-learning algorithms [14, 31, 43, 2] due to its model-agnostic nature. Specifically, we choose the recently introduced algorithm, model-agnostic meta-learning (MAML) [14] as it is compatible with any model which is trained with gradient descent and applicable to a variety of different learning problems. MAML aims to encode the prior knowledge into optimization process for fast adaption, and several variants have been proposed [31, 2, 38, 39]. Later MAML++ [2] introduced the sets of tricks to stabilize the training of the MAML. MetaSGD [27] proposes to train learnable learning rates for every parameter. Inspired by the success of domain adaptive dialog generation [36], we introduce MAML for writer adaptive handwritten text recognition. Nevertheless, we extend MAML for instant-adaptive sequence-recognition task over its off-the-shelf version [14] that was initially proposed for non-sequential few-shot classification problem.

3. Methodology

Overview: Traditionally, HTR model inputs a handwritten text image X and generates output character sequence $Y = (y_1, y_2, \ldots, y_L)$, where L is the variable length of text. Conventional HTR models [4] learn from multiple data instances often denoted as training dataset $\mathcal{D} = \{(X_1, Y_1), (X_2, Y_2), \cdots, (X_N, Y_N)\}$. Due to data instance specific training, it ignores the writer specific data distribution, without modeling the shared common knowledge [22] across different writers. Henceforth, the performance deteriorates on handwritten text images of unseen writers because of poor generalisation on diverse handwriting styles.

In contrast, we take a meta-learning approach which seeks to learn the general rules of handwritten recognition from distribution of multiple writer specific handwritten text recognition tasks. Let W_S and W_T denote the disjoint training and testing writer set respectively, i.e., $W_S \cap W_T = \emptyset$. The training and testing sets are denoted as $\mathcal{D}^S = \{\mathcal{D}^S_1, \cdots, \mathcal{D}^S_{|W_S|}\}$ such that $|W_S| > 1$ and $\mathcal{D}^T = \{\mathcal{D}^T_1, \cdots, \mathcal{D}^T_{|W_T|}\}$ such that $|W_T| \geq 1$. Every ith writer in both training and testing set, has its own set of N_i labelled images as $\mathcal{D}_i = \{(X_1, Y_1), (X_2, Y_2), \cdots, (X_{N_i}, Y_{N_i})\}$. During training, data is sampled across writer specific tasks from training set $\mathcal{D}^S$ to learn a good initialization point θ, by modeling the shared knowledge across different writers – such that it can quickly adapt to any new writer using few examples. During inference, with respect to jth writer from testing set as $\mathcal{D}^T_j$, we consider to have access to k (very few) labelled samples, based on which we update $\theta \mapsto \theta_j$ using just one gradient step to obtain a writer-specialised HTR model – this is called k-shot adaptation.

3.1. Baseline HTR Models

Nowadays, state-of-the-art text recognition networks, many of which were originally proposed for scene text [40], are now simultaneously validated [6, 29, 52, 47] over HTR datasets, as both follows a unified framework and objective. Therefore, attentional decoder based pipeline being the current state-of-the-art for text recognition, we select three seminal works, namely ASTER [40], SAR [26] and SCATTER [28], to use as our baseline HTR models. Moreover, ours is a meta-framework and could be adopted with most deep-text recognition pipelines.

For completeness, we briefly summarise the outline of text recognition model. In general, they consist of four components: (a) a convolutional feature extractor, (b) BLSTM layers for context modeling (c) a RNN decoder predicting the characters autoregressively one at a time step, and (d) an attentional block. Let the extracted convolutional feature map be $\mathcal{F} \in \mathbb{R}^{h' \times w' \times d}$ for a rectified image input, where h', w' and d signify height, width and number of channels. Every d dimensional feature at $\mathcal{F}_{i,j}$ encodes a particular local image region based on the receptive fields, which can be reshaped into list of vectors $\mathrm{F} = [f_1, f_2, \cdots, f_Q]$, where $Q = h' \times w'$. Thereafter BLSTM is employed to capture the long range dependencies on every position, thus alleviating the constraints of limited receptive field giving list of context rich vectors as: $\mathrm{H} = [h_1, h_2, ..., h_Q] = \texttt{BLSTM}([f_1, f_2, \cdots, f_Q])$. At every time step t, the decoder RNN predicts an output character or end-of-sequence (EOS) y_t based on three factors: a) previous internal state s_{t-1} of decoder RNN, (b) the character y_{t-1} predicted in the last step, and (c) a glimpse vector g_t representing the most relevant part of $\mathcal{F}$ for predicting y_t. In order to get g_t, the previous hidden state s_{t-1} acts as a query to discover the attentive regions as $g_t = \sum_{i=1}^{Q} a_t(i) \cdot h_i$. Attention score

