Deep Adaptive Learning for Writer Identification based on Single Handwritten Word Images

Sheng He*, Lambert Schomaker

Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence, University of Groningen, PO Box 407, 9700 AK, Groningen, the Netherlands

Abstract

There are two types of information in each handwritten word image: explicit information which can be easily read or derived directly, such as lexical content or word length, and implicit attributes such as the author's identity. Whether features learned by a neural network for one task can be used for another task remains an open question. In this paper, we present a deep adaptive learning method for writer identification based on single-word images using multi-task learning. An auxiliary task is added to the training process to enforce the emergence of reusable features. Our proposed method transfers the benefits of the learned features of a convolutional neural network from an auxiliary task such as explicit content recognition to the main task of writer identification in a single procedure. Specifically, we propose a new adaptive convolutional layer to exploit the learned deep features. A multi-task neural network with one or several adaptive convolutional layers is trained end-to-end, to exploit robust generic features for a specific main task, i.e., writer identification. Three auxiliary tasks, corresponding to three explicit attributes of handwritten word images (lexical content, word length and character attributes), are evaluated. Experimental results on two benchmark datasets show that the proposed deep adaptive learning method can improve the performance of writer identification based on single-word images, compared to non-adaptive and simple linear-adaptive approaches.

Keywords: Writer identification, Deep adaptive learning, Handwritten word attributes, Multi-task learning

1. Introduction

Writer identification is a typical pattern-recognition problem which aims to recognize the author of a handwritten passage from an image of it. The authorship is an implicit (indirect) attribute of a handwritten document. A

*Corresponding author

Email addresses: heshengxgd@gmail.com (Sheng He), L.Schomaker@ai.rug.nl (Lambert Schomaker)

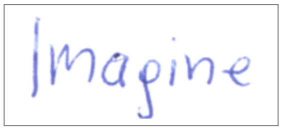
writer-identification system usually extracts the handwriting-style information from the query document image and compares it with the style information of known writers. The handwriting style is usually measured by a number of geometric features, such as global statistics of ink traces [1, 2] or the distribution of graphemes [3, 4]. The reliability of a typical writer-identification system using handcrafted features depends on the amount of text in handwritten images. In [5] it was found that when using traditional writer identification approaches, about 100 letters are needed per sample of Western handwriting to achieve the very satisfactory results.

However, in the digital era, handwriting is an increasingly rare activity. In forensic applications, this requires a new approach to be able to recognize the writer based on the very small amount of available text, which may be as little as a single word. In this paper, we study the writer-identification problem based on single-word images, which is a challenging problem because the information contained in a single word is a highly limited information source for modelling an author's writing style. In order to solve this problem, the convolutional neural network (CNN) [6] is used for writer identification in this paper because it can learn discriminative and hierarchical features at different abstraction levels from raw data and it has achieved good performance on various applications in computer vision [6, 7, 8] and handwriting recognition [9, 10].

There are two types of information in any given image of a handwritten word: explicit information, such as the lexical content, word length and character attributes, and implicit information, such as the writer's identity. Explicit information can be derived relatively easily from the image sample itself, whereas implicit information must be derived from a separate source. An example is shown in Fig. 1. The derivation or estimation of implicit and explicit information actually corresponds to different tasks, such as word recognition and writer identification, which would be treated separately in traditional pattern recognition methods. Word recognition methods extract shape features which come from a sequence of curvilinear strokes in word images [11], while writer-identification methods extract the slant, curvature or ink-width distribution to capture the writing style applied to form the handwritten word [1, 2]. This distinction appears to involve a loss of resources and a lack of generalizability, which becomes clearer as more tasks are attempted - such as document dating or historical writing-style classification - for which completely new approaches need to be designed. At the same time, specific aspects of shape information can be expected to be useful for more than one task.

Performing more than one task on the same input data corresponds to the multi-task learning problem [12, 13, 14] and this has been achieved successfully in many applications. In this paper, we apply multi-task learning to the same input to train neural networks on writer identification with an additional auxiliary task, i.e., word-text recognition, which addresses the explicit information present in a handwritten word image.

It has been shown in [15] that the layers of learned convolutional neural networks transition from being more general, towards the input layer, to being more task-specific, towards the output layer. The layers close to the input



Implicit:
Writer identity: "bob"

Explicit:
Word content: 'Imagine'
Word length: 7 letters
Character attribute: a, e, g, i, m, n

Figure 1: There are two types of information in any given image of a handwritten word: implicit information, such as the writer and explicit information such as the exact word content, word length and character presence.

will contain more general representations which can be shared between different tasks in multi-task learning. However, layers close to the output become more specific to each task and they cannot be used directly for other goals than the one trained for. In the literature, transfer learning is usually adopted to transfer general features between multiple tasks by sharing several lower layers closer to the input. Adaptive learning can be applied to transfer the specific features of the auxiliary task to the main task by a linear combination of input activation maps, in order to achieve better performance in the main task [16, 17].

Because the information capacity of the convolutional neural network is quite large, as expressed in the number of weights, it is to be expected that it can learn different features for different tasks. For example, the features learned for word recognition might capture word-shape information, while features learned for writer identification might capture the ink density or curvature information in the handwritten images. Deep adaptive learning aims to transfer and mix the learned features from one task to and with another in order to improve performance by using an integral end-to-end training procedure. This is expected to work due to the following two reasons: (1) A deep neural network that is trained just for the writer identification task might be overfitted for the writer identification problem and therefore it is possible that it does not generalize well within this task. Conversely, adapting the trained features to an additional task during the training itself is assumed to introduce a regularization which can reduce the risk of over-fitting [18] and improve the performance on unseen data. (2) Transferring the learned features from other tasks can be considered to be feature combination over different pathways in a particular layer. Feature combination has been shown to provide better performance [1, 19].

In this paper, we will apply deep adaptive learning to the application of writer identification under the difficult condition of a very small sample, for instance an isolated word. This is a highly challenging problem because the writer-related style information will be very limited in the small word image. We will choose different attributes of handwritten word images as the auxiliary task to demonstrate the effectiveness of the proposed deep adaptive learning method. In particular, we will choose three tasks as the auxiliary task in multi-task learning: word recognition, word length estimation and character attribute recognition. When showing a word image to a human reader, the word content will be recognized first, but we can ask additional questions about word length or about the shape attributes of the characters it contains. In fact, there may be several other explicit pieces of information when we read a handwritten word image, such as the stroke width of the ink caused by the writing instruments, or the number of circle and cross line intersections present in the word image, etc. To test the hypothesis that the proposed deep adaptive learning method works, we selected explicit information which is very easy to derive (word label, word length, number of letters), and does not require additional complicated pattern-recognition tools such as a circle detector. In general, the auxiliary tasks should not introduce expensive additional labelling in a real-world application.

