Pattern Recognition Letters 121 (2019) 104–112

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Text-independent writer identification using convolutional neural network

Hung Tuan Nguyen[a], Cuong Tuan Nguyen[a], Takeya Ino[a], Bipin Indurkhya[b], Masaki Nakagawa[a,*]

[a] Department of Computer and Information Sciences, Tokyo University of Agriculture and Technology, 2-24-16 Naka-cho, Koganei-shi, Tokyo 184-8588 Japan
[b] Institute of Philosophy, Jagiellonian University, Cracow, Poland

ARTICLE INFO

Article history:
Available online 25 July 2018

ABSTRACT

The text-independent approach to writer identification does not require the writer to write some predetermined text. Previous research on text-independent writer identification has been based on identifying writer-specific features designed by experts. However, in the last decade, deep learning methods have been successfully applied to learn features from data automatically. We propose here an end-to-end deep-learning method for text-independent writer identification that does not require prior identification of features. A Convolutional Neural Network (CNN) is trained initially to extract local features, which represent characteristics of individual handwriting in the whole character images and their sub-regions. Randomly sampled tuples of images from the training set are used to train the CNN and aggregate the extracted local features of images from the tuples to form global features. For every training epoch, the process of randomly sampling tuples is repeated, which is equivalent to a large number of training patterns being prepared for training the CNN for text-independent writer identification. We conducted experiments on the JEITA-HP database of offline handwritten Japanese character patterns. With 200 characters, our method achieved an accuracy of 99.97% to classify 100 writers. Even when using 50 characters for 100 writers or 100 characters for 400 writers, our method achieved accuracy levels of 92.80% or 93.82%, respectively. We conducted further experiments on the Firemaker and IAM databases of offline handwritten English text. Using only one page per writer to train, our method achieved over 91.81% accuracy to classify 900 writers. Overall, we achieved a better performance than the previously published best result based on handcrafted features and clustering algorithms, which demonstrates the effectiveness of our method for handwritten English text also.

1. Introduction

Writer identification has been studied for many years because it has practical applications for forgery detection and forensic science. Since the mid-1960s, computational approaches have been developed for writer identification. A review of the state-of-the-art methods for writer verification and identification from the 1960s to the 1980s can be found in Plamondon and Lorette [23]. In the early days, this research was focused on using offline signatures to identify the writer [5,11] because the signatures were considered as individual signs from the past. Yoshimura et al. proposed a text-dependent writer identification method using handwritten text [33,34]. Due to the difficulty of collecting multiple signature patterns and the availability of large databases of handwritten text from the early 2000s, there have been many studies on writer identification employing handwritten text. Srihari et al. [29] argued that the individuality of handwritten text is appropriate for the writer identification task and machine learning algorithms can be trained for this task with a large number of writers.

There are two main approaches for writer identification: text-dependent and text-independent. The text-dependent approach demands the same text to be written while the text-independent approach does not require any particular text. The writer identification research so far has focused on deciding whether the given handwritten characters are written by the person whose handwritten sample patterns in the same categories are available. The problem with this approach is that we cannot always find the sample patterns in the same categories as the target patterns. The text-independent approach does not require collecting the same category patterns, which is useful in situations like forensic science. However, the text-independent approach is more compli-

* Corresponding author.
E-mail addresses: satuki_sasimi@yahoo.co.jp (T. Ino), bipin.indurkhya@uj.edu.pl (B. Indurkhya), nakagawa@cc.tuat.ac.jp (M. Nakagawa).

https://doi.org/10.1016/j.patrec.2018.07.022
0167-8655/

cated than the text-dependent approach because it must extract writer-specific features regardless of signatures, specified letters, or symbols to be compared. In the research presented here, we focus on solving the text-independent writer identification task.

Recently, Sreeraj and Idicula [28] reviewed both the text-dependent and text-independent writer identification methods from the late 1990s to 2010. These methods employed textual features [24], edge-based features [8] and allograph features [6]. In addition, they reviewed many approaches for writer identification based on extracted features: for instance, cosine similarity [1], *k*-nearest neighbor [17] and clustering methods [6,7]. Among these methods, the approach by Bulacu and Schomaker [6] provides the theoretical foundation for writer identification based on handwritten text, where each handwriting pattern is described by a two-level psychomotor process. In this approach, the handwritten patterns are produced from a sophisticated sequence of thinking and hand movements, which are characteristic of each writer depending on his/her cognitive system, nervous system, muscle system, gender, age, schooling, and so on. Therefore, each writer has his/her allographs whenever he/she writes a letter which suggests that it is possible to identify a person based on his/her writing behavior. Bulacu and Schomaker [6] conducted experiments on the handwriting databases with 900 writers using their handcrafted features and clustering methods. The best accuracy results achieved were 80% when using a single handcrafted feature and 87% when using combined features.

There are two kinds of handwritten text patterns: online patterns (time series of pen trajectories to write text) and offline patterns (handwritten text images). Consequently, there are online and offline methods of writer identification. Moreover, two kinds of features are usually employed for writer identification: local features and global features. For offline writer identification, local features are extracted from sub-regions of an image based on local descriptors such as scale-invariant features transform (SIFT) [9,31], local binary pattern (LBP), local phase quantization (LPQ) [2], and contour features [27]. Global features are extracted from an input image at the document level and paragraph level using ink width [4], structural features [21], and texture features [1,13]. Moreover, there are several studies on combining local and global features [6,10,29].

The approach presented in this paper deals with offline writer identification using the trained features from Convolutional Neural Networks (CNNs). The main contributions of this paper are the sub-region level and character level local feature extractors, three aggregation methods to form global features and a sampling mechanism for training a CNN. Our method not only uses isolated characters as inputs to the CNN but is also able to work well with sub-images extracted from handwritten text pages. Moreover, the effectiveness of our method is demonstrated by experimenting with both the Japanese and the English handwriting databases.

The rest of this paper is organized as follows: Section 2 presents a brief survey of recent research on text-independent writer identification using CNNs. Section 3 presents the details of our method, which incorporates local feature extraction, feature aggregation to form global features and a sampling mechanism. Section 4 presents experimental results demonstrating the efficacy of our proposed method on Japanese and English handwriting databases. In this section (Section 4), our method is compared with the best existing approach that uses handcrafted features and clustering methods on English databases. Finally, Section 5 presents our conclusions and suggestions for future research.

