[Egyptian Informatics Journal 23 (2022) 373–382](https://doi.org/10.1016/j.eij.2022.02.007)

Contents lists available at [ScienceDirect](http://www.sciencedirect.com/science/journal/11108665)

Egyptian Informatics Journal

journal homepage: [www.sciencedirect.com](http://www.sciencedirect.com/)

[](http://crossmark.crossref.org/dialog/?doi=10.1016/j.eij.2022.02.007&domain=pdf)Writer adaptation for E2E Arabic online handwriting recognition via adversarial multi task learning

Fakhraddin Alwajih [a](#_bookmark0),[b](#_bookmark1),[⇑](#_bookmark3), Eman Badr [a](#_bookmark0),[c](#_bookmark2), Sherif Abdou [a](#_bookmark0)

a *Department of Information Technology, Cairo University, Giza, Egypt*

b *Department of Computer Science and Information Technology, Ibb University, Ibb, Yemen*

c *University of Science and Technology, Zewail City of Science, Technology and Innovation, Giza, Egypt*

a r t i c l e i n f o

*Article history:*

Received 20 September 2021

Accepted 12 February 2022

Available online 1 March 2022

*Keywords:*

Writer adaptation Adversarial learning Multi task learning

Arabic online handwriting recognition Connectionist temporal classification Convolutional neural networks Bidirectional long short-term memory

a b s t r a c t

The importance of online handwriting recognition has been rapidly increasing over recent years due to the rapid technological advances in handheld devices and communication software with handwriting interfaces. Deep learning end-to-end (E2E) models have provided high recognition rates as part of online handwriting recognition systems. However, attaining even higher performance levels requires supple- menting these models with adaptation techniques that cater to individual penmanship. This study pro- poses a writer adaptation technique for Arabic online handwriting recognition systems that employs adversarial Multi-Task Learning (MTL). Adversarial training and MTL modify the deep-features distribu- tion of the Writer Dependent (WD) model, leading its output to closely resemble that of the Writer Independent (WI) model. The design of the proposed method entails two tasks: label classification (pri- mary task) and model features discrimination (secondary task). Our method was designed to jointly opti- mize both sub-networks. The proposed technique was tested against the E2E Connectionist Temporal Classification (CTC) based model, a combination of both Convolutional Neural Networks (CNNs) and Bidirectional Long Short-term Memory (BiLSTM). The proposed models were trained and evaluated against two large datasets (the Online-KHATT and CHAW). In supervised adaptation, it achieved an abso- lute Character Error Rate (CER) of up to 1.83% and an absolute Word Error Rate (WER) reduction of 11.71% over the WI model. Additionally, supervised adaptation achieved an absolute CER of up to 0.84% and an absolute WER reduction of 6.77% over the fine-tuned model. In unsupervised adaptation, the proposed method achieved an absolute CER of up to 0.5% absolute and an absolute WER reduction of 1.74% absolute (WER) reduction over the WI. Our experimental results indicate that our proposed supervised writer adaptation can achieve significant improvements in recognition accuracy compared with the baseline models: WI and fine-tuned models.

© 2022 THE AUTHORS. Published by Elsevier BV on behalf of Faculty of Computers and Artificial Intel-

ligence, Cairo University. This is an open access article under the CC BY-NC-ND license ([http://creative-](http://creativecommons.org/licenses/by-nc-nd/4.0/)

[commons.org/licenses/by-nc-nd/4.0/](http://creativecommons.org/licenses/by-nc-nd/4.0/)).

1. Introduction

\* Corresponding author at: Department of Information Technology, Cairo Univer- sity, Giza, Egypt.

*E-mail addresses:* [f.alwajih@grad.fci-cu.edu.eg](mailto:f.alwajih@grad.fci-cu.edu.eg) (F. Alwajih), [emostafa@zewail-](mailto:emostafa@zewailcity.edu.eg) [city.edu.eg](mailto:emostafa@zewailcity.edu.eg) (E. Badr), [s.abdou@fci-cu.edu.eg](mailto:s.abdou@fci-cu.edu.eg) (S. Abdou).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

**Production and hosting by Elsevier**

The importance of handwriting recognition technologies has been rapidly increasing over recent years, due in part to the contin- uous advancements in both hardware and software. Most smart- phone and tablets are now equipped with pen input enabled interfaces. Handheld devices have also become more popular and affordable to both students and businesses. Various applications such as note-taking, math typing, and drawing tool applications have found a growing market. Most videotelephony applications have added options for note-taking and handwritten annotation using digital pen capabilities.

Handwriting recognition essentially involves the transcription of handwritten input into digital text. Based on the nature of the

<https://doi.org/10.1016/j.eij.2022.02.007>

1110-8665/© 2022 THE AUTHORS. Published by Elsevier BV on behalf of Faculty of Computers and Artificial Intelligence, Cairo University. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

input, handwriting recognition systems can be classified into either online or offline systems. In online handwriting recognition sys- tems, input is captured through the use of a digital pen or stylus. Input format is a sequence of points. Additional information, based on the type of capturing device, may include timestamps and pres- sure. Due to the temporal nature of the input, recognition of online handwriting can be streamed. On the other hand, in offline hand- writing systems, input is in the form of an image with width and height. Sources of offline input include scanned documents and images extracted from online input. As a result of the nature of off- line input, handwriting recognition in offline systems cannot be streamed. In addition, treatment of input from online and offline handwriting during the preprocessing and features extraction stage is dissimilar. Whilst online handwriting is typically treated as a signal or time series, offline handwriting is treated as an image.

Approaches to online handwriting recognition can be divided into three main categories; segment and decode [[18]](#_bookmark20); prototype- based classifiers [[16,9]](#_bookmark20), and sequence modeling [[8]](#_bookmark39). Sequence mod- eling is the most common approach. A number of techniques have been utilized as part of sequence modeling, including Hidden Mar- kov Models (HMMs) [[23]](#_bookmark21), hybrid Neural Networks (NNs) with HMMs (NNs/HMMs) [[6]](#_bookmark40), hybrid deep neural networks (DNNs) with HMMs (DNNs/HMMs) [[19]](#_bookmark22), and end-to-end (E2E) DNNs [[24,8]](#_bookmark23).

Traditional approaches to sequence modeling, including HMMs and hybrid approaches, typically include multiple components that need to be trained separately, leading to suboptimality. Deep learning E2E systems have allowed for the simultaneous training of all components of the model in an E2E fashion. Various deep learning models of different architectural designs have previously been utilized in the online handwriting recognition task, including CTC-based and Sequence to Sequence (seq2seq) models. CTC-based systems [[24]](#_bookmark23) have been used to recognize and map online hand- writing input frames into letters, sub-units, or words. In this study, a CTC-based model was used in the design of the proposed method. CTC-based models can be incorporated with a Language Model (LM), allowing for improved functionality [[8]](#_bookmark39). For a detailed review, please refer to [[3,36,29]](#_bookmark29).

E2E online handwriting recognition systems need to be trained using large datasets of thousands of handwriting samples. E2E sys- tems have shown encouragingly comparable results to those attained through traditional systems [[4,8]](#_bookmark30). Their performance, however, typically degrades when a WI model is tested against the handwriting of an unknown writer. The degradation is attribu- table to a mismatch between the distributions of training samples and the ‘unseen’ samples of a specific person. Differences in writ- ing styles and devices used may also contribute to such a mis- match. Arabic handwritten script has been traditionally classified into six main styles: Naskh, Thuluth, Nastaliq, Dewani, Reqaa, and Kufi. [Fig. 1](#_bookmark5) illustrates these six styles of traditional Arabic handwritten script. Some of these styles have been more com- monly used in writing than others. Moreover, modern handwritten Arabic script typically shows elements of a number of learned tra- ditional styles, leading to unpredictability. This unpredictability adds another challenge to the design of an effective Arabic hand- writing recognition system.

One solution to reduce the mismatch between training data and unseen samples of a specific writer is to modify the model into a WD model. Writer adaptation is a process of converting the WI model to the WD model, given adaptation data from a specific wri- ter. The main challenge to writer adaptation is having limited data. A WI model with a large number of parameters can easily get over- fitted if adapted with limited adaptation data. Adaptation in deep learning is closely related to such techniques as domain adapta- tion, transfer learning and fine-tuning.