The contributions of this paper are summarized as follows: (1) We study the writer-identification problem based on single word images, which is a very challenging real-life application problem. (2) We propose a non-linear deep adaptive learning method to transfer the features learned from an auxiliary task to the writer-identification task, fully integrated within the training procedure. We will demonstrate that the proposed deep adaptive learning method will provide better performance than non-adaptive or linear-adaptive learning methods. (3) We evaluate three different auxiliary tasks for writer identification (word recognition, word-length estimation and character-attribute recognition), which all improve the performance to different degrees.

Signature identification or verification aims to verify the individual's identity from handwritten signatures [20]. The problem of writer identification based on single-word images is somewhat similar to the signature identification problem, since both extract an individual's writing style. However, writer identification based on single-word images aims to identify the writer based on any given word, as opposed to the signature, which is stable to the individual and usually designed by that person to have a unique personal shape, unlike isolated handwritten words from a normal piece of text. Our proposed method attempts to model the general writing style from a set of isolated handwritten word images in the training set.

This paper is organized as follows. In Section 2 we provide an overview of related work. We introduce the proposed adaptive learning in Section 3. The experimental results are presented and discussed in Section 4. The last section concludes the paper.

2. Related Work

Most writer identification methods are text-independent, extracting features from large image regions - such as pages, text blocks or sentences - instead of small word images. In the last few decades, many specially handcrafted features have been designed to extract low-level features from handwritten images. These can be roughly grouped into two groups: textural-based and grapheme-based features.

Textural-based methods extract statistical information from the entire text blocks as features. Considering the handwritten text as a texture, textural features are extracted to measure the similarity in handwriting style between different handwritten document images. Local binary patterns (LBP) [21, 22] and local phase quantization (LPQ) are proposed in [21] and the run-length of

Table 1: Advantages and disadvantages of different writer-identification methods.

References	Features	Advantages	Disadvantages
Texture-based features			
[21, 22, 23]	Each pixel is described by local binary patterns (LBP and LPQ) and the feature from the whole text-block is computed by a normalized histogram.	Easy to compute without binarization and segmentation. Parameter-free methods.	The LBP histogram itself is not effective and some post-processing steps are usually applied, such as GLCM, PCA or Run-length.
[24, 25, 26]	Computes the response of hand-crafted Gabor-based filters to describe the texture properties of handwriting style.	Each type of filter captures certain handwritten character shapes, thus the feature is easy to understand and explain to end users.	Requires careful design or selection of the parameter values of filters.
[1, 2, 27, 28]	Extracts the writing style information based on ink trace by edge or contour angles. The feature vector is the joint distribution of angles on each position of ink trace.	Fast and efficient to compute. Captures curvature and slant information of the writing style.	Requires binarization or high-contrast images.
Grapheme-based features			
[29, 30, 31]	Computes contour and stroke fraglets for handwritten characters.	Informative and each grapheme represents an entire letter or parts of letters which are shared between different characters.	Requires binarization, segmentation and an effective fragmentation heuristic for connected-cursive handwritten documents.
[3, 32]	Extracts small patches on handwritten characters.	The patches are small so that they can be used for many different scripts and can be generated randomly.	No pattern in the small patches carries any semantic information. The patches are too small and the distribution is not distinctive enough for graphemic style differences, thus performance is limited.
[33]	Uses the elliptic model to generate an exhaustive number of graphemes.	Model-driven method without codebook training (grapheme selection involved to obtain a compact feature vector).	Morphological operations are needed to match the handwriting contours and graphemes. Due to elliptic model limitation, it is only evaluated for Arabic texts.
[4]	Extracts junction parts on the ink traces.	Junctions are prevalent in different handwritten scripts. Their shape contains the writing style of the author and can be used for cross-script writer identification.	Requires binarized images and the performance is limited in poor-quality images.

LBP is proposed in [34] for writer identification. A run-length histogram with four principal directions is proposed in [23] for writer identification in a multi-script environment. Filter-based features, such as Gabor [24], XGabor [25] and oriented Basic Image Feature Columns (oBIF Columns) [26], have also been studied. Some features can be extracted from the contours of the ink trace, such as Hinge-based features [1, 2, 27, 28], which extract the slant property of characters alongside other information, such as stroke width [2] and curvature information [3]. Other features, such as symbolic representation [35] and k-adjacent segments (kAS) [36, 37] are also used for writer identification. Gaussian Mixture Models (GMMs) are used to model a person's handwriting in [38] and Hidden Markov Model (HMM)-based recognizers are used in [39].

Grapheme-based features extract allographic patterns and map them into a common space (also known as a codebook). Connected-component contours (CO^3) are proposed in [29, 31] for writer identification using upper-case Western scripts, and have more recently been extended to Fraglets [1, 30] for cursive handwriting documents. Small patches extracted from characters are used as graphemes in [3, 32] and synthetic graphemes which have been generated based on the beta-elliptic model are used in [33] on Arabic handwritten document images. The junctions in handwritten images are very useful for measuring the handwriting style and they are considered to be basic elements of the handwritten text for writer identification in [4]. SIFT feature [40] and RootSIFT descriptor are also used for writer identification [41, 42]. Both the textural-based and grapheme-based features can be used to generate more powerful features by the co-occurrence or joint feature principle, which can be found in [19]. Table 1 shows the advantages and disadvantages of each method mentioned above.

Writer identification based on scarce data has also been investigated. For example, Alaei and Roy [35] propose a writer identification method based on the line and page-level, where performance at the page-level is higher than the performance at the line-level. Similar conclusion were obtained in [26], where comparable performance was achieved based on at least three lines using the oBIF features with delta encoding. Adak and Chaudhuri [43] propose a writer identification method for isolated Bangla characters and numerals. The handcrafted features usually need more text because statistical information is used to model the writing style, and the corresponding feature distribution must be stable and representative when more texts are given. However, there are usually only a few letters/characters in single-word images. Therefore, the handcrafted feature distribution extracted on their basis does not approximate the true distribution of the writing style, resulting in poor performance. If the amount of text is limited, the importance of small structural fragments of shape evidence becomes greater. We expect convolutional deep learning to be able to learn the necessary feature-kernel shapes.

Recently, deep learning has also been used for writer identification. For example, a neural network can be trained based on a small block, segmented from the text line with a sliding window [44] or a texture block [45]. A deep multi-stream CNN is proposed in [46] to learn deep features for writer identification. As mentioned above, a deep neural network can learn discriminative and hierarchical features [47] and can recognize writers on the basis of less data. Therefore, deep learning can capture a writing style based on single-word images. However, all of these methods consider writer identification as a single task. Multi-task learning aims to jointly learn classifiers for several related tasks using shared representation. For example, the method proposed in [48] uses an external task to improve semantic segmentation in natural images. Other multiple-task learning methods using CNN include edge labels and surface normals [49] and face detection and face landmark detection [12]. Hwang and Kim [14] propose multi-task learning for the classification and localization of medical images. Misra et al. [16] propose a *cross-stitch unit* in order to learn an optimal combination of shared and task-specific representations among multiple tasks. Multi-task learning is also evaluated in natural language processing, which demonstrates that adding an auxiliary task can help improve the performance of the main task [50]. Our proposed method uses a non-linear adaptive strategy which introduces a convolutional layer to transfer features from the auxiliary task to the main task.