2. Related works

This section briefly reviews related works specific to the text-independent approach. As it requires writer-specific features, several local features for offline writer identification have been proposed to be extracted at the character level. Previous studies focused on writer-specific features from the texture. Bensefia et al. [1] proposed morphological grapheme-based features extracted from the graphical fragments of handwritten patterns. Bulacu et al. [8] proposed edge-based features combined with the connected-component contours and curvatures of handwritten samples. The contour and curvature features were also used by Siddiqi and Vincent [27] to extract writer-specific local information on orientation and shape.

Due to their superior performance on image classification and matching tasks, LBP and SIFT descriptors were employed to extract writer-specific local features. Bertolini et al. [2] employed the LBP-based and LPQ-based local features, which yielded an improved performance on both the writer verification and the identification tasks. Christlein et al. [9] and Wu et al. [31] employed SIFT descriptors to extract scale and orientation features at the SIFT key-points, which resulted in a higher accuracy compared with the edge-based and the contour features.

Bulacu and Schomaker [6] proposed a combination of texture-level and allograph-level features. For texture-level features, they considered probability distribution functions (PDFs) computed from contours, connected components, gray-scale images and binary images. For the allograph features, they employed a common codebook of shapes to represent the writer-specific probability distribution of ink-trace fragments (graphemes) due to the assumption that individual stochastically writes graphemes.

Besides, many global features have been studied to obtain discriminative information of individual writers at the document and the text-line levels. Marti et al. [21] proposed several structural features based on the width, slant and height of the three main writing zones for each text line. By gathering these extracted global features from every text line to identify the writer of a page, their system achieved 90.7% accuracy for identifying 20 writers from 100 pages. Brink et al. [4] proposed quill features based on pixel contours to represent the ink-trace width, which achieved a high enough accuracy to be used by domain experts. In addition, there were several studies on combining local and global features [6,10].

Previous research on writer identification used various classifiers such as distance-based classifier, Support Vector Machines (SVM) [15], Hidden Markov Model (HMM) [25], Fuzzy based classifier [30] and so on. For example, approaches employing distance-based classifiers use different metrics like Euclidean distance [7], Chi-square and Hamming distance [6], Bhattacharyya distance [27], etc.

Neural Network-based techniques have also been applied to writer identification in recent years; for example Fiel and Sablatnig [14], Christlein et al. [10]. These techniques used the merit of CNNs to address the problem of feature extraction automatically, because hand-crafted features are difficult to define as mentioned above. During the last decade, deep learning techniques have been successfully applied to many recognition tasks [12,16,18] because they are effective in learning relevant features automatically. This suggests that these techniques can also be applied to extract effective writer-specific features from handwritten patterns automatically.

For the writer identification task, Fiel and Sablatnig [14] employed CNNs as local feature extractors. In their approach, the last fully connected layer is eliminated because the layer just above it extracted adequate features to identify the writer. Then, the mean vector for the input image is computed using all its local feature vectors, which is used for identifying the writer based on Chi-Square distance. This method requires a preprocessing step for image binarization and normalization, so its performance depends on the database and the preprocessing method.

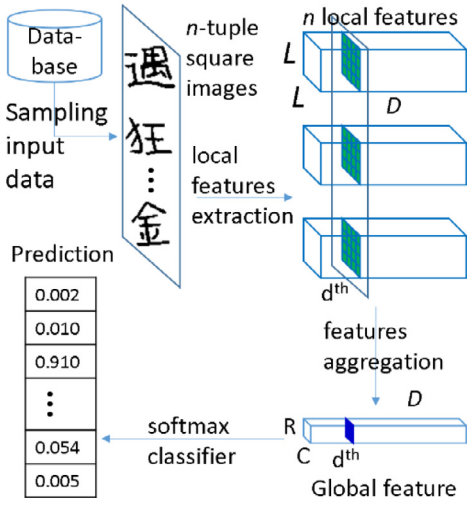


Fig. 1. Overview of our method.

Christlein et al. [10] presented another strategy: after extracting local features from CNNs, they are used to form global features based on Gaussian Mixture Models (GMM) super-vector encoding. This combination method performed better than that of Fiel and Sablatnig [14]. Moreover, CNNs achieved a better performance than the traditional local descriptors such as SIFT.

Yang et al. [32] proposed an end-to-end online text-independent writer identification system, which generates many artificial patterns from a single original online character pattern by dropping one or more strokes. The artificial patterns formed by the same original pattern are fed into CNNs to get a probability distribution of each character. Then, the average distributions of all the character patterns in the document are computed and used for identifying the writer. Even though this system achieved a high accuracy, it requires many character patterns in documents to make the averages reliable. Moreover, its DropSegment method is difficult to apply for offline handwritten text patterns.

As mentioned above, the systems of Christlein et al. [10] and Fiel and Sablatnig [14] have two separate training steps: feature extraction and encoding, where CNNs are pre-trained and employed for local features extraction. That is, the pre-trained CNNs are fixed during the second step while training the encoder, which reduces the performance of the whole system because the CNNs are not updated. Therefore, we propose to use an end-to-end network for writer identification, where the CNN-based feature extractor and the neural network-based classifier are connected and trained together. However, the most notable difference of our approach compared to others is our sampling and feature aggregation methods.

3. Proposed method

3.1. Basic approach

We propose here an end-to-end method based on deep learning, which extracts features using CNNs and combines the extracted features from handwritten text images of multiple characters to retain writer-specific features while reducing character class specific features as shown in Fig. 1.

First, handwritten square images of an individual writer are randomly sampled to form n-tuple images. We employ square shape images since Kanji (Chinese characters) have square shape and the square shape is rather standard for extracting features in a sub-region for Kanji and Western alphabets.

Secondly, every image from n-tuple images is fed to a CNN local feature extractor. Due to the difficulty of handcrafting good features to classify writers, the proposed method uses CNN for automatically learning writer-specific features from handwriting patterns. By using a new way of organizing training samples as n-tuple images, a CNN is able to extract text-independent writer-specific features.