Few approaches have been proposed for writer adaptation in online handwriting recognition systems. They could be classified into two categories: model-based and features transformation- based writer adaptation methods [[26,28,20,2,7,16,34,35]](#_bookmark31). Model- based adaptation methods rely on updating model parameters by optimizing a certain criterion. To that end, a number of techniques have been used, including fine-tuning, linear transformation, regu- larization and subspace methods. On the other hand, features transformation writer adaptation methods have typically depended on features space and typically function through either transforming input features or using auxiliary features [[32]](#_bookmark32). The objective of these methods is to normalize variations in a writer’s handwriting.

Writer adaptation for online handwriting models may be uti- lized in two different settings: supervised and unsupervised learn- ing settings. In the supervised mode, adaptation samples are known to be labeled, while in the unsupervised model, adaptation samples are initially unlabeled and labeling is achieved through the WI model before adaptation is applied.

In this paper, we propose a novel writer model-based adapta- tion method that utilizes a CTC-based model to achieve optimal Arabic online handwriting recognition. Our method employs adversarial multi-task learning (MTL). Adversarial MTL techniques alleviate deep features mismatch between distributions of features generated by the WI model and the WD model. In adversarial learning, two tasks are involved: a primary task of label classifica- tion and a secondary task deep features discrimination. We evalu- ated the proposed method using two datasets; the CHAW and Online-KAHTT datasets. The proposed method has both consis- tently and significantly shown superior performance in compar- ison to the WI and fine-tuned WI models with the same data of target writers.

Contributions made by this work can be summarized as fol- lows: (1) We introduce a new CTC-based method of adaption online handwriting recognition of a specific writer using adversar- ial multi-task learning and a small number of adaptation samples.

(2) We developed a CTC-based model utilizing CNNs and BiLSTMs for Arabic online handwriting recognition, where we utilized two large datasets for training purposes. (3) We evaluated the proposed method using samples of writers from two datasets and bench- marked it against the fine-tuning method.

The paper is organized as follows: Section [2](#_bookmark4) explores previous work done in writer adaptation for online handwriting recognition. In Section [3](#_bookmark6), we explain the architecture of our proposed method. Section [4](#_bookmark13) details the results and provides a comparison of the pro- posed architecture against the original WI with both supervised and unsupervised adaptation settings. Lastly, Section [5](#_bookmark24) details our conclusions and recommendations for possible future work.

1. Related work

Model-based adaptation involves modifying and altering the model parameters or adding new parameters to the model to tune the pre-trained model to a target writer, with only limited adapta- tion data belonging to the specific writer.

Several approaches to model adaptation have been proposed. In [[26]](#_bookmark31), Matic et al. designed a writer dependent system based on a pre-trained Time-Delay Neural Network (TDNN) character-based recognition system. Adaptation is conducted by altering the last layer of TDNN, which acts as an optimal hyperplane classifier. This last layer is retrained to a new writing style. In [[28]](#_bookmark33), authors placed an Output Adaptation Module (OAM) on top of WI neural net- works. OAM is a radial basis function (RBF) network that maps the output of the WI module into a writer-adapted output.

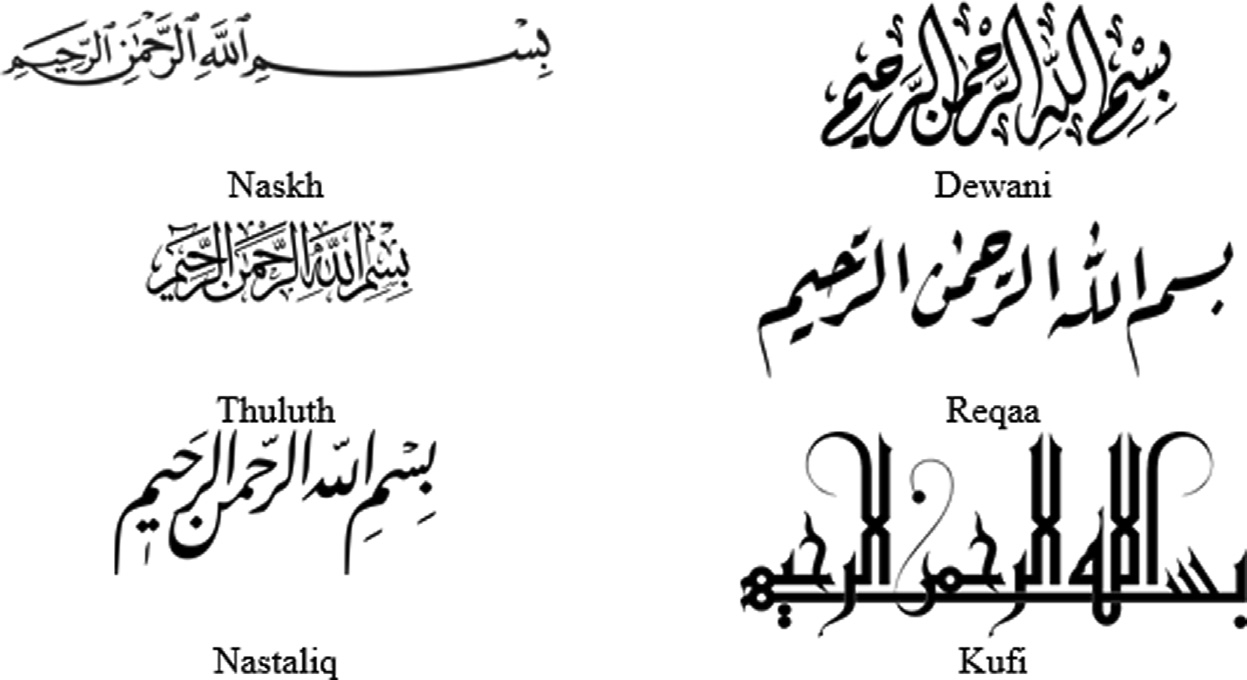


Fig. 1. Traditional Arabic handwriting styles.

The use of regularization can also aide in writer adaptation. In [[20]](#_bookmark25), researchers proposed a writer adaptation method for Support Vector Machines (SVMs) using biased regularization. Adaptation is accomplished by minimizing an adjusted regularized risk functional on adaptation data. Biased regularization provides a trade-off between WI and WD (generic and personalized information). As HMMs had provided remarkable results in online handwriting recognition, some studies utilized adaptation methods from speaker adaptation in Automatic Speech Recognition (ASR) [[2]](#_bookmark34). In [[7]](#_bookmark41), writer adaptation was augmented with a Maximum Likelihood Linear Regression (MLLR) for a HMMs based recognition system. In that approach, MLLR ‘attempts’ to find the best transformation matrix that would maximize the likelihood of the adaptation data. The transformation matrix can then be applied to the HMMs state obser- vation densities to adjust the WI model to the new writer’s data.

Features transformation adaptation, on the other hand, relies on applying transformations to reduce the variations between indi- vidual writers’ data. Transformations can be either applied directly to the features vector or raw handwriting input or could function to extract deep independent features (writer’s code), subsequently incorporating these features into the recognition phase. Huang et al. [[16]](#_bookmark20) utilized Linear Discriminant Analysis (LDA) for writer adaptation of prototype-based methods. They proposed an Incre- mental LDA (ILDA) algorithm for writer adaptation. In their method, the ILDA algorithm was designed to adjust both the LDA transformation matrix and the prototypes.

Zhang et al. [[37]](#_bookmark46) proposed learning a Style Transfer Mapping (STM) paradigm that conducts computations for each individual writer. In STM, the source point set, defined as writer-specific data, is mapped and transformed towards the target point set, defined as the corresponding parameters for the WI classifier. This is achieved through solving a convex quadratic programming problem with the closed-form solution. STM reduces style variations between different writers by projecting their writing onto a style-free space. Consequently, a WI classifier is able to achieve a higher recognition rate. STM can be described as features transformation that is applied in either a supervised, unsupervised or semi-supervised manner. As an extension for STM, Yang et al. [[34]](#_bookmark42) proposed com- plementing the paradigm with an integrated framework for neural network classifiers. In the adaptation, they added a layer to the WI classifier. The adaptation layer weights were updated, and WI model weights were kept fixed. The authors demonstrated that the proposed model outperformed other traditional STM methods, including Learning Vector Quantization (LVQ) and the Modified Quadratic Discriminant Function (MQDF).