3. Proposed Method

In this section, we describe the proposed method for writer identification based on single-word images using deep adaptive learning. We first introduce the structure of the CNN used for the multi-task learning, with the writer identification as its main task. After that, we show how to transfer and adapt the learned features from the auxiliary task to the main task to improve the performance of writer identification.

3.1. Main Architecture of the Convolutional Neural Network

The architecture of our convolutional neural network is a multi-task adaptation of the AlexNet structure [6], which is shown in Fig. 2. The architecture contains a pathway for the main task and a pathway for the auxiliary task. The two pathways interact at several possible layers where adaptation takes place. For the main task, the pathway consists of eight convolutional layers, with four max-pooling layers after every two convolutional layers in order to increase the depth of network and three fully connected layers. All of the inputted handwritten word grayscale images are resized to $120 \times 40 \times 1$. The size of the receptive field is 3×3 for all of the convolutional layers, which is widely used in deep neural networks [51, 52]. The convolutional stride is fixed at one pixel for all of the convolutional layers. The number of filters of each convolutional layer is depicted in Fig. 2. The first two convolutional layers are shared by both task pathways. For the auxiliary task, each layer mirrors a corresponding layer in the pathway for the main task. Details concerning this configuration are presented below.

After each convolutional or fully-connected layer (except for the last softmax layer), the leaky-ReLU (Rectified linear unit) activation function [53] is used to avoid neurons dying if their input activations are below the threshold, which

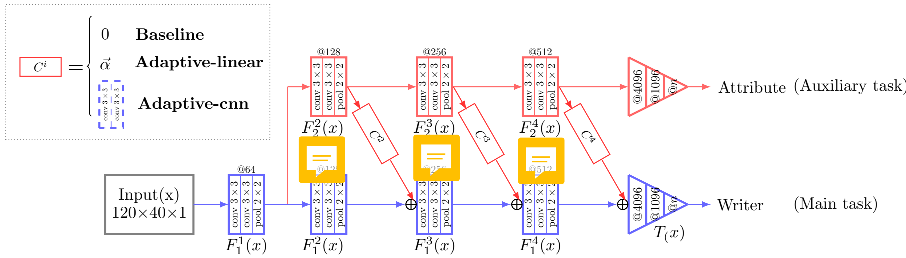


Figure 2: Overall diagram of the proposed deep adaptive convolutional neural network. The input is a grayscale word image of 120 × 40 pixels. There are eight convolutional layers for each task, four max-pooling layers and three fully-connected layers in this framework. Each convolutional block contains two convolutional layers and one max-pooling layer (the kernel size is denoted in the boxes). $F_1^i(x)$ denotes the feature maps on the i-th block for the main writer identification task (blue) and $F_2^i(x)$ for the auxiliary task (red). The notation @'k' above each block indicates the number of kernels used in the convolution. The number n in the last layer represents the number of classes. The block $C^i(\cdot)$ is an adaptive function, which has three types in this paper: **Baseline** when $C^i(\cdot) = 0$, **Linear-adaptive** when $C^i(\cdot) = \vec{\alpha}$ and **Deep-adaptive** when $C^i(\cdot) = cnn$, i.e., a deep network itself.

is defined as: $f(x) = \max(\lambda x, x)$ (in this paper, $\lambda = 0.1$). Spatial pooling is also very important in CNN models to integrate the available information and simultaneously to reduce the size of the feature maps. In our model, a max-pooling layer with a kernel size of 2×2 and a stride step of 2 is implemented after every two convolutional layers (see Fig. 2) to reduce the size of the input representation. Dropout layers [54] a dropout rate of 50% are applied after the first two fully connected layers in order to mitigate the over-fitting problem. The last layer is usually a softmax layer for single label recognition. For the loss function, we applied the cross-entropy loss, which measures the dissimilarity between the true label distribution and the predicted label distribution.

3.2. Auxiliary pathway and adaptive transfer

As shown in Fig. 2, the auxiliary pathway receives shared-feature patterns and the layers are organized in parallel to the main pathway. It would be beneficial to adapt the learned high-level task-specific features from the layers near the output layer of the neural network of the auxiliary task to the main task in order to improve the performance, if the learned features from layers near the output layer are reusable in another task [15]. However, it is unlikely that the learned features can be used as they are, and some task-specific fine-tuning is likely to be required. Therefore, we propose an adaptive network which transfers the representation from layers near the output layer of an auxiliary task to the main task via an adaptive function, $C^i(\cdot)$. Given two activation maps $r(F_1^i)$ and $r(F_2^i)$ from the convolutional layer F^i ($i = 2, 3, 4$ in Fig. 2) for both tasks (F_1 for writer identification and F_2 for the auxiliary task), we learn a combination of $r(F_1^i)$ and $r(F_2^i)$ and feed it as input to the next layer F_1^{i+1} of the main task by:

$$in(F_1^{i+1}) = r(F_1^i) + C^i\Big(r(F_2^i)\Big) \tag{1}$$

where $in(F_1^{i+1})$ is the input of the next layer F_1^{i+1} and $C^i(\cdot)$ is an adaptive function on the layer F_2^i which adapts the representation $r(F_2^i)$ from the auxiliary task to the main task of writer identification.

Different adaptive functions $C^i(\cdot)$ can be applied, and in this paper, we evaluate three types of functions as follows:

1. **Baseline** ($C^i(\cdot) = 0$): The adaptive function is zero, which means that there is no adaptation between two tasks. This can be considered as the baseline, in which two tasks share the first two convolutional layers without adaptation.

2. **Linear-adaptive** ($C^i(\cdot) = \vec{\alpha}$): The adaptive function is a linear mix function, which is similar to the cross-stitch unit proposed in [16]. In this case, Eq. 1 becomes:

$$in_j(F_1^{i+1}) = \alpha_j \cdot r_j(F_1^i) + (1 - \alpha_j) \cdot r_j(F_2^i) \tag{2}$$

where j is the index of the number of activation maps in the layer F^i, α_j is the parameter which weights the activation from the main task and $1 - \alpha_j$ weights the activation from the auxiliary task. Note that we set different α to different activation maps and the dimensionality of the $\vec{\alpha}$ vector is the same as the depth of the layer $r(F_2^i)$, i.e., the number of filters in the layer $r(F_2^i)$. Given the initialization (α=0.5), the $\vec{\alpha}$ is also learned during training and the network can find the optimal weights of the adaptive function between the activation maps of the auxiliary and the main tasks.