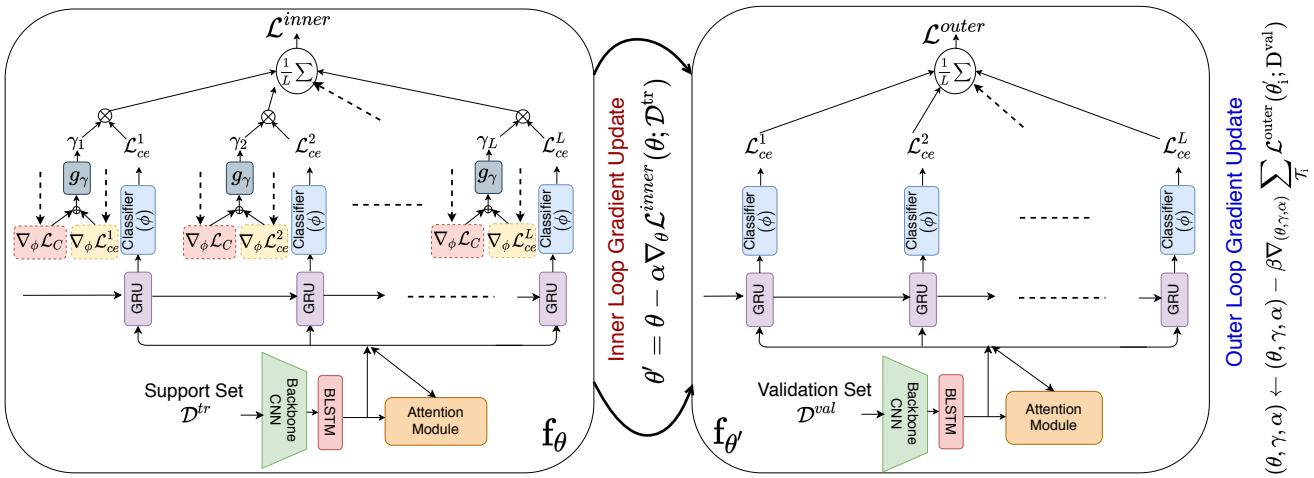


Figure 2. Our MetaHTR framework involves a bi-level optimisation process. The *inner loop optimisation* computes learnable character instance-weighted loss $\mathcal{L}^{inner}$ upon the support set, followed by obtaining a pseudo-updated model ($\theta^{'}$). This includes a learnable character instance specific weight prediction module (g_γ) and learnable layer-wise learning rate parameters (α). We expect $\theta^{'}$ to generalise well on remaining validation set, thus finally updating the meta-parameters (θ, γ, α) by *outer-loop* loss $\mathcal{L}^{outer}$ over the validation set.

at t^{th} time step $a_t(i)$ is computed by a function $\mathcal{A}(\cdot)$ as $\mathrm{a_t(i)} = \sigma(\mathrm{v}^\top \tanh(\mathrm{W_s s_{t-1}} + \mathrm{W_h h_i} + \mathrm{b_a}))$, σ being softmax across spatial positions $i \in [1, Q]$, and $\{W_s, W_h, b_a\}$ are learnable weights. The current hidden state s_t is updated by: $(\mathrm{o_t}, \mathrm{s_t}) = \mathrm{RNN}(\mathrm{s_{t-1}};\ [\mathrm{E(y_{t-1})},\ g_t]))$, where $E(.)$ is a character embedding layer with embedding dimension $\mathbb{R}^{128}$, and [.] signifies a concatenation operation. Finally, $\tilde{y}_t$ is predicted as:

$$p(\tilde{y}_t) = \mathrm{softmax}(\mathrm{W_o o_t} + \mathrm{b_o}) \tag{1}$$

We denote the complete set of parameters for every baseline as θ, and particularly that for the final classification layer as $\phi = \{W_o, b_o\}$. SAR [26] addresses 2D attention to eliminate the need of image rectification network [40] and SCATTER [28] couples multiple BLSTM layers for richer context modelling on the top of [40]. We refer the reader to [40, 26, 28] for further architectural details.

3.2. Basics: Gradient Based Meta-Learning

A popular optimization-based meta-learning algorithm is model-agnostic meta-learning (MAML) [14]. Here, the goal is to learn good initialization parameters θ that represent an across-task shared knowledge among related tasks, so that it can quickly adapt to any novel task of same distribution with only a few gradient update iterations.

Let $\mathcal{T}$ represent multiple tasks where $\mathcal{T}_i$ denotes the i^{th} task sampled from some task distribution $p(\mathcal{T})$ i.e. $\mathcal{T}_i \sim p(\mathcal{T})$. In our case, $\mathcal{T}_i$ is sampled across a task containing labelled training data from a specific writer $\mathcal{D}_i^S$. Each task $\mathcal{T}_i$ consist of a support set D^{tr} and a validation set D^{val}. Additionally, let a neural network be represented by f_θ, where θ is the initial parameter of the network. Intuitively, MAML tries to find a good initialization of parameters θ, representing the prior or meta-knowledge, so that a few updates of θ using D^{tr} can make large improvements by reducing the error measures and boosting the performance in D^{val}. To learn this optimal initialisation parameter θ, we first adapt (task-specific) f_θ to $\mathcal{T}_i$ using D^{tr} by fine-tuning:

$$\theta^{'} = \theta - \alpha \nabla_\theta \mathcal{L}^{inner}(\theta; \mathrm{D}^{tr}) \tag{2}$$

Evaluation of the adapted model is performed on unseen examples sampled from the same task $\mathrm{D}^{val} \in \mathcal{T}_i$, to measure the generalisation of $f_{\theta'}$. This acts as a feedback for MAML to adjust its initialization parameters θ to achieve better generalisation on any $\mathcal{T}_i$ (across-task):

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i} \mathcal{L}^{outer}(\theta^{'}; \mathrm{D}^{val}) \tag{3}$$

3.3. Meta-learning for Writer Adaptive HTR

Let the baseline text recognition model in our case (parameterized by θ) be represented as f_θ. Instead of naive writer-specific fine-tuning that usually requires hundreds of gradient updates, we seek to learn the general rules [14] of handwriting recognition using multiple writer specific handwritten text recognition tasks. Meta-training involves sampling tasks which here is defined with respect to each specific writer. In particular, $\mathcal{T}_i \sim p(\mathcal{T})$ indicates selecting 2B labelled samples from i-th writer training set $\mathcal{D}_i^S$, out of which we make D^{tr} for inner loop update and D^{val} for outer loop update each containing B samples. It should be noted that model parameters are updated by averaging gradients [11] of outer loop loss over a meta-batch $\{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_M\}$ having size of M.