Yang et al. [[35]](#_bookmark43) proposed four types of transformations that they specifically designed for CNN layer adaptation. They proposed

that adaptation could be performed deeply on multiple layers of the model, including the CNN and fully connected layers, rather than exclusively on the top layers. The authors referred to their method as Deep Transfer Mapping (DTM), entirely run in an unsu- pervised manner. Adaptation is achieved through the capturing of additional information from different abstraction levels. During the adaption procedure, adaptation layers serve as the linear transformation.

Deep learning methods were also utilized in features transfor- mation adaptation. In [[9]](#_bookmark44), researchers used deep learning features to perform writer adaptation for online Chinese handwriting recognition. In this method, DNNs and CNNs are utilized to extract these features (called tandems). Extracted features are then fed into the prototype-based classifier. A linear transformation of the deeply learned features is used to perform adaptation.

Designers of the aforementioned writer adaptation methods for online handwriting recognition systems primarily utilized tradi- tional techniques such as HMMs, SVM, TDNN and prototype classi- fiers. Despite that a small number of these methods did incorporate deep learning in their features representation, none of their creators tackled the use of E2E model-based writer adaptation. Languages addressed were primarily either Latin or Chinese languages.

E2E online handwriting systems have recently provided remarkable recognition rates [[8]](#_bookmark39) and have become the dominant online handwriting systems in the field. E2E systems rely on deep learning neural networks such as BiLSTMs and CNNs. Most of these systems utilize the CTC algorithm in the training process. CTC allows training on seq2seq problems. There has been extensive research into domains of E2E ASR [[22,36,27]](#_bookmark26) and optical character recognition (OCR) systems [[33,17]](#_bookmark45) which are closely related to online handwriting recognition. In [[22]](#_bookmark26), they investigated two- speaker adaptation methods, the Kullback-Leibler Divergence (KLD) and MTL adaptation, using an E2E ASR CTC-based model. Adversarial based adaptation methods are mainly imported from the domain adaptation area [[10,11]](#_bookmark47). To the best of our knowledge, our proposed model is the first E2E online handwriting recognition model that examines writer adaptation for Arabic online handwrit- ing. In this work, we introduce a regularization technique as a model-based writer adaptation for online handwriting recognition task using adversarial learning.

1. Writer Adaptation for End-to-End CTC based models

In this work, we propose a novel model-based writer adaptation method utilizing a CTC-based model and an adversarial MTL. In adversarial learning, a primary task of label classification and a secondary task of deep features discrimination are defined. WD

is created and initialized by cloning WI, then divided into features extraction and label classification sub-networks. In the secondary task, a discriminative sub-network is attached on top of the fea- tures extraction sub-network in parallel with a label classification sub-network of the primary task. The two tasks are jointly opti- mized in a minimax manner with multi-task learning, where fea- tures extraction layers are shared among the tasks. The goal of a minimax game is to minimize label classification loss while jointly maximizing features discrimination loss. By the end of this game, the regularized WD model should reach a higher recognition rate against the test data of the target writer than that of the WI model. The WI is kept fixed during adaptation and discarded with the dis- criminative sub-network during the test.

In the following section, we describe the CTC-based model used in this work, as well as the Generative Adversarial Networks (GANs) and Domain Adaptation Neural Networks (DANNs). We then describe the writer adversarial training for online handwrit- ing writer adaptation.

* 1. *CTC based online handwriting recognition system*

We used a combination of CNN, BiLSTM and fully connected layers to build our model architecture, as shown in [Fig. 2](#_bookmark9). During training, CTC loss is utilized as an objective function [[14,24]](#_bookmark20) for the optimization of the prediction of the labels’ sequence. Typi- cally, the length of the output sequence for handwriting recogni- tion is less than the length of the handwriting input. In CTC, an additional blank label e is introduced to ensure that the length of outputs remains the same, and to allow for repeated labels in the output.

The CTC objective function *LCTC* is defined as the summation of negative log conditional probabilities of the correct labels, as shown in Eq. [1](#_bookmark7) below. In this summation, *X* = *x*1, *x*2, .., *xT* , where

*X* is the handwriting input sequence with length *T*, *Y* = *l*1, *l*2, ... , *lL*

is denotes the label sequence with length *L*, *Theta* denotes model

weights, *p* denotes the CTC single path, and *B*—1(*L*) denotes as all possible CTC paths generated from original *L* labels’ sequence.

*LCTC* = —*lnP*H(*Y*|*X*)= —*ln P*H(*p*|*X*) (1)

X

*p*∈*B*—1(*L*)

The CTC assumes that output units are conditionally indepen- dent. Hence, *P*H(*p*|*X*) can be expressed as the product of posteriors from each time step t:

*T*

Y

*P*H(*p*|*X*)= *P*H(*pt* |*xt*) (2)

*t*=1

where *pt* is the output unit at time *t*, *xt* is the input handwriting at time *t*. The posterior probability of the labels at each time step is computed using CNN-BiLSTM deep network (*encoder*). The encoder outputs *henc* = (*henc*1, .., *hencT* ), are treated as logits and are passed to the softmax layer to compute the probability distribution over labels *l*. An estimation of model parameters H is then made by max- imizing Eq. [1](#_bookmark7) using gradient descent.The gradients required in gra- dient descent can be calculated using the forward-backward algorithm [[15]](#_bookmark20).

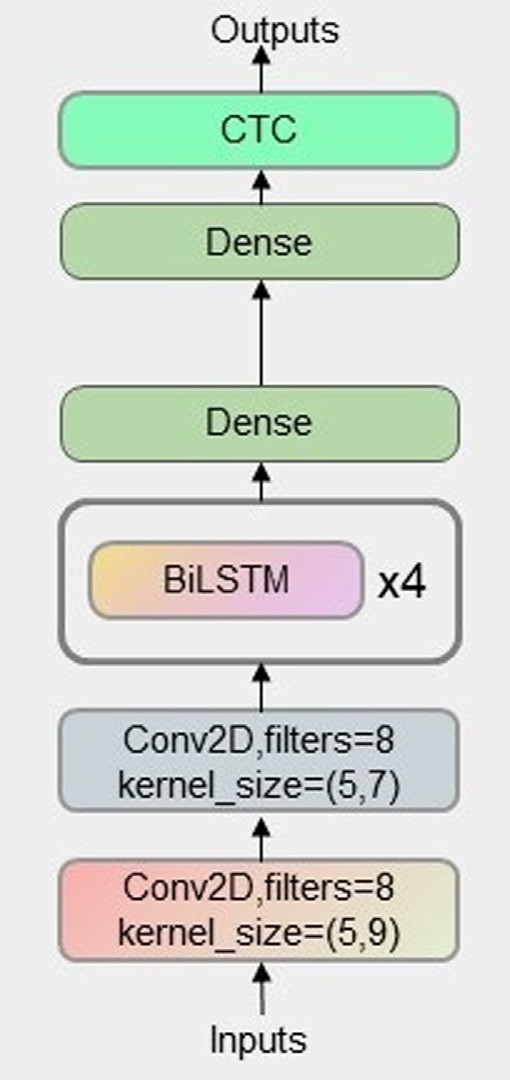
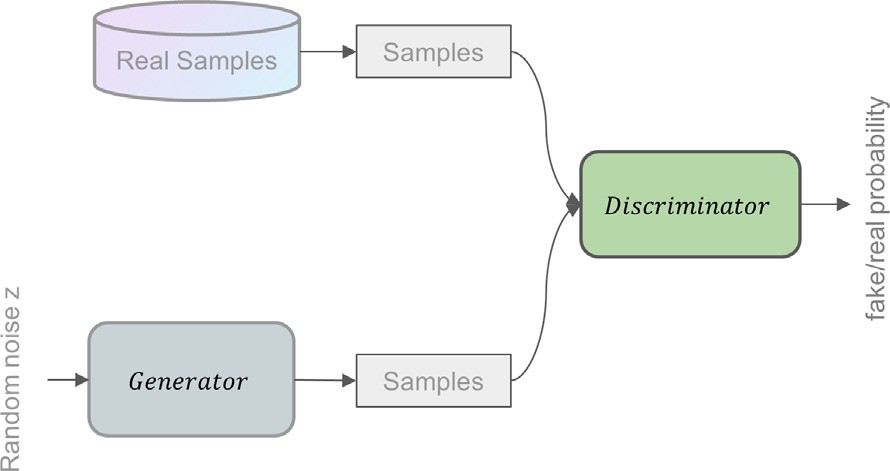
To find the most probable sequence of labels given a handwrit- ing input, greedy decoding is utilized [[14]](#_bookmark20). Greedy decoding involves two steps. The first step is concatenating the most proba- ble label for every time step. The second step is removing the duplicate labels and all blanks.