3. **Deep-adaptive** ($C^i(\cdot)$ =CNN): In this case, the adaptive function is a convolutional neural network itself. In this paper, we use two convolutional layers with the kernel 3×3 and the number of kernels of each $C^i(\cdot)$ is the same as the corresponding layers F_1^i and F_2^i in order to make the dimension equal for the add operation. From Eq. 1 we can obtain:

$$C^i\Big(r(F_2^i)\Big) = in(F_1^{i+1}) - r(F_1^i) \tag{3}$$

where $r(F_1^i)$ is the features on the i-th layer and $in(F_1^{i+1})$ is the input features of the $(i + 1)$-th layer of the main task. Therefore, $C^i\Big(r(F_2^i)\Big)$ is the residual features of the main task learned from layer F_2^i of the auxiliary task. Using the convolutional layers as the adaptive function makes it possible to capture more complex structures between the activation maps of the different tasks and find the best adaptive representations between two different tasks. These adaptive layers are also learned jointly during the training, and the loss of the main task is back-propagated through these adaptive layers.

3.3. Training

There are two losses in our network: $\mathbf{Loss}_{au}$ for the auxiliary task and $\mathbf{Loss}_{wi}$ for the writer-identification task. The cross-entropy loss function is computed in this paper for both the auxiliary and the main tasks. The network is trained jointly for the auxiliary and writer-identification task, based on a weighting strategy in our paper. The objective function is defined as:

$$\mathbf{Loss}_{total} = (1-\lambda)\mathbf{Loss}_{au} + \lambda \mathbf{Loss}_{wi} \tag{4}$$

where λ is the trade-off weight of the two losses. At the beginning of training, these two losses are equal, so we set $\lambda = 0.5$. In practice, we have found that the loss of the auxiliary task, which recognizes the explicit information, decreases faster than the loss of the writer-identification task. Therefore, we increase the λ after a given iteration to fine-tune the network for writer identification. As explained in [14], the relative importance of the two losses weighted by λ can be back-propagated to the adaptive layers $C^i(\cdot)$.

4. Experiments

In this section, we conduct experiments on two benchmark datasets for writer identification based on single-word images with three different auxiliary tasks.

4.1. Datasets

We evaluate our proposed methods through the use of two publicly available **CVL** and **IAM** datasets which present segmented word images with labels for both word and writer. The proposed method is evaluated through using these two datasets separately, because the writers from these two datasets differ.

CVL [55] consists of 310 writers, each of which contributing at least five pages in English and German. The word regions were automatically labelled and were evaluated by two students independently. In order to train the network for this paper, we select word images with at least twenty instances. Ultimately, this yielded 99,513 selected word images which were randomly split into training (70,778 word images) and testing (28,735 word images) sets.

[redacted]

4.2. Implementation details

The neural network was first initialized using the Xavier method proposed in [57], which has proven to work very well in practice and can speed up training. The adaptive learning rate algorithm Adam proposed in [58] was used to train the neural network, with an initial learning rate of 0.0001. The size of the mini-batch was set to 100 and the number of training iterations was set to 40,000.

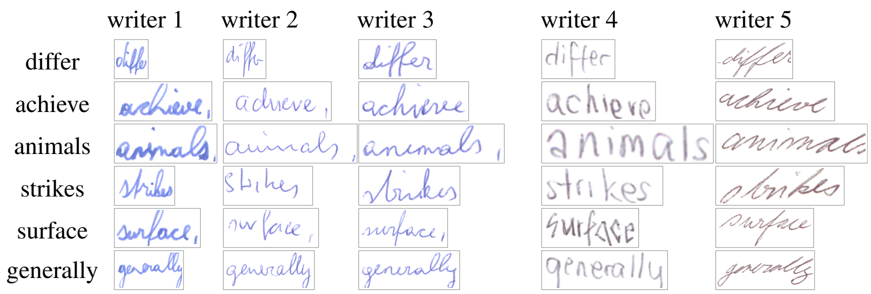


Figure 3: Examples of handwritten word images from the **CVL** dataset with different words and writers. Each image has two attributes: lexical content and the writer's identity.

During training, the parameter of λ in Eq. 4 was set to 0.5 for the first 10,000 iterations. It was then increased by 0.066 at every 5,000 iterations, up to 0.9 at the end of training. Our network was trained using the Tensorflow platform [59]. Training took about 7.5 hours for the **Baseline** and **Linear-adaptive** CNN models and 8.5 hours for the **Deep-adaptive** model, on a single GPU (NVIDIA GTX 960 with 4G memory).

4.3. Performance of writer identification with word recognition as auxiliary task

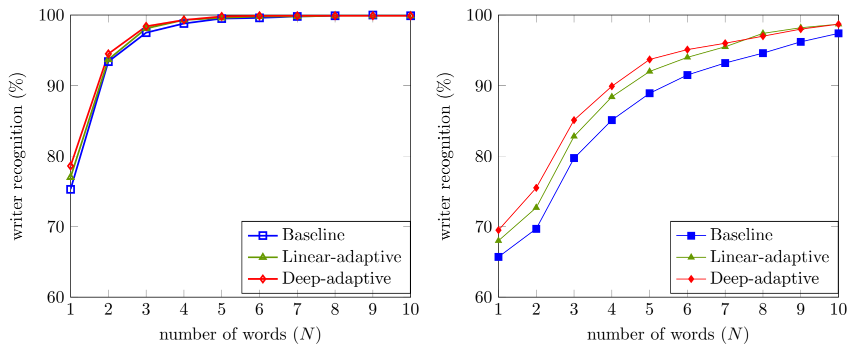
The lexical content of the word image is a very important information, which corresponds to the word recognition or spotting problem [60, 61]. This section reports the experimental results with word recognition as the auxiliary task to improve the performance of writer identification based on single-word images. Three hundred different words were selected from the **CVL** dataset and 446 different words from the **IAM** data set. Fig. 3 presents an example of the word images with two attributes: writer and lexical content.

Table 2 shows the performance of writer identification with word recognition as the auxiliary task. From the table we can see that the word-recognition accuracies are higher than those of writer identification, which demonstrates that writer identification (implicit information) based on single-word images is more challenging than word recognition (explicit information). In addition, adaptive learning methods provide better results than the baseline for writer identification and the **Deep-adaptive** model achieves the best performance on the two datasets, outperforming the **Baseline** and **Linear-adaptive** models by 3.3% and 1.6% on **CVL** and 3.8% and 1.5% on **IAM** in terms of the Top-1 recognition rate.

Since the writer-identification performance based on single-word images is lower than that of word recognition, another interesting question is raised: how many words are needed to achieve a higher performance for writer identification, similar to the performance for word recognition? To answer this question, we did another set of experiments about writer identification based on N word images from the same writer. We randomly selected N word images for each writer and put them into the trained CNN model. The average response of the

Table 2: Performance of writer identification using different adaptive learning methods with **word recognition** as the auxiliary task on the **CVL** and **IAM** datasets.