Character-Wise (CW) Loss for Meta-learning: Given the backdrop of MAML (section 3.2) and any baseline text recognition model (section 3.1), one can train a meta-learning model [11], given an access to the loss function. A naive approach would be using traditional cross-entropy loss [40], which usually trains any attentional-decoder based text recognition system, for both inner (Eqn.

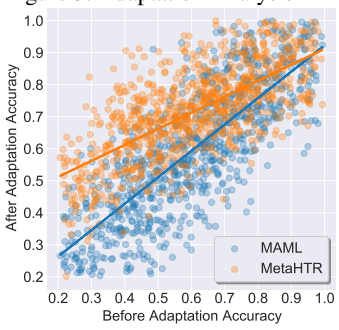
2) and outer loop (Eqn. 3). If output from text-recognition model is $\bar{Y} = \{\bar{y_1}, \bar{y_2}, \cdots, \bar{y_L}\}$, character-wise (CW) cross-entropy (ce) loss summed over the ground-truth output sequence $Y = \{y_1, y_2, \cdots, y_L\}$ can be defined as:

$$\mathcal{L}_{ce} = -\frac{1}{L}\sum_{t=1}^{L} L_{ce}(y_t, \bar{y_t}) = -\frac{1}{L}\sum_{t=1}^{L} y_t \log p(\bar{y_t}) \quad (4)$$

3.4. Learning-to-Learn weights for CW Loss

Motivation: Being a sequence recognition problem, $\mathcal{L}_C$ involves a summation operation [40] over the character sequence, thus treating every character specific cross-entropy loss equally. We conjecture this task specific adaptation for sequence recognition could be boosted if weight values for each character instance-specific loss are learned, such that the model adapts better with respect to those characters having a high discrepancy. Intuitively speaking, our model learns knowledge across tasks, where given a word 'covid' from a new specific writer, properties of certain handwritten characters (e.g. 'c', 'v', 'i') could be close to the encoded knowledge of MAML's initialisation parameter [5], to enhance easier recognition. On the contrary, significant discrepancy could exist among certain characters (e.g. 'o', 'd') that are difficult to recognise using average knowledge encapsulated inside MAML's initialisation parameter. Thus, during fast adaptation, the model needs to update itself by prioritising the optimisation with respect to those particular characters (e.g. 'o', 'd') whose style variation is more towards unknown to the model's initialization. In other words, for faster adaptation via inner loop loss, we intend to learn the instance specific weight of character-wise cross-entropy loss instead of simply averaging over all characters. Recent literature shows that meta-learning provides the flexibility to learn any hyper-parameters [14], parameterized loss functions [7], learning rates [27], or weight attenuation [5] in the meta-learning process itself. Character wise recognition accuracy from different writers before and after adaptation are plotted in Figure 3. It can be seen that the characters getting low accuracy using MAML's initialisation parameter before adaptation (X axis) also get low accuracy even after the adaptation (Y axis). However, our proposed learnable character-instance specific weight of MetaHTR helps to enhance the performance of those discrepant characters after adaptation. More insightful analysis is in Section 4.3.

Figure 3. Adaptation Analysis



Meta-Optimisation: Naturally, the question arises what information could be used to determine these weights. Recent studies show that the gradients used for fast adaptation (inner loop) contains the information [5] related to disagreement (e.g. this knowledge further needs to be learned or accumulated in the adaptation process) with respect to model's initialization parameters. As calculating gradients with respect to all the model's parameters is quite cumbersome, we calculate gradient of t-th character specific cross-entropy loss with respect to final classification layer (parameter ϕ) as $\nabla_\phi \mathcal{L}^t_{ce}(\theta)$. It is then concatenated with gradients of mean loss (Eqn. 4) which sums over character sequence with respect to ϕ (both gradient matrix being flattened) as $\mathcal{G}_t = \texttt{concat}\big(\nabla_\phi \mathcal{L}^t_{ce}(\theta), \nabla_\phi \mathcal{L}_c(\theta)\big)$. We postulate that gradient of the mean and character-instance specific losses provide knowledge towards determining how to weigh different character specific losses. Thus, we pass this $\mathcal{G}_t$ through a network g_γ predicting a scalar weight value for t-th character specific loss as:

$$\gamma_t = g_\gamma\big(\texttt{concat}\big(\nabla_\phi \mathcal{L}^t_{ce}(\theta), \nabla_\phi \mathcal{L}_c(\theta)\big)\big) \quad (5)$$

Here, g_γ is a 3-layer MLP network of parameters γ followed by a sigmoid to generate weights. Therefore, the instance weighted inner loop loss becomes:

$$\mathcal{L}^{inner} = -\sum_{t=1}^{L} \gamma_t \cdot L_{ce}(y_t, \bar{y_t}) \quad (6)$$

Traditional MAML uses a predefined constant learning rate α in the inner-level optimization. Inspired from [27, 46], we specify a learnable rate for each layer as follows:

$$\theta^{'} = \theta - \alpha \cdot \nabla_\theta \mathcal{L}^{inner}(\theta, D^{tr}) \quad (7)$$

where α is a vector of size equal to number of layers in the baseline HTR model. The outer-loop loss is kept as traditional $\mathcal{L}_C$ (see Figure 2). Please note that $\theta^{'}$ is dependant on $\{\theta, \gamma, \alpha\}$ through the inner loop update (Eqn. 7), and all three meta-parameters (θ, γ, α) are meta-learned via the outer-loop update as $(\theta, \gamma, \alpha) \leftarrow (\theta, \gamma, \alpha) - \beta \nabla_{(\theta,\gamma,\alpha)} \sum_{\mathcal{T}_i} \mathcal{L}^{outer}(\theta_i^{'}; \mathrm{D}^{val})$. Training and inference process is summarised in Algorithm 1 and 2, respectively.