* 1. *Generative Adversarial Networks (GANs)*

GANs are a form of neural network architecture that allows deep learning models to learn and capture the training data distri- bution, allowing for the generation of new data instances from learned distribution [[12]](#_bookmark20). Typically, GANs consist of two models; a discriminator model and a generator model, as shown in [Fig. 3](#_bookmark10). The discriminator is a binary classifier that functions to distinguish real instances from fake instances generated by the generator. The generator is, on the other hand, the part of GANs that generates fake instances with the same distribution as that of the training data.[4](#_bookmark11).

Assuming that *x* is a sample instance that represents input data, then *D*(*x*) is the discriminator. *D*(*x*) is a binary classifier that pro- duces a probability with *x* as a real instance from the training data. *D*(*x*) should output a high probability with respect to real instances and a low probability with respect to synthesized instances. Simi- larly, assuming that *z* is a latent vector sampled from Gaussian dis- tribution, then *G*(*z*) is the generator that transforms *z* to the data space.

The main aim of *G* is to produce fake samples from estimated distribution (*pg* ) by capturing the training data distribution(*pdata* ). According to [[12]](#_bookmark20), *G* and *D* play a minimax game. *G* attempts to minimize the probability that *D* would classify its synthesized out-

Fig. 2. The architecture of the E2E system is used to train Arabic online handwriting  recognition. The proposed CTC-based model consists of CNN layers followed by

BiLSTM layers and fully connected layer. Fig. 3. General architecture of Generative Adversarial Networks (GANs).

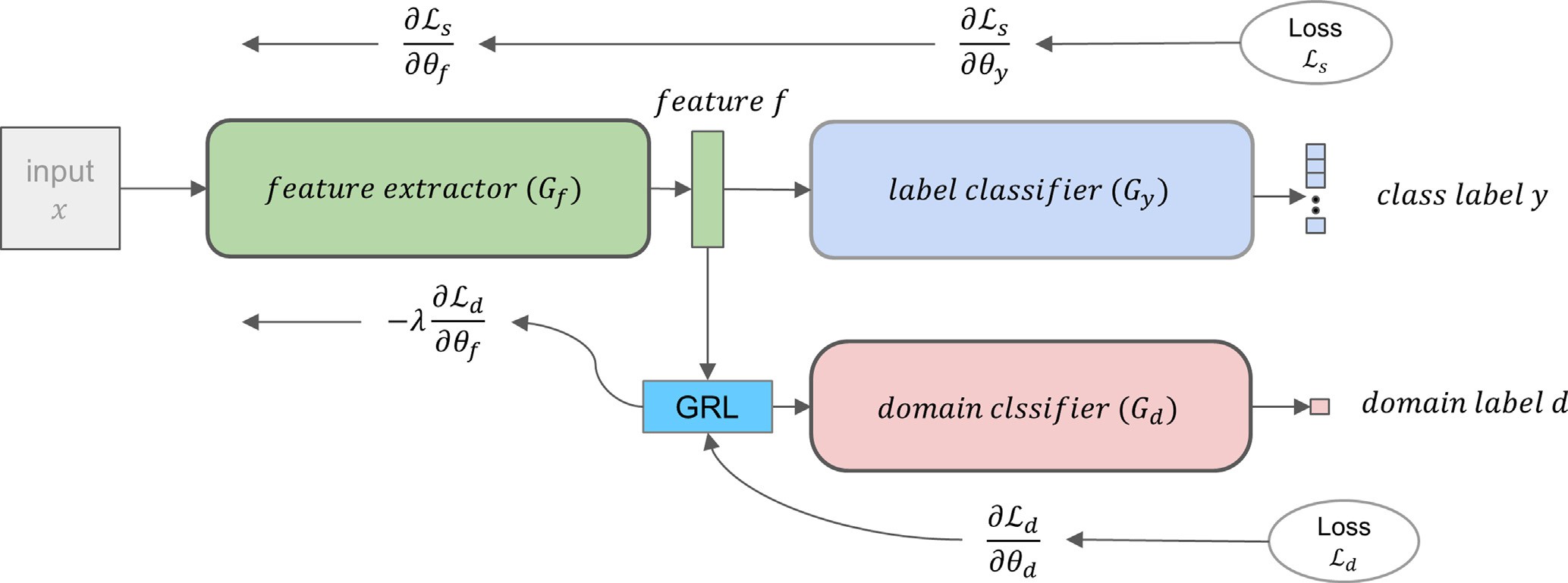


Fig. 4. The general architecture of DANN is composed of three main sub-networks: a features extractor sub-network (*Gf* ), a label classifier (*Gy*) sub-network, and a domain classifier (*Gd*) sub-network. The figure was adapted from [[11]](#_bookmark20).

puts as fake instances (*log*(1 — *D*(*G*(*x*)))), whilst *D* attempts to max- imize instances of correctly classifying synthesized and realistic samples (*logD*(*x*)). According to [[12]](#_bookmark20), the GAN loss function may be expressed as follows:

minmax*V* (*D*, *G*) = E*x*~*pdata* (*x*)[*logD*(*x*)]

(*h*^*f* , *h*^*y* )= argmin *E*(*hf* , *hy*, *h*^*d* ) (7)

(*hf* ,*hy*)

(*h*^*d*)= argmax *E*(*h*^*f* , *h*^*y*, *hd*) (8)

(*hd* )

*G D*

+E*z*~*pz* (*z*)[*log*(1 — *D*(*G*(*z*)))]

(3)

Applying stochastic gradient descent or one of its variants, the above DANN objective yields the following updating rules:

Theoretically, the optimal solution for this minimax competi-

*h* = *h* — *a* @L*y* — *k* @L*d* (9)

tion is attained when the estimated distribution by the generator

*G* equals the training data distribution, and the discriminator can- not distinguish between real and fake inputs. In practice, GANs

*f f* @*hf*

@L*y*

@*hf*

models are not trained up to this point.

* 1. *Domain Adaptation Neural Networks (DANNs)*

*hy* = *hy* — *a* (10)

@*hy*

—*k*@L*d*

DANNs are deep domain adaptation networks that mainly rely

*hd* = *hd* + *a*

@*hd*

# (11)

on the representation learning (RL) technique, where both domain invariance and discriminative properties are considered during the learning process [[11]](#_bookmark20).

The goal of DANNs is to learn a classifier */* , which can be broken down into two parts, */* = *Gf* ◦ *Gy*, where *Gf* is a features extractor that represents underlying features and *Gy*is a label classifier that predicts the target labels. Both *Gf* and *Gy* are used during training and testing and optimized by minimizing the following equation:

As a typical training phase in NNs, gradients are calculated by differentiating losses with respect to parameters using standard back-propagation. In DANNs, the authors in [[11]](#_bookmark20) proposed GRL for the backward pass.

* 1. *Writer adaptation using Adversarial Training with MTL*

To perform the writer adaptation task, we start with a sequence

of features *X* = *x*1, *x*2, ... , *xT* , target labels *Y* = *y*1, *y*2, ... , *yL* ,and a

1

X

L (*h* , *h* )= ‘(*G* (*G* (*x*)), *y*) (4)

*y*

*f*

*y*

*Ns*

*y*

*f*

(*x*,*y*)∈*Ds*

In addition, DANNs architecture proposes a domain classifier *Gd*, which is branched after features representation *Gf* to distinguish between the target and source domains. *Gd* is used during the training phase and thrown after training. It is optimized jointly with *Gf* and *Gy* by maximizing the following function:

trained WI model H*WI*. As shown in [Fig. 2](#_bookmark9), we spilt WI into two sub-networks *Gf* and *Gy* . *Gf* acts as a features extractor, map- ping each input vector to its representation. *GWI*sub-network acts as a classifier to predict posterior probabilities of classes given the representation from *GWI*. We then construct our WD model by copying the WI model with the same architecture. As a result,

we have *GWD* = *GWD*.*GWD* initialized from H*WI*. *GWD* architecture is

*y*

*f*

*WI*

*WI*

*WI*

*f y*

L*d*(*hf* , *hd*) =  X‘(*Gd*(*Gf* (*x*)), *s*)