Thirdly, the extracted local features are aggregated by a global feature aggregator in different ways, for example based on the average or the maximum. By forming global features from multiple character images, we expect that the writer specific features such as structure, balance, slant, ligature, serif, and so on could be obtained independently from the text.

Fourthly, the aggregated features are fed to a softmax classifier (fully connected layer consisting of N_{fc} units which equals the number of writers) to make a prediction.

As all the components (local feature extractor, global feature aggregator and classifier) of the proposed network are differentiable, the whole network can be trained by the stochastic gradient descent algorithm, which helps to discover not only the local features but also the global features that are hard to define by handcrafted features.

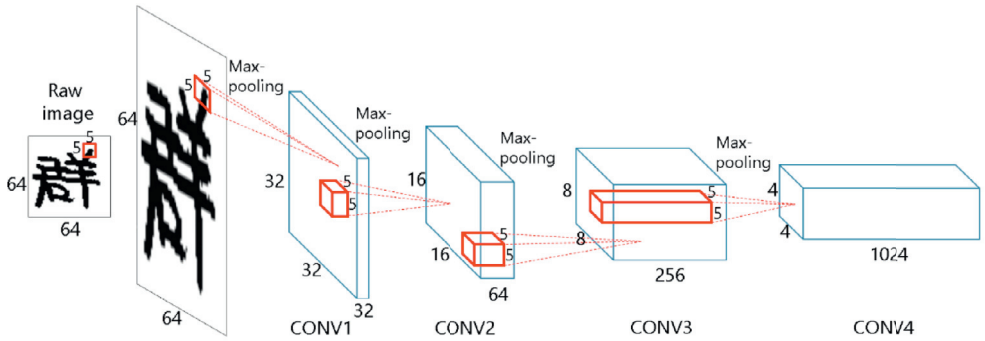
Our approach incorporates both local and global features, which are text independent, i.e., character category invariant. We consider two levels of local features: the sub-region level and the character level. The sub-region level, extracted from sub-regions of a character image, captures writer-specific features in writing strokes, which are directly related to the psychomotor process to identify the writer. The character level, extracted from a character image, captures features related to writing styles such as character balancing, character layouts or stroke combinations.

3.2. Local feature at the sub-region level

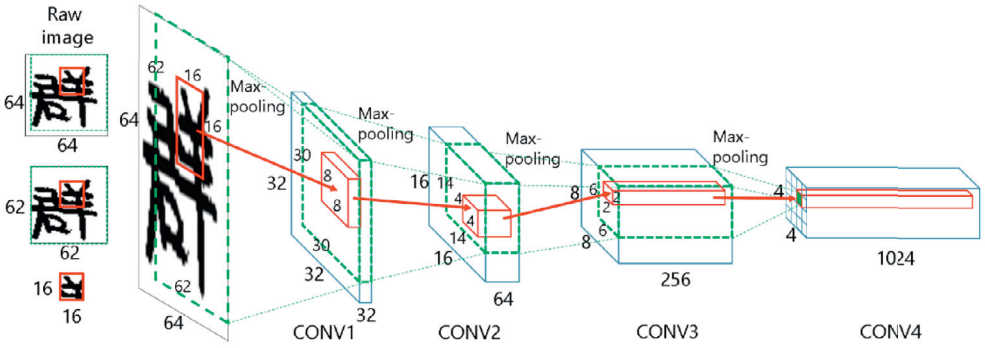
We use the model illustrated in Fig. 2(a) for extracting features from sub-regions in a character image. Each convolution layer uses a kernel size of 5×5, stride step of 2 and padding size of 2. Each max-pooling layer uses a kernel size of 2×2 and stride step of 2. We use 4 stages of processing: in each stage, a convolutional layer is followed by a max-pooling layer to produce the 4×4 feature maps from a 64×64 input image. Each column in these feature maps can be considered as the features from a sub-region of an input character image as shown in Fig. 2(b). Specifically, each column of the 4×4 feature maps represents the features extracted from a 16×16 sub-region of the entire character image. These local features (at the sub-region level) of each character image are extracted as a $4 \times 4 \times 1024$-dimensional matrix.

3.3. Local features at the character level

For extracting features at the character level, we use 3 blocks of a convolutional layer followed by a max-pooling layer to produce 8×8 feature maps from a 64×64 input image as illustrated in Fig. 3. These feature maps are then fed to a fully connected layer to extract features of the entire image of the input character. Consequently, these local features (at the character level) of each character are extracted and represented by a $1 \times 1 \times 1024$-dimensional matrix. In summary, the sub-region level local feature extractor receives a square input image and returns a local feature of size $4 \times 4 \times 1024$ as shown in Fig. 2. The character level local feature extractor also receives a square input image and returns a local feature of size $1 \times 1 \times 1024$ as shown in Fig. 3. Thus, both the sub-region level and the character level local features can be generalised as $L \times L \times D$ shapes.



(a) The model architecture consists of 4 blocks of a convolution layer followed by a max-pooling layer with the numbers of filters being 32, 64, 256, and 1024, respectively. The red block in each convolution layer indicates the convolution kernel.



(b) The red block of CONV1 is the extracted features from the 16x16 sub-region of an input image. Each red block of other convolution layers indicates the features from the red block of its previous convolution layer. Each green block in the convolution layer shows the bounding box from which the sub-region image features could be extracted.

Fig. 2. CNN model for extracting sub-regions level local features.

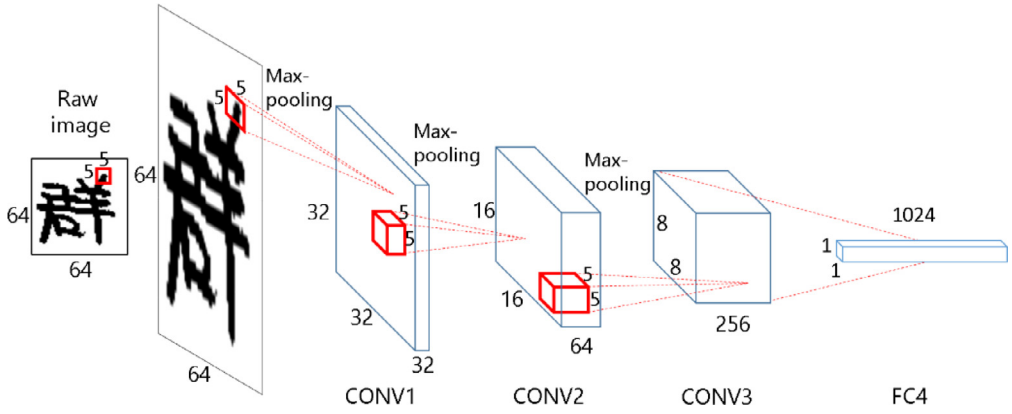


Fig. 3. CNN model for extracting character level local features.
After three convolution and pooling blocks, the extracted features are fed to a fully connected layer FC4 to get the character level local features.