4. Experiments

Datasets: We evaluate the performance of our writer adaptive MetaHTR on two popular datasets of Latin scripts, IAM [30] and RIMES [18]. While IAM contains a total number of 1,15,320 English handwritten word images written by 657 different writers, RIMES consists of 66,982 French word images of 1300 different writers. Both datasets contain word samples with annotated writer information, thus enabling sampling of writer specific meta-batches, to perform episodic training [14]. For RIMES, we use samples from a subset of 375 writers which is usually used in writer identification task [41] as well. Following [6], we use the same partition for training, validation and testing as provided for IAM, while the partition released by ICDAR 2011 competition is used for RIMES.

Algorithm 1 Training for Writer Adaptive MetaHTR

Input: Training dataset $\mathcal{D}^S = \{\mathcal{D}_1^S, \mathcal{D}_2^S \ldots, \mathcal{D}_{|W_S|}^S\}$; β as learning rate.
Initialise: Initialise θ, γ, α
Output: Optimised meta-parameters $\{\theta, \gamma, \alpha\}$
while not done **do**
  Sample writer specific $\mathcal{T}_i = \{\mathrm{D}_i^{tr}, \mathrm{D}_i^{val}\} \sim p(\mathcal{T})$
  for each task $\mathcal{T}_i$ **do**
    Evaluate inner objective: $\mathcal{L}^{inner}(\theta; \mathrm{D}_i^{tr})$
    Adapt: $\theta_i^{'} = \theta - \alpha \nabla_\theta \mathcal{L}^{inner}(\theta; \mathrm{D}_i^{tr})$
    Compute outer objective: $\mathcal{L}^{outer}(\theta_i^{'}; \mathrm{D}_i^{val})$
  end for
  Update meta-parameters: $(\theta, \gamma, \alpha) \leftarrow (\theta, \gamma, \alpha) - \beta \nabla_{(\theta,\gamma,\alpha)} \sum_{\mathcal{T}_i} \mathcal{L}^{outer}(\theta_i^{'}; \mathrm{D}^{val})$
end while

Algorithm 2 Inference for Writer Adaptive MetaHTR

Input: Testing dataset $\mathcal{D}^T = \{\mathcal{D}_1^T, \mathcal{D}_2^T, \ldots, \mathcal{D}_{|W_T|}^T\}$; meta-learned model parameters $\{\theta, \gamma, \alpha\}$, number of gradient updates n, a given writer j.
Get the available support set $\mathrm{D}_j^{tr} \in \mathcal{T}_j$
for n steps **do**
  Evaluate inner objective: $\mathcal{L}^{inner}(\theta; \mathrm{D}_j^{tr})$
  Adapt: $\theta_j^{'} = \theta - \alpha \nabla_\theta \mathcal{L}^{inner}(\theta; \mathrm{D}_j^{tr})$
end for
Return *Writer specialised* HTR model params. $\theta_{\mathbf{j}}^{'}$.

Implementation Details: Following traditional supervised learning protocols [4], we first pre-train every considered baseline HTR models using ADADELTA optimiser with learning rate 1, and a batch size 64. Thereafter, we perform the meta-training process on pre-trained baseline model's parameters for 20 epochs according to Algorithm 1. Only one inner loop update is used during inference (Algorithm 2) unless otherwise mentioned. Additionally, the effect of increasing inner loop updates is shown in our ablative study (section 4.3). During meta-training, we consider meta-batch size of $M = 8$ – our meta-batch comprises 8 different writer specific tasks $\mathcal{T}_i$, that are used for updating the meta-parameters by taking average gradient. Within each task, the batch-size of support and validation set is $B = 16$. We use ADAM as meta-optimiser with outer-loop learning rate β as 0.0001, while the inner-loop learning α is meta-learned along with instance-specific weight γ_t of character-wise cross-entropy loss. We implemented our framework in PyTorch [33] and conducted experiments on a 11 GB Nvidia RTX 2080-Ti GPU.

Evaluation Metric: We use Word Recognition Accuracy (WRA) [6] for both *with*-Lexicon (**L**) and *with* No-Lexicon (**NL**) (unconstrained) HTR. As there is no separate adaptation set (support set) explicitly defined for testing set writers W_T in either of these datasets, we do the following : let N_j^T be the total number of images under test writer j, we take random k images (for k-shot adaptation) as the support set for adaptation and the adapted model is evaluated on remaining $(N_j^T - k)$ images. We do this for ten times, and cite average result to reduce the randomness. We use $k = 16$ for our cited results unless mentioned otherwise. Due to this adaptation set constraint, only those writers having more than 32 word images, contribute towards accuracy calculation. For fairness, we ensured uniform adaptation and testing set for all the competitive baselines.