1

*Ns*

*x*∈*Ds*

trained through the minimization of the following CTC loss

5) objective:

+ 1 X‘(*G* (*G* (*x*)), *t*) ( 0 1

*Nt d f*

*WD WD*

*WD WD*

*x*∈*Dt*

*L*CTC(H , H )=— log @ X *p*(H , H )(*p*|x)A (12)

Total training loss may be expressed as follows:

*E*(*hf* , *hy*, *hd*)= L*y*(*hf* , *hy*)— *k*L*d*(*hf* , *hd*) (6)

where is the trade-off between and . The parameters

*k*

L*y*

L*d*

*hf* , *hy*

*f y f y*

*p*∈B—1(L)

*p*(H*WD*, H*WD*)(*p*|x)= Y*p*(H*WD*, H*WD*)(*pt* |xt) (13)

*f*

*y*

*t*=1

*f*

*y*

*T*

and *hd* are obtained by optimizing the following equations:

An important goal is ‘force’ the output of *GWD* to be closely resemble the output of *GWI* for adaptation samples of the target

*f*

*f*

writer. To reach this goal, an additional discriminator binary clas- sifier *Gd* is introduced. *Gd* accepts input form both *GWD* and *GWI*.

one. As CTC deals with mapping an input sequence to an output sequence in the primary task, we added max and average pooling layers to the discriminator network. These layers aggregate the variable-length hidden vector into two fixed hidden vectors to

*f f* the next dense layer.

Adversarial training of GW D and Gd is then conducted. During this training, *Ldisc* is minimized with respect to *hd* and maximized *Ldisc* with respect to *hWD*. This minmax objective rivalry would

*f*

improve the ability of *GWD* to produce features with a distribution closer to the features distribution produced by *GWI*, and would improve the discriminator capability of *Gd*, allowing it to better

*f*

*f*

to distinguish between features generated by *GWI* versus *GWD*.

1. Experiments and results

We performed writer adaptation for an Arabic online handwrit- ing recognition task. We utilized two large datasets for training and testing of our end-to-end CTC-based model. For the adaptation task, individual writer samples were chosen from the test sets.

*f f*

Eventually, the minimax competition would converge to the point where *GWD* produces highly perplexing features that *Gd* is unable to differentiate whether these features are produced by *GWD* or *GWI* for the same sample. Reaching convergence, the WD

*f*

*f*

*f*

* 1. *Datasets*

We trained and validated the WI and WD models using two well-known Arabic online handwriting datasets; the CHAW [[19]](#_bookmark22)

model would effectively be regularized and would have an excel- lent capability to generalize with respect to target writer data, without diverging too much from the WI model. *Gd* is trained by maximizing the standard cross-entropy loss as follows:

L*disc*(H*WD*, H*WI*, H*d*) = 1 X‘(*Gd*(*GWD*(*xt*)); H*WD*, H*d*)

*T*

and Online-KHATT [[25]](#_bookmark28) datasets. The CHAW comprised data col- lected from 1250 writers and was proposed by Cairo University, whilst the Online-KHATT comprised data collected from 623 writ- ers and was proposed by King Fahd University; a total of 1873 writers from both datasets. Writers varied in gender, country of

origin, age, handedness, and level of education. Both datasets con-

*f f T*

*y f*

*t*=1

*T*

# (14)

tained natural writing with unrestricted writing styles. This varia- tion attests to the reliability of datasets. The test set for both

+ 1 X‘(*Gd*(*GWI*(*xt*)); H*WI*, H*d*)

*T*

*t*=1

*y*

*f*

datasets consisted of 240 writers. [Table 1](#_bookmark15) compares and summa-

rizes the both datasets.

In this context, WD and *Gd* models are jointly trained through adversarial MTL training. In the MTL setting, there is the primary task of label classification and the secondary task of featuress dis- crimination. Features discrimination functions to predict if features

are produced by *GWD* or *GWI*. The primary and the secondary tasks

* 1. *Writer independent model setup*

We used a CTC-based model with CNN and BiLSTM neural net- works as the WI model for the Arabic online recognition system in

*f f*

are optimized adversarially using the following objective function:

L*total* (H*WD* , H*WD* , H*d*) = L*CTC* (H*WD* , H*WD* )

our experiments. We chose 20 handwriting features described for

each point described in [[24,23,4]](#_bookmark23) as the input features. Our design was included of two CNN layers followed by 4 BiLSTM layers with

*f y f y*

—*k*L*disc* (H*WD* , H*WI* , H*d*)

*f f*

# (15)

80 units each. Finally, two fully connected layers were added at the point where the output layer had a total of 160 units, needed due

H^ *WD* , H^ *WD* = arg min L*total*(H*WD*, H*WD*, H^ *d*) (16)

*f y f y*

H^ *d* = arg min L*total*(H^ *WD* , H^ *WD* , H*d*) (17)

to the nature of Arabic script. The Arabic alphabet is comprised of 28 letters shaped differently according to their position in a word. We thus expanded each letter into four distinct ‘position-

dependent letters’ (isolated, start, middle, and end). This expansion

*f y*

resulted in 160 classes in the output layer. In decoding, we utilized

where *k* is a hyperparameter which controls the trade-off between

*LCTC* and *Ldisc* during the optimization, H^ *WD* , H^ *WD* , and H^ *d* are param-

the simplest greedy CTC decoder to select the highest probability over characters distribution. During the training phase (to increase

*f y*

eters to be optimized. H*WI* is used to generate the features and set to be fixed during the training process along with H*WI*. In addition,

*f*

*y*

H^ *WD* , H^ *WD* , H^ *d* parameters are updated during training via back-

network robustness), dropout implied dropping random neurons [[30]](#_bookmark35). We utilized dropout in our WI model with a rate of 0.3 for all layers. Batch normalization (BN) is another technique used to

accelerate the convergence of NNs and improve the capability of

*f y*

propagation algorithm with gradient descent, expressed as follows:

*hWD* ← *hWD* — *a* @L*CTC* — *k* @L*disc* ! (18)

the network to generalize [[5]](#_bookmark36). In our model, a BN layer was placed after each CNN layer in order to increase speed during training and attain better recognition accuracy during testing. WI was trained

*f f*

*hWD* ← *hWD* — *a*

*y y*

@*hWD*

*f*

*f*

@L*CTC*

@*hWD*

*y*

@*hWD*

# (19)

with CTC loss using Adam Optimizer [[21]](#_bookmark27). TensorFlow 2.2 [[1]](#_bookmark37) was used for the purposes of building, training, and testing all models.

* 1. *Writer adaptation model setup*

*hd* = *hd* + *a* —*k*@L*disc*

@*hd*

# (20)

For our writer adaption model, we used the writer adaptation method described in Section [3.4](#_bookmark12). and the WI trained model

where *a* is the learning rate. GRL then allow for adversarial learning

in the back-propagation phase. Upon completion of training, both the secondary task network *Gd* and fixed WI network are discarded. We keep the adapted primary task network for testing.

While the architecture of the primary task is sequence to sequence, the secondary task model architecture is a sequence to

described in Section [4.2](#_bookmark14) in order to create the same architecture and initialize the weights of the WD model. The CNN, BiLSTM, and dense layers constituted the features extractor sub-network. The last dense layer constituted the label classifier sub-network, as shown in [Fig. 5](#_bookmark16). The WI features extractor sub-network was used during the adaptation process to generate WI features from

Table 1

Comparison between CHAW and Online-KATHH datasets.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Dataset |  | CHAW |  |  |  | Online-KHATT |  |  |
|  | Train set |  | Test set |  | Train set |  | Test set |
| Number of writers | 1146 |  | 104 |  | 623 |  | 136 |  |
| Number of samples Count of sentences  Count of words | 180k –  180k |  | 12k –  12k |  | 6974  6974 (lines)  56547 |  | 1533  1533 (lines)  12100 |  |
| Size of vocabulary | 17800 |  | 500 |  | 19437 |  | 5887 |  |

adaptation samples. However, WI features extractor sub-network weights were kept fixed during the optimization process. We also defined the discriminator model (sub-network) as two BiLSTM lay- ers with 64 units each, followed by average and max-pooling lay- ers. A dense output layer with sigmoid activation was added. We attached the GRL layer to the front of the discriminator network, as illustrated by [Fig. 6](#_bookmark17). The discriminator network was set to receive input from the WI features extractor sub-network and the WD features extractor sub-network and to attempt to distin- guish between these outputs. As the structure of the discriminator model followed the sequence to one architecture, we employed a pooling layer in order to aggregate the output sequence of the BiLSTM layers - in both the WI and WD features extractor sub- networks - as a fixed vector and then pass it onto the dense output layer.