Model	Writer Identification				Word Recognition (*aux.*)			
	CVL		**IAM**		**CVL**		**IAM**	
	Top1	Top5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Baseline	75.3	92.4	65.7	83.5	95.1	99.1	93.5	98.7
Linear-adaptive	77.0	93.1	68.0	84.7	94.1	98.9	91.3	98.1
Deep-adaptive	**78.6**	**93.7**	**69.5**	**86.1**	94.5	99.0	92.6	98.4



(a) **CVL** data set (b) **IAM** data set

Figure 4: Performance (Top1) of writer identification using different numbers of words (from 1 to 10 words), using CNN models trained with word recognition as the auxiliary task on the **CVL** dataset (Figure (a)) and the **IAM** dataset (Figure (b)).

last layer of the CNN model from all N word images was used to recognize the writer by:

$$y = \frac{1}{N}\sum_{i}^{N} \mathrm{CNN}(x_i) \tag{5}$$

where x_i is the i-th input image and $\mathrm{CNN}(x_i)$ is the response of the last layer of the CNN model. The procedure was repeated 20 times for each writer and the average results for different values of N are reported in Fig. 4.

From Fig. 4 we can see that writer-identification performance increases with more word images from the same writer. The **Deep-adaptive** model achieves the best results with different numbers of words for writer identification. The Top-1 performance for writer identification using the **Deep-adaptive** model was 79.1% and 68.3% when using one word, and this increases to 99.8% and 92.0% when using five words on **CVL** and **IAM**, respectively. For the specialized textural features such as the Hinge [1], the minimum text for writer identification is 100 characters [5]. However, the write-identification performance using CNN models with five words are comparable to the results obtained for textural features.

Figure 5: Examples of handwritten word images from the **CVL** dataset with different word lengths and writers. Each image has two attributes: word length and the writer's identity.

Table 3: Performance of writer identification using different adaptive learning methods, with **word length estimation** as the auxiliary task on the **CVL** and **IAM** datasets.

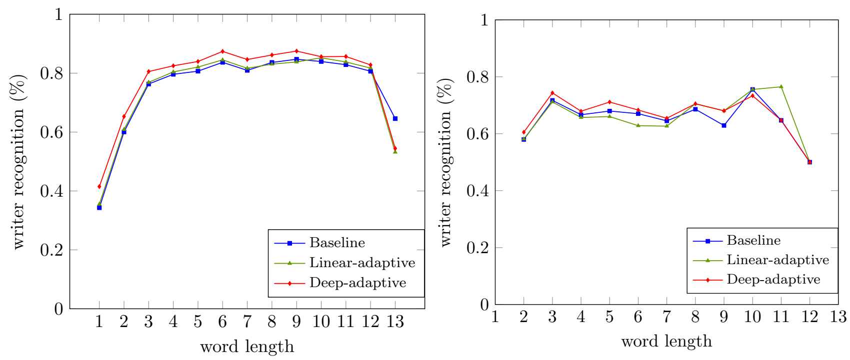
Model	Writer Identification				Word Length Estimation (*aux.*)			
	CVL		**IAM**		**CVL**		**IAM**	
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Baseline	75.3	92.5	66.0	82.9	94.3	99.9	91.5	99.8
Linear-adaptive	75.9	92.7	65.4	83.1	92.7	99.9	90.4	99.8
Deep-adaptive	**79.1**	**94.3**	**68.3**	**85.2**	93.6	99.9	91.6	99.9

4.4. Performance of writer identification with word length estimation as auxiliary task

Word length (number of letters in a word) is another visual attribute of handwritten word images. In this section, we report on writer-identification experiments using word length estimation as the auxiliary task. The maximum word length for both **CVL** and **IAM** is 13 characters. Therefore, the number of classes for word length estimation is 13. Fig. 5 shows an example of word images with different word lengths.

Table 3 shows the writer-identification performance based on single-word images with word length estimation as the auxiliary task. From the table we can see that the word length is also an important attribute and transferring the learned features from word length estimation can also improve writer-identification performance. Like the results in Table 2, the **Deep-adaptive** model provides the best performance.

Fig. 6 shows the writer-identification performance for different word lengths. From the figure, we can see that the performance of writer identification is much less sensitive to word length, unless this is greater than 2. This could be because word images with more than two characters contain more texts which can help to extract stable writing style information by deep learning. Another reason might be that resizing the word images with one or two characters introduces more noise than word images with more than two characters. Note that the performance for word images longer than eleven characters decreases because there are few words with more than eleven characters on the CVL and IAM datasets, thus the number of training samples is not sufficient.



(a) **CVL** data set

(b) **IAM** data set

Figure 6: Performance of writer identification (Top-1) for different CNN models with different word lengths on the **CVL** (Figure (a)) and **IAM** (Figure (b)) datasets.

4.5. Performance of writer identification with character attribute recognition as auxiliary task

Characters contained in the word are also important attributes and are used for word spotting in [62, 9]. In this section, we also report on writer-identification experiments using character attribute recognition as the auxiliary task. We use similar attributes to [62] and each word is represented by a binary histogram with 26 bins, corresponding to 26 English letters. Each element of this histogram represents whether the word being studied contains the relevent letter. Note that we consider lower-case and upper-case letters as the same attribute because there are few upper-case letters in handwritten documents. We also do not consider the spatial information about the characters in a word. For example, the word "are" contains characters 'a', 'e' and 'r', and their corresponding histogram bins are set to 1 and the others are zeros, the same as the PHOC histogram [62] at the first level. Character attribute recognition is a multiple-label learning problem. Therefore, we use the sigmoid activation function instead of softmax on the last layer of the auxiliary task.

Table 4 presents the writer-identification performance based on single-word images with character attribute recognition as the auxiliary task. From the table we obtain the same conclusion: the **Deep-adaptive** model improves the performance of writer identification in both datasets.

Fig. 7 shows the writer-identification performance of word images containing different characters. From the figure we can see that all characters contain writing style information about the writer. The performance for word images which contain the characters 'a','d','h','t' is slightly higher than word images which contain other characters. There are two possible reasons for different letters containing different amounts of handwriting style information: (1) the shapes of these characters are written differently by different writers. (2) These characters typically touch others in a cursive handwritten word and the con-

Table 4: Performance of writer identification using deep adaptive learning with **character attribute recognition** as the auxiliary task on the **CVL** and **IAM** datasets.