3.4. Global features from multiple characters

Local features from a single character image may not contain enough information for identifying the writer. Therefore, we consider extracting global features for writer identification by aggregating local features. Instead of using the whole document or paragraph, we extract global features from multiple character images. Global features such as structural features, slant, and so on may vary from one text image to another, but we expect that they can be extracted reliably from multiple character images.

Fig. 4 shows three aggregation methods for obtaining global features from n-tuple images, which are fed to CNNs to extract local features. The local feature vector of each image is of the size $L \times L \times D$. Next, an aggregation method is applied to get a global feature vector.

We consider two basic methods for combining the local features (features extracted from the sub-region level and the character level) to form global features (features from multiple characters) as shown in Fig. 4. One is "Average Aggregation" (AA), which works by computing the average of all the values along the depth of n local feature vectors, as shown in Eq. (1).

$$\underset{1\le d\le D}{\text{global\_feature}[d]} = \frac{1}{nL^2}\left(\sum_{t}^{n}\sum_{i}^{L}\sum_{j}^{L}\text{local\_feature}_t[i,j,d]\right) \quad (1)$$

where $global\_feature[d]$ is the dth element of the global feature vector. The $local\_feature_t$ is the tth extracted local feature vector among n extracted local feature vectors from n-tuple images. For AA, each element of the global feature vector is computed by averaging all values of n local feature vectors at depth d. Thus, AA

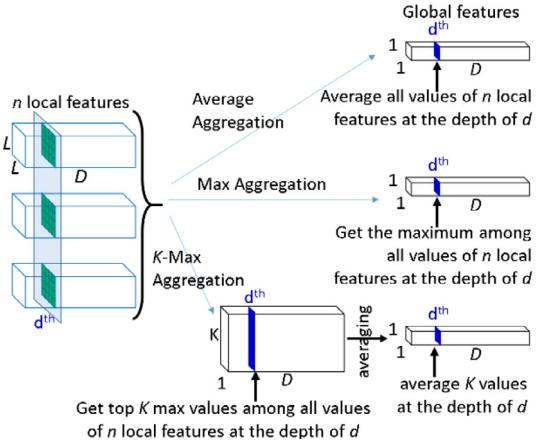


Fig. 4. Three aggregation methods to form the global features.

produces more robust global features compared with the local features.

The other method is "Max Aggregation" (MA), which works by selecting the maximum value along the depth dimension, as shown in Eq. (2).

$$\underset{1\le d\le D}{\text{global\_feature}[d]} = \max_{1\le t\le n;\, 1\le i,j\le L} \{\text{local\_feature}_t[i,j,d]\} \tag{2}$$

MA selects the most relevant local feature from all the elements of n local feature vectors at depth d for writer identification.

For the AA method, the common features shared among different characters or their sub-regions are made more robust by averaging the local features. On the other hand, the MA method can discover relevant features that may not appear in all the local features in some dimension. The difference between these two methods suggests that combining them may result in a better aggregation. We also consider incorporating the method of "Average of K-Max Aggregation" (AKMA), which computes the average of K largest values from all the elements of n local feature vectors at depth d, to yield a global feature as shown in Eq. (3).

$$\underset{1\le d\le D}{\text{global\_feature}[d]} = \frac{1}{K}\sum_{k}^{K}\left(K - \max_{1\le t\le n;\, 1\le i,j\le L} \{local\_feature_t[i,j,d]\}\right) \tag{3}$$

where the K-max function yields K maximum values among all the n local feature vectors at depth d.

Fig. 4 shows three aggregation methods to form a global feature. The global features of these three aggregation methods have the same shape of $1 \times 1 \times 1024$, which is an instance of the general $R \times C \times D$ global shape.

3.5. Random sampling of training patterns

We assume that each writer provides at least Ns samples. For every training epoch, we need to prepare the training data by iterating the following steps p times.

Step 1: Prepare m sets of n-tuples from Ns images (m is the greatest integer less than or equal to Ns/n), where each n-tuple contains n images as shown in Fig. 5.
Step 2: Randomly shuffle all Ns images.
Step 3: Go to Step 1.

Thus, the prepared data contains m sets of n-tuple images p times as shown in Fig. 5. After all the n-tuples of the prepared data are used to train the network, we move to the next epoch.

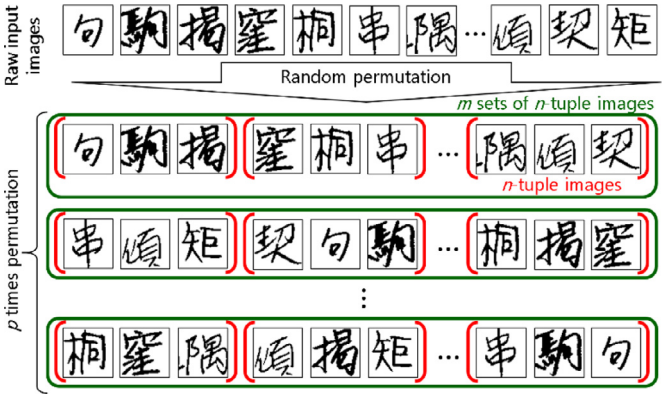


Fig. 5. Randomly sampled training images.

The training epochs are repeated until there is no improvement in the validation accuracy. By reordering images of n-tuples in each iteration, and in each epoch, the network layers are trained to extract writer-specific features without text dependency.