The dropout, which was used in the training WI model, was also utilized during adaptation training. BN layers weights were kept frozen during adaptation training. We employed early stopping regularization in order to prevent overfitting of the adapted models.

For supervised adversarial writer adaptation, we used the labels provided with handwriting samples in the datasets, while during unsupervised adversarial writer adaptation, we used the WI to label adaptation samples. We selected two writers from each data- set to assign and pick up the best hyperparameters for adaptation

training. Random search was conducted to find the best hyperpa- rameters (batch size, learning rate, dropout rate, and *k*). The results of this assignment were then used for all writers.

* 1. *Evaluation*

For the purposes of evaluating our method, we used standard metrics character error rate (CER) and word error rate (WER). We used the same handwriting samples from the Online-KHATT and CHAW datasets (see Section [4.1](#_bookmark13)) that were used to train the base- line model. Samples used in the adaptation process were not used in the training of the WI model.

For evaluation, selected handwritings of 84 writers from the test sets of both datasets; 42 writers from each dataset. For each writer, we split his/her samples into training and test sets. From the CHAW dataset, we selected 40 samples (words) for training and 80 samples for testing (to be used in all experiments). The online-KHATT dataset is comprised of sentence-based samples. From that dataset, we choose four samples per writer for adapta- tion training and an average of 10 samples for testing.

To evaluate our proposed adaptation method against other methods, we ran two additional experiments on a WI model and a fine-tuned model for each writer fine-tuned with the writer’s training data. First, we ran WI for both datasets. Values of 13.21% CER, 41.50% WER, 16.84% CER and 56.05% WER were achieved

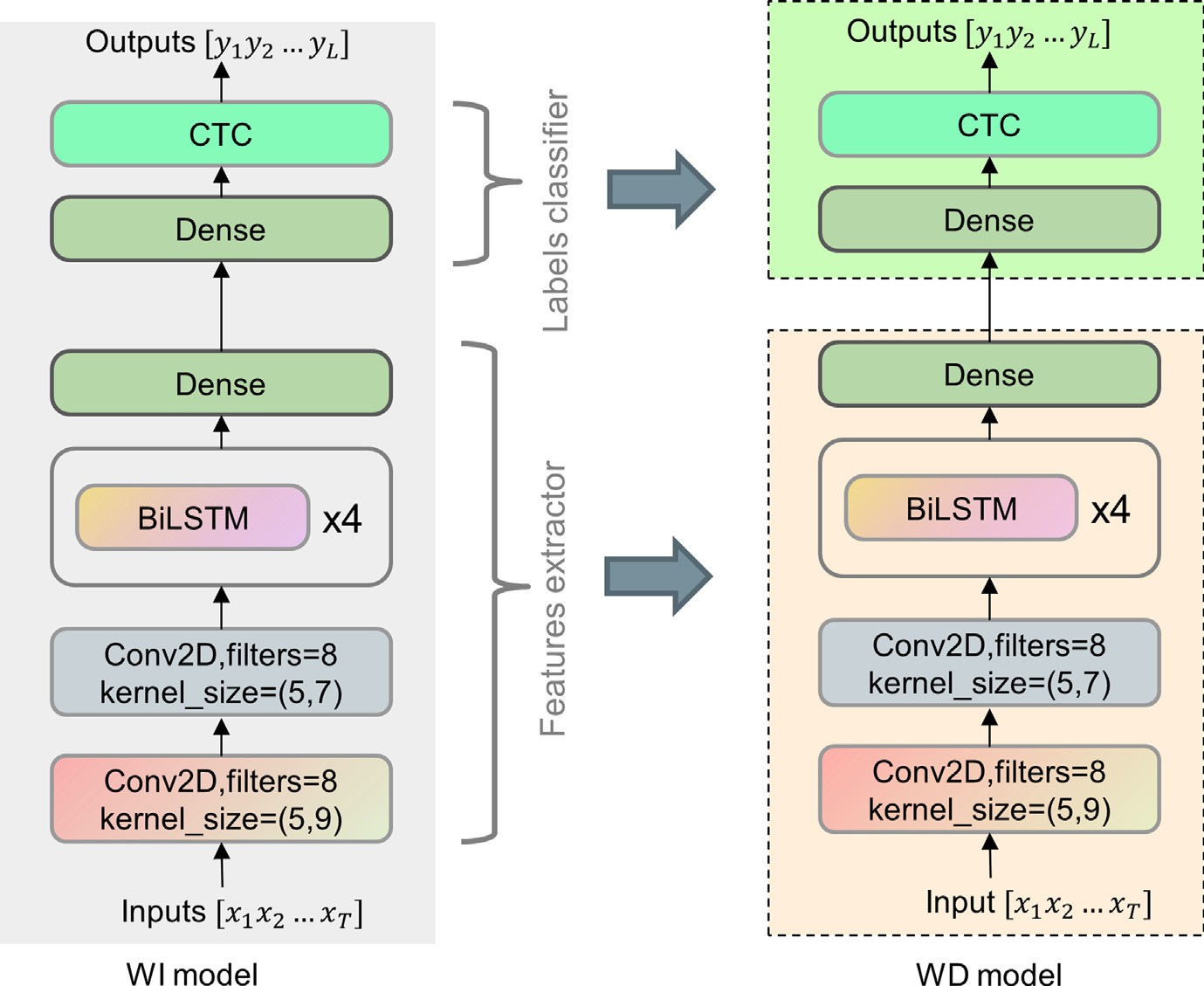


Fig. 5. To the left is the WI model, divided into a features extractor and label classifier. To the right is the WD model created and initialized from the trained WI model; also divided into a features extractor and label classifier.