Model	Writer Identification				Character Attribute Recognition (*aux*)	
	CVL		**IAM**		**CVL**	**IAM**
	Top1	Top5	Top1	Top5	Accuracy	Accuracy
Baseline	75.1	92.6	65.9	83.4	93.4	91.3
Linear-adaptive	75.3	92.4	65.5	83.4	82.8	77.9
Deep-adaptive	**76.5**	**93.2**	**67.6**	**84.3**	85.1	81.6

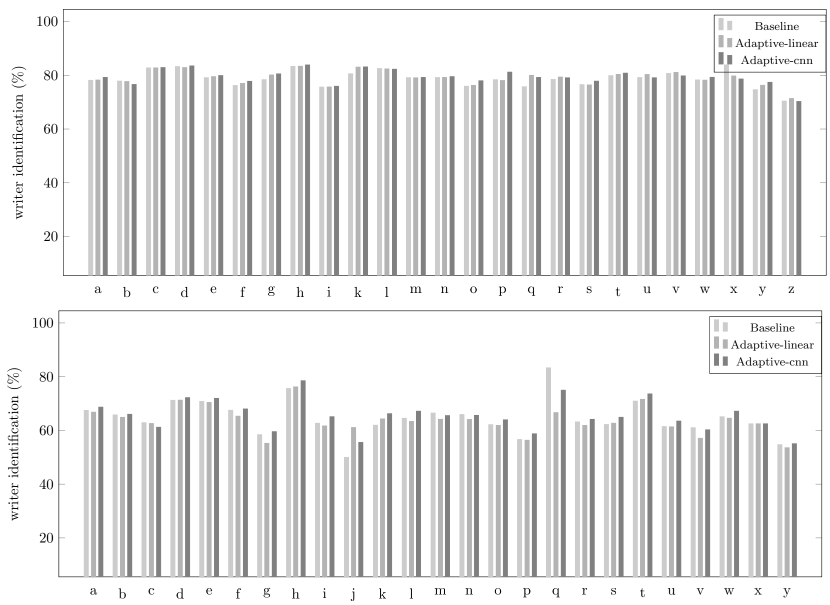


Figure 7: The performance (Top-1) of writer identification with different character attributes. The top figure shows the performance for the **CVL** dataset in which there is no word containing the character 'j' and the bottom figure shows the performance for the **IAM** dataset in which there is no word containing the character 'z'.

necting shapes (ligatures) between the characters are also written differently by different writers.

4.6. Performance with reduced input image sizes

In this section, we evaluate the writer-identification performance to test the effect of reduced input image sizes. A smaller input size of $32\times96\times1$ was chosen to make sure that the minimum size of the last convolutional layer is greater than 1 pixel, since there are four max-pooling layers with stride 2 in our network. Tables 5, 6, and 7 show the writer-identification performance for different adaptive methods with different auxiliary tasks. From these tables we can see that the input size affects the writer-identification performance and that a smaller input size provides poorer results. However, the recognition performance of

Table 5: Performance (Top-1) of writer identification with different input sizes, using different adaptive learning methods with **word recognition** as the auxiliary task on the **CVL** and **IAM** datasets. W.I. means Writer Identification while W.R. means Word Recognition in this table.

Model	Input size: 40×120×1				Input size: 32×96×1			
	CVL		**IAM**		**CVL**		**IAM**	
	W.I.	W.R.	W.I.	W.R.	W.I.	W.R.	W.I.	W.R.
Baseline	75.3	95.1	65.7	93.5	66.7	95.1	61.6	94.2
Linear-adaptive	77.0	94.1	68.0	91.3	69.3	94.0	61.8	91.2
Deep-adaptive	78.6	94.5	69.5	92.6	69.9	94.5	63.5	92.2
Training Time	8.5 hours				5.6 hours			

Table 6: Performance (Top-1) of writer identification with different input sizes, using different adaptive learning methods with **word length estimation** as the auxiliary task on the **CVL** and **IAM** datasets. W.I. means Writer Identification while W.L.E. means Word Length Estimation in this table.

Model	Input size: 40×120×1				Input size: 32×96×1			
	CVL		**IAM**		**CVL**		**IAM**	
	W.I.	W.L.E.	W.I.	W.L.E.	W.I.	W.L.E.	W.I.	W.L.E.
Baseline	75.3	94.3	66.0	91.5	66.4	94.5	60.5	91.4
Linear-adaptive	75.9	92.7	65.4	90.4	68.4	92.8	59.2	89.4
Deep-adaptive	79.1	93.6	68.3	91.6	69.9	93.6	61.8	90.2
Training Time	8.5 hours				5.6 hours			

the explicit information is approximately the same. This is because recognition of the explicit information extracts whole-word characteristics, such as word shape and outline, which are less-sensitive to the word image size. Conversely, the writer-identification model requires detailed features, such as the curvature information of the ink traces, which are missing or deformed in the small images. It should be noted that the proposed **Deep-adaptive** model provides the best writer-identification performance for reduced image sizes, albeit less than when using large images with the same model. Although training on large images takes more computing time, it provides better performance for writer identification (74.1% vs 66.7% average of **CVL** vs **IAM** with word recognition as the auxiliary task). Therefore, we selected a 40×120×1 input size, which is a good trade-off between accuracy and efficiency.

4.7. Comparison with other studies

This section compares other writer identification methods using the **CVL** and **IAM** datasets based on single-word images. For the handcrafted features, we used the "leave-one-out" strategy, the same as the traditional writer identification approach [1, 3]. The representation of each writer is computed as the average word features except the query one. Table 8 shows the performance of the different writer-identification methods. From the table, we can see that the traditional handcrafted features fail to identify the writer based on single-word images, which is also shown in [5]. The CNN model provides much better results

Table 7: Performance (Top-1) of writer identification with different input sizes, using different adaptive learning methods with **word attribute recognition** as the auxiliary task on the **CVL** and **IAM** datasets. W.I. means Writer Identification while W.A.R. means Word Attribute Recognition in this table.

Model	Input size: 40×120×1				Input size: 32×96×1			
	CVL		**IAM**		**CVL**		**IAM**	
	W.I.	W.A.R.	W.I.	W.A.R.	W.I.	W.A.R.	W.I.	W.A.R.
Baseline	75.1	93.4	65.9	91.3	67.6	93.6	60.1	90.6
Linear-adaptive	75.3	82.8	65.5	77.9	69.7	83.9	60.6	76.8
Deep-adaptive	76.5	85.1	67.6	81.6	70.4	86.1	63.5	82.3
Training Time	8.5 hours				5.6 hours			

than the handcrafted features, and our proposed deep adaptive learning method provides the best results.

Table 8: Single-word writer-identification performance using different approaches on the **CVL** and **IAM** datasets.

Method	**CVL**		**IAM**	
	Top1	Top5	Top1	Top5
Hinge [1]	25.8	48.0	26.7	45.4
Quill [2]	29.4	52.6	35.9	57.8
Chain Code Pairs [3]	22.4	44.6	21.6	39.7
Chain Code Triplets [3]	28.8	51.4	30.5	49.8
COLD [34]	12.8	29.6	15.7	32.1
QuadHinge [28]	30.0	52.4	37.2	57.8
CoHinge [28]	25.9	46.9	26.8	47.2
CNN [6]	75.3	92.6	66.0	83.5
CNN+Adaptive	79.1	93.7	69.5	86.1

4.8. Discussion

From Tables 2, 3 and 4, we can see the following. (1) Generally, other conditions being equal, recognizing implicit information (writer identification) is more difficult than recognizing explicit information such as word recognition, word length estimation and character attribute recognition. Since the implicit information is embedded in the patterns of handwritten characters or ink traces, it usually needs more reference data to be recognized correctly. (2) Adaptive learning can improve the performance of the main task. For example, the writer identification performance of the **Linear-adaptive** and **Deep-adaptive** models with three different auxiliary tasks is better than that of the **Baseline** model on both two datasets. (3) The writer identification performance of the **Deep-adaptive** model is better than that of the **Linear-adaptive** model. This is because the deep adaptive learning model can learn the non-linear relationship between different tasks. (4) The performance of the auxiliary task decreases in adaptive learning because the main task information is back-propagated to the

Table 9: Final overview of the writer-identification performance using the **Deep-adaptive** model with different auxiliary tasks.