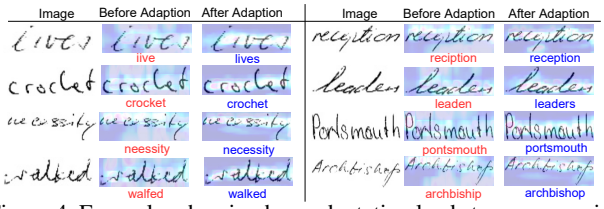


Figure 4. Examples showing how adaptation leads to more consistent character-aligned attention map, followed by better recognition performance. (Red: Incorrect, Blue: Correct)

4.1. Competitors

To the best of our knowledge, there exists no prior work particularly dealing writer specific adaptation for offline handwritten word images. However, we design several strong baselines from *five* different perspectives to justify our MetaHTR framework. ***(i)* Learning Augmentation Approach:** Recently, there have been attempts to learn efficient data-augmentation strategy to learn the style-variation present in the handwritten data using a learnable agent [29] or an adversarial feature deformation module [6]. ***(ii)* Generative Approach:** One can synthetically generate [23] multiple handwritten images with different words mimicking someone's handwriting style from few given handwritten examples (adaptation set). Thereafter, naive fine-tuning [44] could be done over large (5K in our experiment) synthetically generated data considering them as writer's style specific training-set. ***(iii)* Meta-Learning based Adaptation:** We follow this paradigm in our MetaHTR framework, however, there could be some off-the-shelf alternatives. An obvious choice could be naive-fine-tuning [44] over the same labelled images of the adaptation set. We compare our method with typical MAML [14] formulation, along with its first-order (MAML-FO) approximated version [14] to judge how far the performance drops, while improving the computational speed. We compare with MetaSGD [27] which uses learnable learning rate for each parameter and ANIL [37], where only final classification layer (ϕ) is pseudo-updated in the inner loop for computational ease. ***(iv)* Domain Adaptation (DA) Approach:** All the training writers are considered as a single source domain, and the trained model is adapted using samples from a specific writer using adversarial learning as used in [24]. ***(v)* Domain Generalisation (DG) Approach:** Domain generalisation aims to learn a generalised

Table 1. Comparison among Baselines, naive Fine-tuning, and MetaHTR for using Lexicon (L), No Lexicon (NL). GAP: difference between MetaHTR (NL) vs Baseline (NL). We almost get around 5-7% WRA improvement over respective baselines under NL setting.

Methods	IAM [30] (WRA)							RIMES [3] (WRA)						
	Baseline		Fine-tuning		MetaHTR			Baseline		Fine-tuning		MetaHTR		
	L	NL	L	NL	L	NL	GAP	L	NL	L	NL	L	NL	GAP
ASTER [40]	90.3	81.3	90.5	81.7	94.1	89.2	7.9 ↑	93.6	87.4	93.7	87.7	96.5	93.4	6.0 ↑
SAR [26]	91.6	84.4	91.7	84.7	94.8	91.5	7.1 ↑	93.8	88.7	93.8	88.8	96.5	93.7	5.0 ↑
SCATTER [28]	91.7	84.6	92.0	85.1	94.8	91.6	7.0 ↑	93.8	88.8	93.9	89.0	96.6	93.9	5.1 ↑

Table 2. Performance analysis with different approaches.

		IAM		RIMES	
		L	NL	L	NL
DG	ASTER [40] + DG	91.7	84.6	93.9	89.1
	SAR [26] + DG	92.4	85.7	94.3	89.5
	SCATTER [28] + DG	92.5	85.9	94.6	89.7
DA	ASTER [40] + DA	90.3	81.1	93.8	87.4
	SAR [26] + DA	91.9	84.8	93.7	88.9
	SCATTER [28] + DA	92.0	84.6	93.6	89.2
GA	ASTER [40] + GA	91.2	83.6	93.7	88.0
	SAR [26] + GA	91.8	84.8	93.8	88.4
	SCATTER [28] + GA	92.0	85.2	93.8	88.7
Augmnt.	Luo *et al.* [29]	92.5	86.0	95.6	90.8
	AFDM [6]	91.2	83.6	95.2	88.2
	Luo *et al.* + AFDM [6]	92.7	86.7	96.1	91.3
Meta Learning based Adaptation	ASTER [40] Baseline	90.3	81.3	93.6	87.4
	ASTER [40] + MAML	93.0	87.1	96.3	91.9
	ASTER [40] + MAML-FO	92.9	86.9	96.2	91.6
	ASTER [40] + MetaSGD	91.1	83.4	93.7	88.0
	ASTER [40] + ANIL	93.0	87.0	96.2	91.7
	ASTER [40] + **Ours (MetaHTR)**	94.1	89.2	96.5	93.4
	SAR [26] Baseline	91.6	84.4	93.8	88.7
	SAR [26] + MAML	94.1	89.1	96.4	92.4
	SAR [26] + MAML-FO	94.0	88.8	96.3	92.2
	SAR [26] + MetaSGD	91.8	84.9	93.9	88.9
	SAR [26] + ANIL	94.0	88.9	96.3	92.3
	SAR [26] + **Ours (MetaHTR)**	94.8	91.5	96.5	93.7
	SCATTER [28] Baseline	91.7	84.6	93.6	88.8
	SCATTER [28] + MAML	94.1	89.3	96.4	92.5
	SCATTER [28] + MAML-FO	94.0	88.9	96.3	92.3
	SCATTER [28] + MetaSGD	92.0	85.2	93.9	89.1
	SCATTER [28] + ANIL	94.1	89.1	96.4	92.4
	SCATTER [28] + **Ours (MetaHTR)**	94.8	91.6	96.6	93.9

model via episodic training from writer-specific task distribution, which can directly perform well across unseen writers without any further gradient update. Following [16], we can twist our meta-learning pipeline to fit the objective of DG. Thus, we optimise the baseline HTR model using weighted ($\lambda = 0.5$) summation for gradient (over meta-train set) and meta-gradient (over meta-test split through inner loop update). Mathematically, using our notation: $\text{argmin}_\theta \, \lambda \cdot \mathcal{L}(\theta; D^{tr}) + (1-\lambda) \cdot \mathcal{L}(\theta'; D^{val})$, where $\mathcal{L}$ is the loss function (see Eqn. 4) and θ' is pseudo-updated parameter by inner loop with learning rate 0.0005. It is worth noting that although DG [16] and augmentation based approaches [29] cannot be compared directly to ours, as they do not involve any model-updation step at test time.