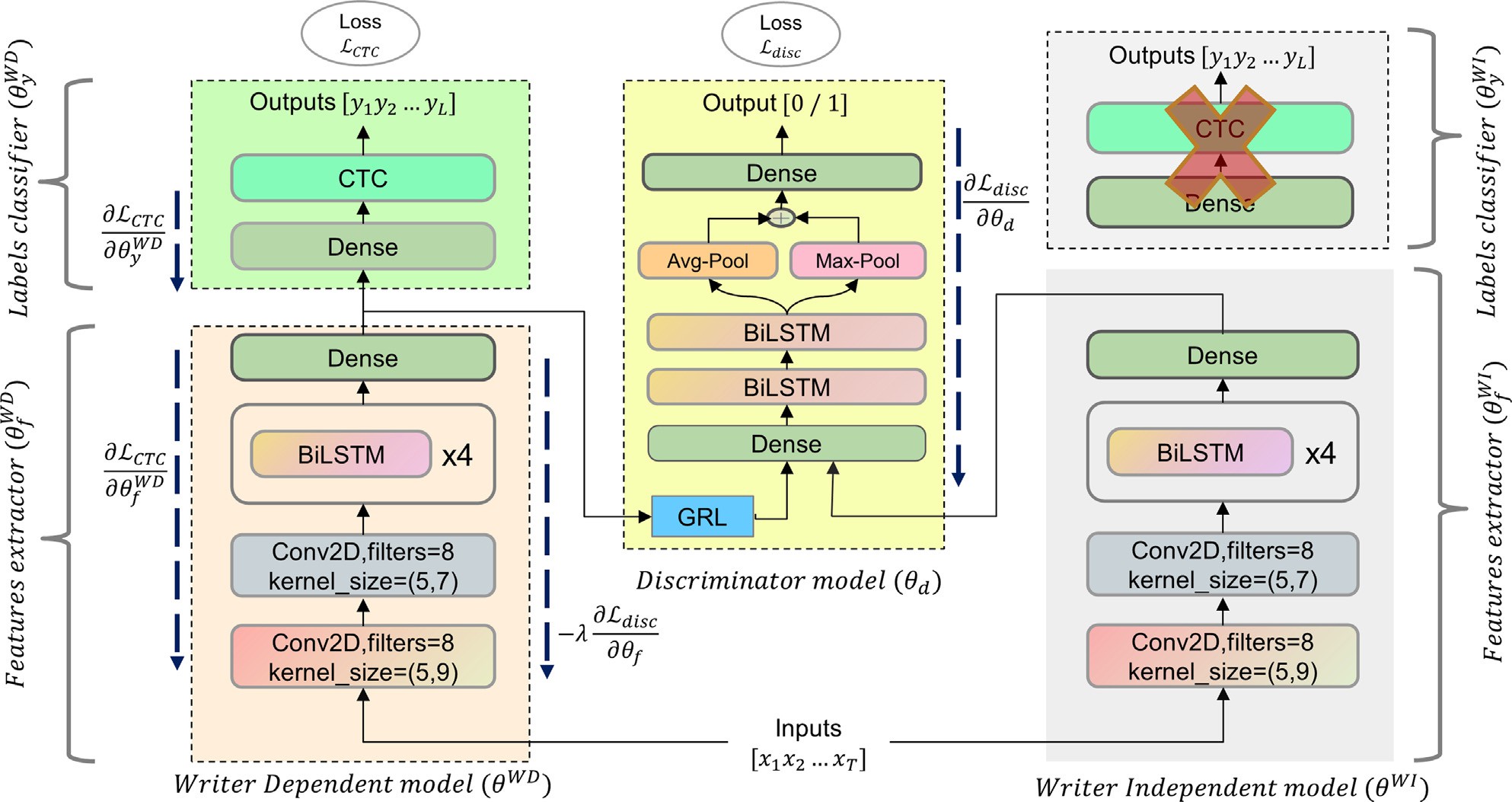


Fig. 6. The architecture of the adversarial writer adaptation. On the left is the Writer Dependent (WD) model *hWD* to be adapted. In the middle is the discriminator model *hd*

that distinguishes between features generated by the *hWI* versus the *hWD* sub-networks. Finally, on the right is the Writer Independent (WI) model *hWI* , used to generate WI

*f f*

features of the input by *hWI* . The *hWD* model and discriminator *hd* are optimized adversarially, while the *hWI* is kept fixed during adaptation. The *hWI* sub-network is discarded

*f f y*

during the training and the *hWI* and the *hd* are discarded after the training.

*f*

against the CHAW and Online-KHATT datasets, respectively. Simi- larly, we applied fine tuning to the WI model as a second baseline. Average result over all writers stood 12.77% CER, 39.62% WER 15.30% CER, 48.04% WER against the CHAW and Online-KHATT datasets, respectively ([Table 2](#_bookmark18)).

* 1. *Discussion*

An important finding during adaption involved the level of reductions in WERs. Reductions in WERs were higher than CERs both during adaptation ([Table 2](#_bookmark18) and [Fig. 7](#_bookmark19)) and in the WI model. This may attributable to the fact that the WI model has the capabil- ity to recognize most of the target characters, but the result may contain some missed, misspelled or disordered words that cause higher WERs. Supervised adaptation helps in learning to overcome these errors and improves some word character ordering as well.

The Arabic script is comprised of 28 main characters, each of which is written differently according to their position in a word. In our CTC-based model, we expanded each charterer to 4 different ‘position-based’ characters (isolated, start, middle, and end). In addition, we added additional special Arabic characters such as the ‘lam-alf,’ as well as number figures and punctuation marks, cre- ating a total of 160 units in the output layer. Expanding recognition of a character to include its shape in multiple positions contributes

to the improvement of results compared with the use of the basic characters only. In Arabic script, character shape not only differs according to position in a word, but is also influenced by both the preceding and succeeding letters. Character with position car- ries contextual information which helps in the decoding phase [[4]](#_bookmark30). Adaptation training data would not however cover all 160 target units for each individual writer. The proposed adaptation method works on features representation and performs the distribution shift based on the features detected. Thus, our model is able to sat- isfactorily generalize, despite possibly not covering all target units of a specific writer.

[Fig. 7](#_bookmark19) details a comparison of the performance of the four mod- els for each writer data from both datasets in terms of CER and WER. Evidently, the use of supervised and unsupervised adaptation constantly improved the recognition rate compared with WI and fine-tuned models for each writer. General robustness of proposed methods in relation to different handwriting styles was evident from most results. Two exceptions involved one writer from the CHAW dataset, and one from the Online-KHATT dataset. The writ- ers wrote their writing samples with multiple lines, which limited the ability of the WI model to recognize letters/words. In both cases, adaptation was not effective.

Fine tuning gives better results than unsupervised adaptation. Unsupervised adaptation depends on the quality of the WI model

Table 2

The WER (%) and the CER (%) (lower is better) performance of WI, fine-tuned, supervised adversarial adaptation, and unsupervised adversarial adaptation models.

System Dataset

CHAW Online-KHATT

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | CER[%] | WER[%] |  | CER[%] | WER[%] | Avg. CER[%] | Avg. WER[%] |
| Writer Independent model | 13.21 | 41.50 |  | 16.84 | 56.05 | 15.02 | 48.77 |
| Fine tuning (*k* = 0) | 12.77 | 39.62 |  | 15.30 | 48.04 | 14.03 | 43.83 |
| Supervised adversarial adaptation (*k* = 3) | 11.97 | 35.22 |  | 14.42 | 38.90 | 13.20 | 37.06 |
| Unsupervised adversarial adaptation (*k* = 3) | 12.94 | 40.74 |  | 16.12 | 53.36 | 14.53 | 47.04 |



Fig. 7. Achieved CERs and WERs across different models (WI, fine-tuned, supervised adversarial adaptation, and unsupervised adversarial adaptation models) per writer for CHAW and Online-KHATT datasets. (a) CERs of CHAW dataset. (b) WERs of CHAW dataset. (c) CERs of Online-KHATT dataset. (d) WERs of Online-KHATT dataset.

which is used to generate labels for unlabeled adaptation data. In case of labels availability, it is recommended to go with supervised adaptation.

In all our experiments, we used greedy decoding for the model outputs. It selects the most probable character over characters dis- tribution for each time step as described in Section [3.1](#_bookmark8). Using decoding techniques like beam search decoder with LM or dic- tionary integration could improve the recognition rate. In our

study, we primally focused on improving upon model and the use of a greedy decoder that would be capable of highlighting improvements. Integration with LMs can be done after writer adaptation feasibility in the production systems.

Adaptation methods could improve baseline online handwriting recognition models. Moreover, investigating advanced sequence models like seq2seq with attention and self-attention transformers

[[31]](#_bookmark38) can boost the result of baseline models before adaptations in

case of data availability. Then, we can apply adversarial multi-task learning adaptation to these models in the same manner as was done in this work. In addition, data augmentation is used to increase the training data. Thus, synthetic data generation [[13]](#_bookmark20) for writers with style transfer would increase the adaptation sam- ples for writers, resulting in increased overall performance.

1. Conclusions

In this article, we presented a new method of model-based wri- ter adaptation for CTC-based online handwriting recognition using adversarial multi-task learning. In this work, WD was cloned from WI and initialized by WI’s weights. WD was then split into a fea- tures extractor and a label classifier. A GRL layer with a discrimina- tor model was placed after the features extractor in parallel with the label classifier. Adversarial multi-task learning was used to ensure the distribution of deep features generated by the WD was similar to the distribution of deep features generated by the fixed WI. Experimental results indicate that the proposed super- vised writer adaptation constantly improves the performance of adapted models over two baseline models; a WI and a fine-tuned model. In unsupervised writer adaptation, experimental results showed improvement over WI and degradation in comparison with fine-tuned models.

Our work is the first work that tackles writer adaptation for online handwriting recognition for E2E models. It will contribute to opening more doors for Arabic language applications which could be used in different domains. Also, The proposed method can be applied to other languages as well such as Latin languages. Adversarial multi-task learning adaptation is not limited to CTC- based models and it can be applied to other E2E models in the same manner such as attention and self-attention models. In the future, we will explore the possibility of applying adversarial multi-task learning on attention-based models. In addition, we will explore the effects of synthesized adaptation training data on the performance of adaptation.