Auxiliary Tasks	CVL		IAM	
	Top1	Top5	Top1	Top5
Baseline	75.2	92.5	65.8	83.3
Word Recognition	78.6	93.7	**69.5**	**86.1**
Word Length Recognition	**79.1**	**94.3**	68.3	85.2
Character Attribute Recognition	76.5	93.2	67.6	84.3
Combined	78.5	94.0	67.5	84.3

auxiliary task layers. However, the **Deep-adaptive** performance is better than that of the **Linear-adaptive** model, showing that the residual adaptive blocks $C(\cdot)$ can transfer the useful information from the auxiliary task to the main task on the forward phase and mask the useless information back-propagated to the auxiliary task. (5) Using word recognition and word length estimation as the auxiliary tasks yields better results for writer identification in the two datasets than using character attribute recognition (see Table 9). This could be because the character attribute recognition results are not a good choice as an auxiliary task, thus the learned features contain less useful information. Therefore, choosing a high performing auxiliary task can also result in a greater improvement in the main task. (6) We also attempted to combine all three auxiliary tasks together in our experiments, considering the word itself and word length as attributes, similar to the character attributes. The results are shown in Table 9 and we can see that combining all the auxiliary tasks cannot improve performance. This could be because during training, the loss is dominated by the character attributes. For example, the word "Imagine" has 7 character bits and only 1 word bit and 1 word length bit. Thus, the neural network focuses on recognizing the character attributes, which results in a poorer performance than that of the other two auxiliary tasks. (7) The large performance difference between traditional methods and CNN for writer identification based on difficult single-word images (see Table 8) indicates that the necessary information for writer identification is somehow present in individual words. However, as with most CNN methods, there may be some over-fitting which led to the current results. More research is needed to assess the effectiveness of the dropout mechanism used during training, for instance.

The experimental results provide several interesting factors to consider when designing a modern writer identification system: (1) it is better to ask writers to write more words, with at least five words to achieve a high performance. (2) Since the writer identification performance of word images with less than two characters is very low, it is better to ask writers to write words with as least three characters and each word should contain writing-sensitive letters, such as 'a', 'd', 'h', and 't'.

5. Conclusion

This paper has studied the writer identification problem based on single-word images using deep adaptive learning in a multi-task learning framework. Three different tasks which recognize the explicit information of handwritten word images were used as the auxiliary tasks to improve the performance of writer identification. The experimental results on two benchmark datasets have shown several interesting conclusions. Firstly, writer identification is more difficult than other attribute recognition problems because the writer's identity is the implicit information, and even people themselves find recognizing a writer based on single-word images difficult. Secondly, adaptive learning can improve the performance of writer identification since different tasks learn different features and the specific representations of the auxiliary task can be transferred to the main task. Thirdly, deep adaptive learning can capture the complex relationship between the specific features of different tasks and can thus provide better performance.

The performance of writer identification based on single-word images is still much poorer compared to the performance of other tasks using deep learning, and it still needs to be improved in the future. Recently, there has been a big shift from handcrafted features to handcrafted structures in neural networks. Therefore, more complex neural network structures can be investigated in the future for writer identification.

References

References

[1] M. Bulacu, L. Schomaker, Text-independent writer identification and verification using textural and allographic features, IEEE transactions on pattern analysis and machine intelligence 29 (4).

[2] A. Brink, J. Smit, M. Bulacu, L. Schomaker, Writer identification using directional ink-trace width measurements, Pattern Recognition 45 (1) (2012) 162–171.

[3] I. Siddiqi, N. Vincent, Text independent writer recognition using redundant writing patterns with contour-based orientation and curvature features, Pattern Recognition 43 (11) (2010) 3853–3865.

[4] S. He, M. Wiering, L. Schomaker, Junction detection in handwritten documents and its application to writer identification, Pattern Recognition 48 (12) (2015) 4036–4048.

[5] A. Brink, M. Bulacu, L. Schomaker, How much handwritten text is needed for text-independent writer verification and identification, in: International Conference on Pattern Recognition, 2008, pp. 1–4.

[6] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 25, 2012, pp. 1097–1105.

[7] D. Zhang, J. Han, J. Han, L. Shao, Cosaliency detection based on intrasaliency prior transfer and deep intersaliency mining, IEEE transactions on neural networks and learning systems 27 (6) (2016) 1163–1176.

[8] R. Girshick, Fast R-CNN, in: International Conference on Computer Vision, 2015, pp. 1440–1448.

[9] S. Sudholt, G. A. Fink, PHOCNet: A deep convolutional neural network for word spotting in handwritten documents (2016) 277–282.

[10] X.-Y. Zhang, Y. Bengio, C.-L. Liu, Online and offline handwritten Chinese character recognition: A comprehensive study and new benchmark, Pattern Recognition 61 (2017) 348–360.

[11] J. Dasgupta, K. Bhattacharya, B. Chanda, A holistic approach for off-line handwritten cursive word recognition using directional feature based on Arnold transform, Pattern Recognition Letters 79 (2016) 73–79.

[12] Z. Zhang, P. Luo, C. C. Loy, X. Tang, Facial landmark detection by deep multi-task learning, in: European Conference on Computer Vision, Springer, 2014, pp. 94–108.

[13] T. Zhang, B. Ghanem, S. Liu, N. Ahuja, Robust visual tracking via multitask sparse learning, in: Computer vision and pattern recognition, 2012, pp. 2042–2049.

[14] S. Hwang, H.-E. Kim, Self-transfer learning for fully weakly supervised object localization, arXiv preprint arXiv:1602.01625.

[15] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, in: Advances in neural information processing systems, 2014, pp. 3320–3328.

[16] I. Misra, A. Shrivastava, A. Gupta, M. Hebert, Cross-stitch networks for multi-task learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3994–4003.

[17] S. Ruder, J. Bingel, I. Augenstein, A. Søgaard, Sluice networks: Learning what to share between loosely related tasks, arXiv preprint arXiv:1705.08142.

[18] S. Ruder, An overview of multi-task learning in deep neural networks, arXiv preprint arXiv:1706.05098.

[19] S. He, L. Schomaker, Beyond OCR: Multi-faceted understanding of handwritten document characteristics, Pattern Recognition 63 (2017) 321–333.

[20] D. Impedovo, G. Pirlo, Automatic signature verification: The state of the art, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 38 (5) (2008) 609–635.