4.2. Result Analysis and Discussion

The unconstrained WRA on IAM is used to cite any performance gap for describing rest of the paper, unless mentioned otherwise. In Table 1, we compare our MetaHTR framework with corresponding state-of-the-art (SOTA) baselines [40, 26, 28] and naive fine-tuning [44] method. MetaHTR outperforms (Figure 4) every SOTA baseline by a significant margin of around 5-7%.

Furthermore, we compare with *five* classes of alternative approaches in Table 2 to tackle the style variation from different writers. We observe the following: **(i) GA:** While naive-fine-tuning hardly gives any improvement, the generative approach opens the room for generating multiple images with different words by mimicking some particular writer's handwriting style from the same adaptation set as used in MetaHTR. This being followed by naive fine-tuning does improve over the baseline HTR models by 2.3%, but lags behind our MetaHTR framework by 5.6% for ASTER baseline. We attribute this to the inconsistency of style in the generated image [23] with respect to any given writer. Fine-tuning via synthetically generated images hurts the HTR model performance on real handwritten samples due to the inherent domain gap [16]. Furthermore, style conditioned handwritten image generation involving a separate cumbersome network make it computationally more expensive. **(ii) DG:** Although DG approach [16] improves the performance for unseen writers compared to the baseline models, it still lags behind our MetaHTR framework by $4.6\%, 5.8\%, 5.7\%$ with respect to our three baselines due to obvious reasons of not exploiting writer specific few-shot labelled data during inference. Moreover, this could be a very straight forward alternative for cases where we do not have any access to specific writer's samples, but can enrich the model with writer-specific data distribution via episodic training [16], to learn a common knowledge across writers for better performance than baseline models. **(iii) Augmnt:** Augmentation based approach improves the performance on top of baseline models by incorporating synthetic learnable deformations, both in image space [29] and feature space [6]. Having no option of using specific writer's handwritten examples, this falls inferior to our proposed method as well. **(iv) DA:** The performance of DA is found to be limited due to scarce adaptation data scenario and alleged instability of adversarial training. **(v) Meta Learning based Adaptation:** Our close competitors are gradient-based meta-learning alternatives [22]. Out all of these, MAML [14] scores quite close to ours, yet lags by 2.1%, 2.4% 2.3% for ASTER, SAR and SCATTER respectively. Although first-order approximation of MAML (MAML-FO) is computationally simpler, unconstrained WRA drops by 0.2% compared to MAML on ASTER baseline. We want to emphasise that our model needs second-order gradient computation [14] in the outer loop process as g_γ is related to $\mathcal{L}^{outer}$ through inner loop update. Meta-SGD [27] needs double the number of parameters than MAML as it meta-learns learning rate value for every parameter. To our surprise however, it performs lower than MAML fitted on top of our baseline text recog-

nition models. Probably, the need of extensive parameter updates leads towards over-fitting by the meta-learning process, thus failing to generalise. In contrast, we use layer-wise learnable learning-rate in MetaHTR which is computationally way less expensive and provides better generalisation. Although computationally cheaper, ANIL [37] is still inferior to MAML baseline. We attribute the superiority of our method over other gradient based meta-learning algorithms for sequence recognition task (e.g. HTR) to two main factors: learnable character-instance specific weighting mechanism for inner-loop loss and re-designing layer-wise learnable learning rate.

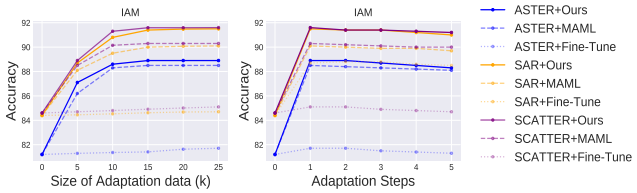


Figure 5. Unconstrained WRA of MetaHTR with varying adaptation set-size (k), and adaptation steps (best viewed when zoomed).

4.3. Ablation Study

[i] Significance of learnable γ_t***:*** (a) To show the efficacy of the learnable instance specific weight for character specific loss, we remove g_γ and use simple mean cross-entropy loss (Eqn. 4) for inner-loop update. By doing this, the performance drops by 1.9%, 2.2% and 2.1% for ASTER, SAR and SCATTER, respectively on IAM dataset. (b) Next, we get deeper to verify whether different characters of a same writer really shows discrepancy [5] or not. For that, we evaluate character specific accuracy using our Meta-HTR model with learned initialization parameter [14]. HMM-based Viterbi Forced alignment is used to locate and crop out every character from word images in a cost-effective way. From the Figure 6, it is qualitatively evident that there exist significant variation in terms of recognition accuracy across different characters – signifying that a few characters are harder to adapt or recognise, than others due to wide style discrepancy. For further analysis, we plot the average (for support set) character instance specific weight predicted by our meta-learned model with respect to a particular writer, for which the result is fairly consistent compared to character recognition result. This indicates that those characters which obtain low recognition accuracy, mostly receive higher weights in the inner loop loss calculation, and vice-versa, which strongly supports our intuition. (c) Furthermore, we try to explore individual gradient $\nabla_\phi \mathcal{L}_{ce}^t$ coming from every t-th character prediction of attentional decoder without concatenating it with the mean gradient $\nabla_\phi \mathcal{L}_c$ for γ_t calculation (Eqn. 5). However, the performance drop (by 1.9%) using ASTER baseline implies that character-instance specific gradient along with mean gradient, provides more context to judge the character wise style discrepancy with respect to the initialisation parameters.