4. Experiments

4.1. Databases

4.1.1. JEITA-HP database

We employed the JEITA-HP handwritten Kanji character pattern database (hereafter, JEITA-HP), which is prepared by Hewlett-Packard Japan and distributed by Japan Electronics and Information Technology Industries Association (JEITA) [19]. The JEITA-HP database consists of Dataset A and Dataset B, containing handwritten character patterns from 480 and 100 writers, respectively. In JEITA-HP, for each writer, there are 3306 handwritten samples (patterns) belonging to 3,214 categories including 2,965 Kanji, 82 Hiragana, 10 numerals, and 157 other categories (English alphabet, Katakana and symbols). Every writer wrote twice for each Hiragana and numeral category and once for each of the other categories, which resulted in $3306 = 2 \times (82 + 10) + 2965 + 157$ samples. In our experiments, we used Kanji character patterns only from the writers in the Dataset A. First, we randomly split 2,965 Kanji character categories into 3 subsets as follows: 2,000 categories for the training set, 400 categories for the validation set and the remaining categories for the testing set. The validation set is used to stop the training process earlier to avoid overfitting the network.

As the number of writers and the number of training samples are different for each experiment, we randomly selected writers as well as samples from the training set at the beginning of each experiment. In the experiments described in Sections 4.3.1, 4.3.2 and 4.3.3, we chose 100 writers at random ($N_{writers} = 100$) while in Section 4.3.4, we chose a variable number of writers $N_{writers}$ to verify the performance of our network. For each writer, we randomly selected a subset of samples ($N_{character\_img}$) for training, instead of using all the 2000 samples in the training set. Thus, the selected subset is different for each experiment.

4.1.2. Firemaker and IAM databases

In order to validate our method for English, and to compare our approach with the handcrafted-feature-based approach of [6], we employed the Firemaker and IAM databases of handwritten text.

For the Firemaker database provided by 250 writers, there are four subsets which are collected by different requirements [26]. The first subset contains the text-copying pages using normal handwriting which was used as the training and validation sets. The second subset contains the text-copying pages using only uppercase characters, which is unusual for handwritten text and

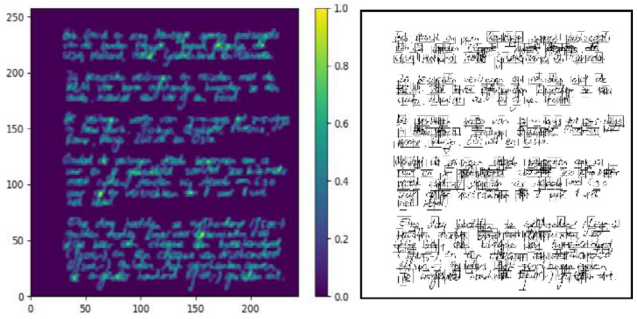


Fig. 6. Sub-images extraction from a handwritten text page in the Firemaker and IAM databases.

therefore was not used in our experiments. The third subset contains forged text, where the writers were asked to write in a different style, which was also not used in our experiments. The fourth subset contains free handwritten text to describe the content of a given cartoon which was used as the testing set. Thus, for each writer, we used his/her text-copy page for training/validating and free handwritten text page for testing.

In the IAM database provided by 650 writers, the character patterns provided by each writer ranged from 1 page (350 writers) to 59 pages (1 writer) [22]. Due to this imbalance in the number of patterns from the writers, we tried to balance it as follows. For the writers who provided only one page, their pages were divided roughly into two halves: the first half for training/validation and the second half for testing. For the writers who provided two or more pages, only the first two pages were selected: the first page was used for training/validation and the second page for testing. Hence, 350 writers had a half page and other 300 writers had one page for training.

For preprocessing pages from the Firemaker and IAM databases, we employed the Otsu binarization method. The handwritten text in these two databases is written cursively, which is difficult to segment and separate into individual characters. However, our method is designed to extract writer-specific features based on the sub-regions of characters, which means that we do not require isolated characters as input. Therefore, we do not need to segment the handwritten text in the Firemaker and IAM databases. Instead, as shown in Fig. 6, we extracted N_{sub_img} sub-images of $k_{sub\_img} \times k_{sub\_img}$ pixels from the binarized page based on the handwritten text probabilities. In order to extract the probability of every pixel as shown in the left image of Fig. 6, we applied a 32×32 average filter on the binarized page image, and then normalized the obtained values into [0,1] range. Using the probabilities in the left image, we sampled 500 ($N_{sub\_img} = 500$) sub-images of size 64×64 ($k_{sub\_img} = 64$). Note that each of these 500 sub-images could be an isolated character, a sub-region of a character, or sometimes even sub-regions of several characters. Similar to the experiments on JEITA-HP, we randomly selected writers and training samples from the training set based on the number of writers ($N_{writers}$) and the number of sub-images (N_{sub_img}), because these values were different in each experiment.

4.2. Parameters

In the experiments of Section 4.3.2, we employed different sizes of n-tuples to find the optimum value of n. Then, we employed the best size of 20-tuples for later experiments. Besides, the number of character images for training ($N_{character\_img}$) is optimized by the experiments describes in Section 4.3.3. For the Firemaker and IAM databases, the value of N_{sub_img} was taken to be 500 as mentioned in Section 4.1.2. The value of m for m-sets is the greatest integer less than or equal to $N_{character\_img}/n$ for the experiments on the JEITA-HP database. For the experiments on the Firemaker and IAM databases, it was the greatest integer less than or equal to N_{sub_img}/n. In all of the following experiments, the number of the training and evaluation iteration was set to 20 for each writer (i.e., $p = 20$) to prepare training patterns for each epoch.

We implemented our network by Tensorflow and executed the training process on a Graphics Processing Unit (GPU) for reducing the training time by the batch size of 10. Thus, a mini-batch contained n-tuples by 10 writers. We also employed the Adam optimizer [20] to train our network, with the learning rate as 10^{-4}, β_1 as 0.9 and β_2 as 0.999. The training process was stopped earlier if there was no improvement in the validation accuracy after 20 epochs.

For evaluating our method on the Firemaker and IAM databases, we sampled a single n-tuple from the handwritten test page by each writer and fed it to the trained network to identify the writer. We made this evaluation process for different n-tuples from the page five times and simply took their average to identify the writer (named as five n-tuples) as presented in Section 4.4.1.