Declaration of Competing Interest

The authors declare that they have no known competing finan- cial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems..
2. [Abdelaziz I, Abdou S, Al-Barhamtoshy H. A large vocabulary system for arabic](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0010) [online handwriting recognition. Pattern Anal Appl 2016;19:1129–41](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0010).
3. [Al-Helali BM, Mahmoud SA. Arabic online handwriting recognition (aohr): A](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0015) [survey. ACM Comput. Surv. 2017;50](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0015).
4. [Alwajih F, Badr E, Abdou S, Fahmy A. Deeponkhatt: An end-to-end arabic](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0020) [online handwriting recognition system. Int J Pattern Recogn Artif Intell 2021](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0020).
5. [Amodei D, Ananthanarayanan S, Anubhai R, Bai J, Battenberg E, Case C, et al.](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0025) [Deep speech 2: End-to-end speech recognition in english and mandarin.](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0025) [International conference on machine learning, PMLR 2016:173–82](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0025).
6. Bengio Y, LeCun Y, Nohl C, Burges C. Lerec: A nn/hmm hybrid for on-line handwriting recognition. Neural Comput 1995;7:1289–303. doi: [https://doi.](https://doi.org/10.1162/neco.1995.7.6.1289) [org/10.1162/neco.1995.7.6.1289](https://doi.org/10.1162/neco.1995.7.6.1289).
7. Brakensiek A, Kosmala A, Rigoll G. Comparing adaptation techniques for on- line handwriting recognition. Proceedings of Sixth International Conference on Document Analysis and Recognition 2001:486–90. doi: [https://doi.org/](https://doi.org/10.1109/ICDAR.2001.953837) [10.1109/ICDAR.2001.953837](https://doi.org/10.1109/ICDAR.2001.953837).
8. [Carbune V, Gonnet P, Deselaers T, Rowley HA, Daryin A, Calvo M, Wang LL,](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0040) [Keysers D, Feuz S, Gervais P. Fast multi-language lstm-based online](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0040) [handwriting recognition. International Journal on Document Analysis and](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0040) [Recognition (IJDAR) 2020:1–14](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0040).
9. [Du J, Zhai JF, Hu JS. Writer adaptation via deeply learned features for online](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0045) [chinese handwriting recognition. International Journal on Document Analysis](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0045) [and Recognition (IJDAR) 2017;20:69–78](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0045).
10. Ganin Y, Lempitsky V. Unsupervised domain adaptation by backpropagation. In Proceedings of the 32nd International Conference on International

Conference on Machine Learning – Volume 37, JMLR.org; 2015. pp. 1180– 1189..

1. [Ganin Y, Ustinova E, Ajakan H, Germain P, Larochelle H, Laviolette F, et al.](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0055) [Domain-adversarial training of neural networks. J Mach Learn Res](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0055) [2016;17:2030–96](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0055).
2. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S. Generative adversarial nets. In Ghahramani Z, Welling M, Cortes C, Lawrence N, Weinberger KQ. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.; 2014. URL: [https://proceedings.neurips.cc/paper/](https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf) [2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf](https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf)..
3. Graves A. Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850..
4. [Graves A, Fernández S, Gomez F, Schmidhuber J. Connectionist temporal](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0070) [classification: labelling unsegmented sequence data with recurrent neural](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0070) [networks. In: Proceedings of the 23rd international conference on Machine](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0070) [learning. p. 369–76](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0070).
5. [Graves A, Jaitly N. Towards end-to-end speech recognition with recurrent](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0075) [neural networks. International conference on machine learning, PMLR](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0075) [2014:1764–72](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0075).
6. [Huang Z, Ding K, Jin L, Gao X. Writer adaptive online handwriting recognition](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0080) [using incremental linear discriminant analysis. In: 2009 10th International](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0080) [Conference on Document Analysis and Recognition, IEEE. p. 91–5](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0080).
7. [Kang L, Rusinol M, Fornés A, Riba P, Villegas M. Unsupervised writer](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0085) [adaptation for synthetic-to-real handwritten word recognition. In:](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0085) [Proceedings of the IEEE/CVF Winter Conference on Applications of Computer](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0085) [Vision. p. 3502–11](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0085).
8. [Keysers D, Deselaers T, Rowley HA, Wang LL, Carbune V. Multi language online](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0090) [handwriting recognition. IEEE Trans Pattern Anal Mach Intell](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0090) [2017;39:1180–94](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0090).
9. [Khaled O, Fahmy A, Abdou S. Large vocabulary hybrid dnn/hmm arabic online](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0095) [handwriting recognition system. In: 2017 4th IAPR Asian Conference on](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0095) [Pattern Recognition (ACPR). p. 876–81](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0095).
10. [Kienzle W, Chellapilla K. Personalized handwriting recognition via biased](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0100) [regularization. In: Proceedings of the 23rd international conference on](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0100) [Machine learning. p. 457–64](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0100).
11. Kingma DP, Ba J. Adam: A method for stochastic optimization; 2014. arXiv preprint arXiv:1412.6980..
12. [Li K, Li J, Zhao Y, Kumar K, Gong Y. Speaker adaptation for end-to-end ctc](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0110) [models. 2018 IEEE Spoken Language Technology Workshop (SLT) 2018:542–9](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0110).
13. Liwicki M, Bunke H. Hmm-based on-line recognition of handwritten whiteboard notes. In Proc. 10th Int. Workshop Frontiers in Handwriting Recognition; 2006. pp. 595–599..
14. [Liwicki M, Graves A, Bunke H, Schmidhuber J. A novel approach to on-line](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0120) [handwriting recognition based on bidirectional long short-term memory](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0120) [networks. Proc. 9th Int. Conf. on Document Analysis and Recognition](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0120) [2007;1:367–71](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0120).
15. [Mahmoud S, Luqman H, Al-helali B, Binmakhashen G, Parvez M. Online-khatt:](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0125) [An open-vocabulary database for arabic online-text processing. Open Cybern](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0125) [Syst J 2018;12:42–59](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0125).
16. Matic N, Guyon I, Denker J, Vapnik V. Writer-adaptation for on-line handwritten character recognition. In Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR’93), IEEE; 1993.

pp. 187–191..

1. [Meng Z, Li J, Gong Y. Adversarial speaker adaptation. In: ICASSP 2019–2019](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0135) [IEEE International Conference on Acoustics, Speech and Signal Processing](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0135) [(ICASSP). IEEE; 2019. p. 5721–5](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0135).
2. [Platt JC, Matic NP. A constructive rbf network for writer adaptation. Advances](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0140) [in Neural Information Processing Systems 1997:765–71](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0140).
3. [Singh H, Sharma RK, Singh VP. Online handwriting recognition systems for](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0145) [indic and non-indic scripts: a review. Artif Intell Rev 2020:1–55](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0145).
4. [Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0150) [simple way to prevent neural networks from overfitting. J Mach Learn Res](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0150) [2014;15:1929–58](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0150).
5. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. In Advances in Neural Information Processing Systems; 2017..
6. [Wang ZR, Du J. Writer code based adaptation of deep neural network for](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0160) [offline handwritten chinese text recognition. In: 2016 15th International](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0160) [Conference on Frontiers in Handwriting Recognition (ICFHR), IEEE. p. 548–53](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0160).
7. [Wang ZR, Du J, Wang JM. Writer-aware cnn for parsimonious hmm-based](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0165) [offline handwritten chinese text recognition. Pattern Recogn](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0165) [2020;100:107102](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0165).
8. Yang HM, Zhang XY, Yin F, Luo Z, Liu CL. Unsupervised adaptation of neural networks for chinese handwriting recognition. In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR). p. 512–7. doi: <https://doi.org/10.1109/ICFHR.2016.0100>.
9. Yang HM, Zhang XY, Yin F, Sun J, Liu CL. Deep transfer mapping for unsupervised writer adaptation. In: 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR). p. 151–6. doi: [https://doi.org/](https://doi.org/10.1109/ICFHR-2018.2018.00035) [10.1109/ICFHR-2018.2018.00035](https://doi.org/10.1109/ICFHR-2018.2018.00035).
10. [Zhang XY, Bengio Y, Liu CL. Online and offline handwritten chinese character](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0180) [recognition: A comprehensive study and new benchmark. Pattern Recogn](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0180) [2017;61:348–60](http://refhub.elsevier.com/S1110-8665(22)00019-6/h0180).
11. Zhang XY, Liu CL. Writer adaptation with style transfer mapping. IEEE Trans Pattern Anal Mach Intell 2013;35:1773–87. doi: [https://doi.org/10.1109/](https://doi.org/10.1109/TPAMI.2012.239) [TPAMI.2012.239](https://doi.org/10.1109/TPAMI.2012.239).