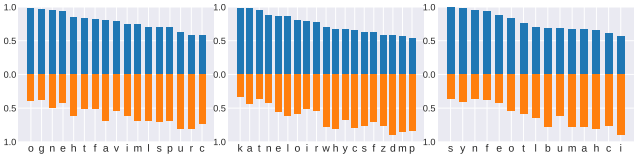


Figure 6. Different sets of characters (support set) of any writer (three writers shown here) show varying recognition accuracy (upper blue), thus signifying different levels of discrepancy against MetaHTR's initialization parameter. Characters having lower accuracy *mostly* get higher weights (lower orange) in the inner-loop adaptation process, and vice versa. Note: X-axis for different writers is sorted in terms of recognition accuracy (Best if zoomed).

[ii] Layer-wise learnable learning rate: To analyse the contribution of learnable layer-wise learning rate mechanism, we replace it with a fixed inner-loop learning rate of 0.001 (optimised) keeping rest of the design same. This leads to a drop of 0.8%, 0.6% and 0.6% using respective three baselines, thus justifying its contribution. Furthermore, MetaSGD [27] is nearly 2.5x times computationally expensive than MAML on our HTR baselines during inference. Although computational overhead rises due to our module g_γ, the performance gain of nearly 2% overshadows additional 0.2x inference time compared to MAML under similar setup. ***[iii] Size of adaptation data:*** A few examples are enough to achieve instant adaptation as suggested by many existing few-shot works [36, 11]. Here, we vary the size of adaptation set (k) in Figure 5 to discover the effect on recognition accuracy. The performance nearly saturates between 10-20 samples, thus justifying the few-shot design. Moreover, our MetaHTR tends towards saturation with slightly less samples compared to MAML under same setup. ***[iv] Number of adaption steps:*** The number of adaptation steps during inference is varied in Figure 5. In summary, just a single gradient step update, used in most of our experiments, shows highest performance gain. On the contrary, more updates sometimes showed diminishing results that contradicts the tendency reported in [14]. The reasons might be that inner loop concentrates on unnecessary style details, thus forgetting the generic prior knowledge learned.

5. Conclusion

In this paper, we proposed a novel writer adaptive offline handwritten text recognition framework which aims to fully utilise the additional writer specific information available at test time. We employ an extension of Model Agnostic meta-learning (MAML) algorithm to train our writer adaptive HTR network that can quickly adapt its parameters according to the writer's handwritten text. The proposed framework is applied to three existing text recognition models without changing its architecture, and shows consistently improved performance on multiple handwritten benchmarks datasets.

References

[1] Emre Aksan, Fabrizio Pece, and Otmar Hilliges. Deepwriting: Making digital ink editable via deep generative modeling. In *CHI*, 2018. 2

[2] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your maml. *ICLR*, 2019. 3

[3] Emmanuel Augustin, Matthieu Carré, Emmanuèle Grosicki, J-M Brodin, Edouard Geoffrois, and Françoise Prêteux. Rimes evaluation campaign for handwritten mail processing. 2006. 7

[4] Jeonghun Baek, Geewook Kim, Junyeop Lee, Sungrae Park, Dongyoon Han, Sangdoo Yun, Seong Joon Oh, and Hwalsuk Lee. What is wrong with scene text recognition model comparisons? dataset and model analysis. In *ICCV*, 2019. 3, 6

[5] Sungyong Baik, Seokil Hong, and Kyoung Mu Lee. Learning to forget for meta-learning. In *CVPR*, 2020. 5, 8

[6] Ayan Kumar Bhunia, Abhirup Das, Ankan Kumar Bhunia, Perla Sai Raj Kishore, and Partha Pratim Roy. Handwriting recognition in low-resource scripts using adversarial learning. In *CVPR*, 2019. 1, 2, 3, 5, 6, 7

[7] Yevgen Chebotar, Artem Molchanov, Sarah Bechtle, Ludovic Righetti, Franziska Meier, and Gaurav Sukhatme. Meta-learning via learned loss. *arXiv preprint arXiv:1906.05374*, 2019. 5

[8] Shiming Chen, Yisong Wang, Chin-Teng Lin, Weiping Ding, and Zehong Cao. Semi-supervised feature learning for improving writer identification. *Information Sciences*, 2019. 2

[9] Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou. Focusing attention: Towards accurate text recognition in natural images. In *ICCV*, 2017. 2

[10] Zhanzhan Cheng, Yangliu Xu, Fan Bai, Yi Niu, Shiliang Pu, and Shuigeng Zhou. Aon: Towards arbitrarily-oriented text recognition. In *CVPR*, 2018. 2

[11] Myungsub Choi, Janghoon Choi, Sungyong Baik, Tae Hyun Kim, and Kyoung Mu Lee. Scene-adaptive video frame interpolation via meta-learning. In *CVPR*, 2020. 4, 8

[12] Scott D. Connell and Anil K. Jain. Writer adaptation for online handwriting recognition. *T-PAMI*, 2002. 3

[13] Brian Davis, Chris Tensmeyer, Brian Price, Curtis Wigington, Bryan Morse, and Rajiv Jain. Text and style conditioned gan for generation of offline handwriting lines. In *BMVC*, 2020. 1

[14] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *ICML*, 2017. 2, 3, 4, 5, 6, 7, 8