Table 1
Accuracy (%) of local features and different aggregation methods on the JEITA-HP testing set.

Aggregation	Sub-region level features	Character level features
None	48.47	40.16
AA	**99.97**	99.78
MA	92.00	91.78
AKMA ($K = 10$)	99.53	98.56
AKMA ($K = 20$)	99.89	99.45
AKMA ($K = 40$)	99.82	99.78
AKMA ($K = 50$)	99.86	99.80

4.3. Experiments on the JEITA-HP database

In the following subsections, we present our experimental results on the JEITA-HP database. First, each local feature was used to build the writer identification model for evaluating its performance. Secondly, the three aggregation methods mentioned above were employed on each local feature to evaluate their individual performances. These two steps are presented in Section 4.3.1.

Thirdly, the hyper-parameters, which consist of the tuple size (n), the number of character images for training ($N_{character\_img}$) and the number of writers ($N_{writers}$) were optimized as shown in Sections 4.3.2, 4.3.3 and 4.3.4, respectively. According to the results from these sections, we chose the best hyper-parameters for later experiments on the Firemaker and IAM databases (Section 4.4).

4.3.1. Experiments on local features and aggregation methods

To compare the local features and global features with different aggregation methods, experiments were conducted using 500 training characters of each writer to identify 100 writers. The results are shown in Table 1. The first row shows the results of local features as there was no applied aggregation. The following rows show the results with aggregations, that is, global features. The global features are much better than the local features and the sub-region level features are better than the character level features. The average aggregation method achieved the best accuracy of 99.97%. Therefore, in the following experiments, we only applied the average aggregation method to form global features from local features.

The value of K in AKMA method was set to be 10, 20, 40 and 50 for the sub-region level features as well as the character level features. The AKMA method achieved a higher accuracy than the MA method for both the sub-region and character level features as it preserves more information from local features than the MA method. The average aggregation can keep all the information from

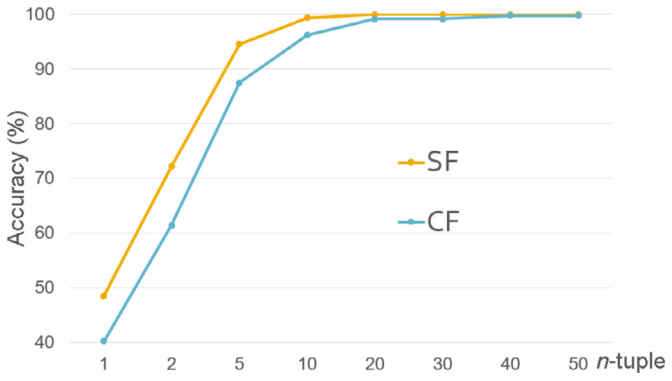


Fig. 7. Accuracy (%) with different tuple sizes n on the JEITA-HP testing set.

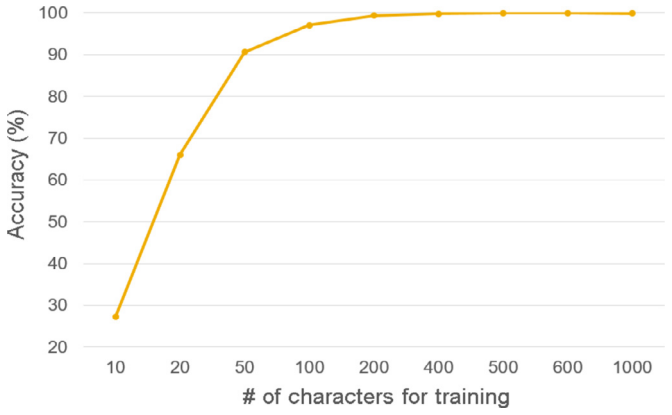


Fig. 8. Accuracy (%) with different number of training characters on the JEITA-HP testing set.

the local features so that its accuracy is always the highest in all the experiments.

4.3.2. Experiments on the tuple size

The following experiments determined the optimum size n for n-tuples. The value of n was varied from 1 to 50 for both the sub-region level features (SF) and character level features (CF) as shown in Fig. 7. We employed 500 characters for each of the 100 writers and applied the average aggregation method as it was found to be the most efficient aggregation method. The SF produced a better recognition rate than the CF and achieved a better identification rate even for smaller n-tuples. With an input from only 10 character images, our method achieved a writer identification accuracy of 99.09%. The highest identification rate was 99.97%, which was achieved by applying SF with 20 character images.

4.3.3. Experiments on the number of training patterns

To determine the fewest characters required for training, we varied the number of training characters ($N_{character\_img}$) from 10 to 1000 for each writer to identify 100 writers. In these experiments, we used 20-tuples ($n = 20$) and the average aggregation method on sub-region level features as these hyper-parameters were confirmed in the experiments described above. The results are presented in Fig. 8. Despite using only 50 characters for training, our method achieved a performance of 92.80%, which suggests that it can extract text-independent writer-specific features using a small number of training samples. The performance is improved by increasing the number of training samples in later experiments: when we increased the number of training samples to 100, an accuracy of 97.52% was obtained.

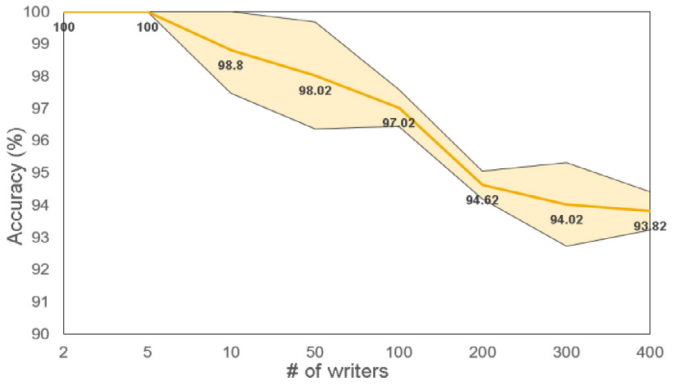


Fig. 9. Accuracy (%) with different number of writers on the JEITA-HP testing set.