[21] D. Bertolini, L. S. Oliveira, E. Justino, R. Sabourin, Texture-based descriptors for writer identification and verification, Expert Systems with Applications 40 (6) (2013) 2069–2080.

[22] A. Nicolaou, A. D. Bagdanov, M. Liwicki, D. Karatzas, Sparse radial sampling lbp for writer identification, in: International Conference on Document Analysis and Recognition (ICDAR), 2015, pp. 716–720.

[23] C. Djeddi, I. Siddiqi, L. Souici-Meslati, A. Ennaji, Text-independent writer recognition using multi-script handwritten texts, Pattern Recognition Letters 34 (10) (2013) 1196–1202.

[24] H. E. Said, T. N. Tan, K. D. Baker, Personal identification based on handwriting, Pattern Recognition 33 (1) (2000) 149–160.

[25] B. Helli, M. E. Moghaddam, A text-independent Persian writer identification based on feature relation graph (FRG), Pattern Recognition 43 (6) (2010) 2199–2209.

[26] A. J. Newell, L. D. Griffin, Writer identification using oriented basic image features and the delta encoding, Pattern Recognition 47 (6) (2014) 2255–2265.

[27] S. He, L. Schomaker, Delta-n hinge: rotation-invariant features for writer identification, in: International Conference on Pattern Recognition (ICPR), 2014, pp. 2023–2028.

[28] S. He, L. Schomaker, Co-occurrence features for writer identification, in: International Conference on Frontiers in Handwriting Recognition (ICFHR), 2016, pp. 78–83.

[29] L. Schomaker, M. Bulacu, Automatic writer identification using connected-component contours and edge-based features of uppercase western script, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (6) (2004) 787–798.

[30] L. Schomaker, K. Franke, M. Bulacu, Using codebooks of fragmented connected-component contours in forensic and historic writer identification, Pattern Recognition Letters 28 (6) (2007) 719–727.

[31] G. Ghiasi, R. Safabakhsh, Offline text-independent writer identification using codebook and efficient code extraction methods, Image and Vision Computing 31 (5) (2013) 379–391.

[32] C. Djeddi, I. Siddiqi, L. Souici-Meslati, A. Ennaji, Codebook for writer characterization: A vocabulary of patterns or a mere representation space?, in: International Conference on Document Analysis and Recognition, 2013, pp. 423–427.

[33] M. N. Abdi, M. Khemakhem, A model-based approach to offline text-independent Arabic writer identification and verification, Pattern Recognition 48 (5) (2015) 1890–1903.

[34] S. He, L. Schomaker, Writer identification using curvature-free features, Pattern Recognition 63 (2017) 451–464.

[35] A. Alaei, P. P. Roy, A new method for writer identification based on histogram symbolic representation, in: International Conference on Frontiers in Handwriting Recognition, 2014, pp. 216–221.

[36] R. Jain, D. Doermann, Offline writer identification using k-adjacent segments, in: International Conference on Document Analysis and Recognition, 2011, pp. 769–773.

[37] R. Jain, D. Doermann, Combining local features for offline writer identification, in: International Conference on Frontiers in Handwriting Recognition, 2014, pp. 583–588.

[38] A. Schlapbach, H. Bunke, Off-linewriter identification using Gaussian mixture models, in: International Conference on Pattern Recognition, Vol. 3, 2006, pp. 992–995.

[39] A. Schlapbach, H. Bunke, A writer identification and verification system using hmm based recognizers, Pattern Analysis and Applications 10 (1) (2007) 33–43.

[40] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International journal of computer vision 60 (2) (2004) 91–110.

[41] X. Wu, Y. Tang, W. Bu, Offline text-independent writer identification based on scale invariant feature transform, IEEE Transactions on Information Forensics and Security 9 (3) (2014) 526–536.

[42] V. Christlein, D. Bernecker, F. Hönig, A. Maier, E. Angelopoulou, Writer identification using GMM supervectors and exemplar-SVMs, Pattern Recognition 63 (2017) 258–267.

[43] C. Adak, B. B. Chaudhuri, Writer identification from offline isolated Bangla characters and numerals, in: International Conference on Document Analysis and Recognition, 2015, pp. 486–490.

[44] S. Fiel, R. Sablatnig, Writer identification and retrieval using a convolutional neural network, in: International Conference on Computer Analysis of Images and Patterns, Springer, 2015, pp. 26–37.

[45] Y. Tang, X. Wu, Text-independent writer identification via cnn features and joint Bayesian, in: International Conference on Frontiers in Handwriting Recognition (ICFHR), 2016, pp. 566–571.

[46] L. Xing, Y. Qiao, Deepwriter: A multi-stream deep CNN for text-independent writer identification, in: International Conference on Frontiers in Handwriting Recognition (ICFHR), 2016, pp. 584–589.

[47] M. D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European conference on computer vision, Springer, 2014, pp. 818–833.

[48] A. Vezhnevets, J. M. Buhmann, Towards weakly supervised semantic segmentation by means of multiple instance and multitask learning, in: Conference on Computer Vision and Pattern Recognition, 2010, pp. 3249–3256.

[49] X. Wang, D. Fouhey, A. Gupta, Designing deep networks for surface normal estimation, in: Computer Vision and Pattern Recognition, 2015, pp. 539–547.

[50] J. Bjerva, Will my auxiliary tagging task help? estimating auxiliary tasks effectivity in multi-task learning, in: Proceedings of the 21st Nordic Conference on Computational Linguistics, 2017, pp. 216–220.

[51] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556.

[52] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[53] A. L. Maas, A. Y. Hannun, A. Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in: Proc. ICML, Vol. 30, 2013.

[54] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, The Journal of Machine Learning Research 15 (1) (2014) 1929–1958.

[55] F. Kleber, S. Fiel, M. Diem, R. Sablatnig, CVL-database: An off-line database for writer retrieval, writer identification and word spotting, in: International Conference on Document Analysis and Recognition, 2013, pp. 560–564.

[56] U.-V. Marti, H. Bunke, The IAM-database: an English sentence database for offline handwriting recognition, International Journal on Document Analysis and Recognition 5 (1) (2002) 39–46.

[57] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: International Conference on Artificial Intelligence and Statistics, 2010, pp. 249–256.

[58] D. P. Kingma, J. L. Ba, Adam: A method for stochastic optimization.

[59] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al., Tensorflow: Large-scale machine learning on heterogeneous distributed systems, arXiv preprint arXiv:1603.04467.

[60] T. Van der Zant, L. Schomaker, K. Haak, Handwritten-word spotting using biologically inspired features, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (11) (2008) 1945–1957.

[61] J. Almazán, A. Gordo, A. Fornés, E. Valveny, Segmentation-free word spotting with exemplar SVMs, Pattern Recognition 47 (12) (2014) 3967–3978.

[62] J. Almazán, A. Gordo, A. Fornés, E. Valveny, Word spotting and recognition with embedded attributes, IEEE Transactions on Pattern Analysis and Machine Intelligence 36 (12) (2014) 2552–2566.