[15] Michel Gilloux. Writer adaptation for handwritten word recognition using hidden markov models. In *ICPR*, 1994. 3

[16] J. Gou, X. Zhu, C. Zhao, D. Cao, Z. Lei, and S. Z. Li. Learning meta face recognition in unseen domains. In *CVPR*, 2020. 7

[17] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *ICML*, 2006. 2

[18] Emmanuèle Grosicki and Haikal El Abed. Icdar 2009 handwriting recognition competition. In *ICDAR*, pages 1398–1402, 2009. 5

[19] Luiz G Hafemann, Robert Sabourin, and Luiz S Oliveira. Learning features for offline handwritten signature verification using deep convolutional neural networks. *Pattern Recognition*, 2017. 2

[20] Sheng He and Lambert Schomaker. Deep adaptive learning for writer identification based on single handwritten word images. *Pattern Recognition*, 2019. 2

[21] Sheng He and Lambert Schomaker. Fragnet: Writer identification using deep fragment networks. *T-IFS*, 2020. 2

[22] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020. 3, 7

[23] Lei Kang, Pau Riba, Yaxing Wang, Marçal Rusiñol, Alicia Fornés, and Mauricio Villegas. Ganwriting: Content-conditioned generation of styled handwritten word images. In *ECCV*, 2020. 1, 2, 6, 7

[24] Lei Kang, Marçal Rusiñol, Alicia Fornés, Pau Riba, and Mauricio Villegas. Unsupervised adaptation for synthetic-to-real handwritten word recognition. In *WACV*, 2020. 2, 6

[25] Chen-Yu Lee and Simon Osindero. Recursive recurrent nets with attention modeling for ocr in the wild. In *CVPR*, 2016. 2

[26] Hui Li, Peng Wang, Chunhua Shen, and Guyu Zhang. Show, attend and read: A simple and strong baseline for irregular text recognition. In *AAAI*, 2019. 1, 2, 3, 4, 7

[27] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017. 3, 5, 6, 7, 8

[28] Ron Litman, Oron Anschel, Shahar Tsiper, Roee Litman, Shai Mazor, and R Manmatha. Scatter: selective context attentional scene text recognizer. In *CVPR*, 2020. 1, 2, 3, 4, 7

[29] Canjie Luo, Yuanzhi Zhu, Lianwen Jin, and Yongpan Wang. Learn to augment: Joint data augmentation and network optimization for text recognition. In *CVPR*, 2020. 1, 2, 3, 6, 7

[30] U-V Marti and Horst Bunke. The iam-database: an english sentence database for offline handwriting recognition. *IJDAR*, 2002. 5, 7

[31] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018. 2, 3

[32] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *NeurIPS*, 2018. 3

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 6

[34] John C Platt and Nada Matic. A constructive rbf network for writer adaptation. In *NeurIPS*, 1997. 3

[35] Arik Poznanski and Lior Wolf. Cnn-n-gram for handwriting word recognition. In *CVPR*, 2016. 1

[36] Kun Qian and Zhou Yu. Domain adaptive dialog generation via meta learning. *ACL*, 2019. 3, 8

[37] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of maml. In *ICLR*, 2020. 6, 8

[38] Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*, 2018. 3

[39] Aneeshan Sain, Ayan Kumar Bhunia, Yongxin Yang, Tao Xiang, and Yi-Zhe Song. Stylemeup: Towards style-agnostic sketch-based image retrieval. In *CVPR*, 2021. 3

[40] Baoguang Shi, Mingkun Yang, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Aster: An attentional scene text recognizer with flexible rectification. *T-PAMI*, 2018. 1, 2, 3, 4, 5, 7

[41] Imran Siddiqi and Nicole Vincent. Text independent writer recognition using redundant writing patterns with contour-based orientation and curvature features. *Pattern Recognition*, 2010. 5

[42] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017. 2, 3

[43] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In *CVPR*, 2019. 2, 3

[44] Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, et al. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *ICLR*, 2020. 6, 7

[45] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NeurIPS*, 2016. 3

[46] Guangting Wang, Chong Luo, Xiaoyan Sun, Zhiwei Xiong, and Wenjun Zeng. Tracking by instance detection: A meta-learning approach. In *CVPR*, 2020. 5

[47] Tianwei Wang, Yuanzhi Zhu, Lianwen Jin, Canjie Luo, Xiaoxue Chen, Yaqiang Wu, Qianying Wang, and Mingxiang Cai. Decoupled attention network for text recognition. In *AAAI*, 2020. 1, 2, 3

[48] MingKun Yang, Yushuo Guan, Minghui Liao, Xin He, Kaigui Bian, Song Bai, Cong Yao, and Xiang Bai. Symmetry-constrained rectification network for scene text recognition. In *ICCV*, 2019. 2

[49] Mohamed Yousef and Tom E Bishop. Origaminet: Weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold. In *CVPR*, 2020. 1

[50] Deli Yu, Xuan Li, Chengquan Zhang, Tao Liu, Junyu Han, Jingtuo Liu, and Errui Ding. Towards accurate scene text recognition with semantic reasoning networks. In *CVPR*, 2020. 2

[51] Fangneng Zhan, Shijian Lu, and Chuhui Xue. Verisimilar image synthesis for accurate detection and recognition of texts in scenes. In *ECCV*, 2018. 2

[52] Yaping Zhang, Shuai Nie, Wenju Liu, Xing Xu, Dongxiang Zhang, and Heng Tao Shen. Sequence-to-sequence domain adaptation network for robust text image recognition. In *CVPR*, 2019. 2, 3