4.3.4. Experiments on the number of writers

In these experiments, we varied the number of writers ($N_{writers}$) from 2 to 400, used 20-tuples ($n = 20$), 100 training characters and the average aggregation method on sub-region level features. Fig. 9 shows the results of these experiments, with the orange line showing the mean accuracy in each experiment. The variance is shown by the yellow shaded area between the upper and lower black lines. The performance falls gradually as the number of writers increases. Our method achieved an accuracy of 93.82% using only 100 training characters for identifying 400 writers. Moreover, we repeated each experiment 20 times to obtain the variance. As shown in Fig. 9, the variances are approximately 1 point, which indicates that the training method could learn writer-specific features independent of the text.

4.4. Experiments on the Firemaker and IAM databases

We now present our experiments on the Firemaker and IAM databases with the best hyper-parameters from the above experiments. We reused the optimized tuple size from JEITA-HP database to train the network from the beginning for each experiment on the Firemaker and IAM databases. In addition, we compared the performance of our method with an existing state-of-the-art method based on handcrafted features and clustering [6].

4.4.1. Performance on handwritten English text

For evaluation, a single n-tuple is sampled from each writer and fed into the trained network, as our method does not require a large page or many characters for identifying writers during the evaluation process. We achieved the top-1 accuracy of 92.38% and top-10 accuracy of 97.67% for 250 writers from the Firemaker testing set. For 650 writers from the IAM testing set, we achieved accuracy figures of 90.12% (top-1) and 97.82% (top-10). Using 20-tuples ($n = 20$), our method achieved a performance of 91.81% for 900 writers from the combined databases of Firemaker and IAM.

As the samples of the Firemaker and IAM databases are cursively handwritten text, the extracted sub-images from them usually contain connected characters or sub-regions of connected characters, which requires the network to represent features of connected characters in addition to the features of isolated characters. Therefore, the performance on the Firemaker and IAM databases (92.38% for 250 writers) seems lower than on the JEITA-HP database (94.62% for 200 writers and 94.02% for 300 writers). There may be another reason for a reduced performance on handwritten English databases. The hyper-parameters were optimized for handwritten Japanese text in JEITA-HP, which might not be optimal for English.

In addition, we employed five 20-tuples of sub-images for each test page to evaluate our method which gives better results than

Table 2
Writer identification accuracy (%) on the Firemaker and IAM databases.

Database Number of writers		Firemaker 250	IAM 650	Firemaker + IAM 900
HF-single	Top-1	81	81	80
	Top-10	92	94	92
HF-comb	Top-1	83	89	87
	Top-10	95	97	96
Our method	Top-1	**92.38** **(1.08)**	90.12 (0.88)	**91.81** **(0.69)**
	Top-10	**97.67** **(0.79)**	97.82 (0.50)	**98.07** **(0.51)**
	Top-1 by five 20-tuples	93.56 (0.91)	93.14 (0.70)	94.75 (0.81)

the top-1 accuracy by a single 20-tuple (94.75% for five 20-tuples compared with 91.81% for single 20-tuple). This suggests that we can extract and use more *n*-tuples of sub-images to achieve a better accuracy of writer identification in practice.

4.4.2. Comparison with previously published results

Table 2 shows a comparison between the method of Bulacu and Schomaker [6] and our method. Bulacu and Schomaker [6] obtained results from 50 identification tests on random selections of writers. They mentioned that "For the size of the data sets used here, the writer identification percentages are 3–4 percent confidence interval at a 95 percent confidence level" . Hence, the results of HF-single and HF-comb based recognizers shown in Table 2 are the average results with standard variation from 3% to 4% at a 95 percent confidence level. On the other hand, our method was evaluated 20 times on the test data so that the results of our network in Table 2 are the average identification rates with variances. For all the experiments on the English databases, the top-1 and top-10 performance of our method is higher than the best single handcrafted feature (HF-single) and even the best combination of handcrafted features (HF-comb) in [6]. This suggests that our network can represent writer-specific features for a large number of writers independent of text and language.

In order to verify statistical significance of the difference between the performance of their method and that of our method, we consider the following Eqs. (4) and (5).

$$\hat{\sigma} = \frac{SD^2_{our\_method}}{20} + \frac{SD^2_{prev\_method}}{50} \tag{4}$$

where SD_{prev_method} is the normalized standard variance to [0,1] of their method while SD_{our_method} is that of our method.

$$true\_difference = Acc_{our\_method} - Acc_{prev\_method} \pm Z_{\alpha/2}\sqrt{\hat{\sigma}} \tag{5}$$

where Acc_{prev_method} is the normalized accuracy to [0,1] of their method while Acc_{our_method} is that of our method. $Z_{\alpha/2}$ is the Z score at the (1-α) confidence level. Here, we verify the *true_difference* at 95% confidence level (thus, $\alpha = 0.05$) with $Z_{\alpha/2} = 1.96$.

For the Firemaker and IAM databases, we obtained *true_difference* and confirmed that it is greater than 0. Hence, the null hypothesis is rejected which implies that the performance of our method is significantly different with $\alpha = 0.05$ from Bulacu and Schomaker's method. As shown in Table 2, the top-1 and top-10 results (in boldface) of our method are significantly different from the results of HF-comb on the Firemaker database and on a combination of the Firemaker and IAM databases.

5. Conclusions

We presented here a CNN-based method for text-independent writer identification. Local features were extracted from the whole character patterns as well as their sub-regions, which were aggregated into global features by three different methods. Random sampling was applied to create a large number of training patterns from a limited number of samples for CNN during the training process. This method learned writer-specific features automatically independent of the text through end-to-end training and achieved an identification rate of 99.97% on the JEITA-HP database of Japanese handwritten character patterns for the task of identifying 100 writers. Moreover, our method achieved an accuracy rate of 92.80%, when trained by only 50 characters for 100 writers and 100 characters for 400 writers. This approach overcomes the difficulties of gathering handwritten character patterns in the same category for writer identification.

In the experiments on English databases (Firemaker and IAM), even though the number of writers was 900 and there was an imbalance of handwriting patterns among the writers, our method achieved an accuracy of 91.81%, which is higher than the existing state-of-the-art methods based on handcrafted features and clustering (80% when using a single handcrafted feature and 87% when using combined features). The high performance on the English databases with a large number of writers suggests that the trained network represents the writer-specific features and plays an important role in solving the text-independent writer identification problem.

We plan to expand the scope of this method in the future. In our experiments, we used the best hyper-parameters tuned for handwritten Japanese characters to handwritten English text. However, the accuracy can be improved by tuning the hyper-parameters for English. It would also be interesting to evaluate our writer identification method for other languages or multiple languages [3]. In addition, it would be challenging to apply the method for identifying unknown writers or registering new writers.

Conflict of interest

None.

Acknowledgment

This work is partially supported by JSPS KAKENHI Grant Number JP 18K18068.

References

[1] A. Bensefia, A. Nosary, T. Paquet, L. Heutte, Writer identification by writer's invariants, in: Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition, 2002, pp. 274–279.
[2] D. Bertolini, L.S. Oliveira, E. Justino, R. Sabourin, Texture-based descriptors for writer identification and verification, Expert Syst. Appl. 40 (6) (2013) 2069–2080.
[3] D. Bertolini, L.S. Oliveira, R. Sabourin, Multi-script writer identification using dissimilarity, in: Proceedings of the International Conference on Pattern Recognition, 2017, pp. 3025–3030.
[4] A.A. Brink, J. Smit, M.L. Bulacu, L.R.B. Schomaker, Writer identification using directional ink-trace width measurements, Pattern Recognit. 45 (1) (2012) 162–171.
[5] D. Bruyne, R. Forre, Signature verification with elastic image matching, in: Proceedings of the International Carnahan Conference on Security Technology: Electronic Crime Countermeasures, 1986, pp. 113–118.
[6] M. Bulacu, L. Schomaker, Text-independent writer identification and verification using textural and allographic features, IEEE Trans. Pattern Anal. Mach. Intell. 29 (4) (2007) 701–717.
[7] M. Bulacu, L. Schomaker, A comparison of clustering methods for writer identification and verification, in: Proceedings of the International Conference on Document Analysis and Recognition, 2005, pp. 1275–1279.
[8] M. Bulacu, L. Schomaker, L. Vuurpijl, Writer identification using edge-based directional features, in: Proceedings of the International Conference on Document Analysis and Recognition, 2003, pp. 937–941.
[9] V. Christlein, D. Bernecker, F. Hönig, E. Angelopoulou, Writer identification and verification using GMM supervectors, in: Proceedings of the IEEE Winter Conference on Applications of Computer Vision, 2014, pp. 998–1005.

[10] V. Christlein, D. Bernecker, A. Maier, E. Angelopoulou, Offline writer identification using convolutional neural network activation features, in: Proceedings of the German Conference on Pattern Recognition, 2015, pp. 540–552.
[11] P.C. Chuang, Machine verification of handwritten signature image, in: Proceedings of the International Conference On Crime Countermeasures Science and Engineering, 1977, pp. 105–109.
[12] R. Collobert, J. Weston, A unified architecture for natural language processing, in: Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 160–167.
[13] C. Djeddi, L.-S. Meslati, I. Siddiqi, A. Ennaji, H.El Abed, A. Gattal, Evaluation of texture features for offline arabic writer identification, in: Proceedings of the 11th IAPR International Workshop on Document Analysis Systems, 2014, pp. 106–110.
[14] S. Fiel, R. Sablatnig, Writer identification and retrieval using a convolutional neural network, in: Proceedings of the International Conference on Computer Analysis of Images and Patterns, 2015, pp. 26–37.
[15] K. Franke, O. Biinnemeyer, T. Sy, Ink texture analysis for writer identification, in: Proceedings of the International Workshop on Frontiers in Handwriting Recognition, 2002, pp. 268–273.
[16] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[17] C. Hertel, H. Bunke, A set of novel features for writer identification, Audio-Video-Based Biometric Person Authent. 2688 (2003) 679–687.
[18] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554.
[19] T. Kawatani, H. Shimizu, Handwritten kanji recognition with the LDA method, in: Proceedings of the 14th International Conference on Pattern Recognition, 2, 1998, pp. 1301–1305.
[20] D.P. Kingma, J.L. Ba, Adam: a method for stochastic optimization, in: Proceedings of the 3rd International Conference for Learning Representations, 2015, pp. 1–15.
[21] U.-V. Marti, R. Messerli, H. Bunke, Writer identification using text line based features, in: Proceedings of the 6th International Conference on Document Analysis and Recognition, 2001, pp. 101–105.
[22] U.V. Marti, H. Bunke, The IAM-database: an English sentence database for offline handwriting recognition, Int. J. Doc. Anal. Recogn. 5 (1) (2003) 39–46.
[23] R. Plamondon, G. Lorette, Automatic signature verification and writer identification - the state of the art, Pattern Recognit. 22 (2) (1989) 107–131.
[24] H.E.S. Said, T.N. Tan, K.D. Baker, Personal identification based on handwriting, Pattern Recognit. 33 (1) (2000) 149–160.
[25] A. Schlapbach, H. Bunke, A writer identification and verification system using HMM based recognizers, Pattern Anal.Appl. 10 (1) (2007) 33–43.
[26] Schomaker, L.R.B., Vuurpijl, L., 2000. Forensic Writer Identification: A Benchmark Data Set and a Comparison of Two Systems, Technical Report. Nijmegen University, Tilburg, The Netherlands.
[27] I. Siddiqi, N. Vincent, Combining contour based orientation and curvature features for writer recognition, in: Proceedings of the International Conference on Computer Analysis of Images and Patterns, 2009, pp. 245–252.
[28] M. Sreeraj, S.M. Idicula, A survey on writer identification schemes, Int. J. Comp. Appl. 26 (2) (2011) 23–33.
[29] S.N. Srihari, S.H. Cha, H. Arora, S. Lee, Individuality of handwriting: a validation study, in: Proceedings of the International Conference on Document Analysis and Recognition, 2001, pp. 106–109.
[30] G.X. Tan, C. Viard-Gaudin, A.C. Kot, Automatic writer identification framework for online handwritten documents using character prototypes, Pattern Recognit. 42 (12) (2009) 3313–3323.
[31] X. Wu, Y. Tang, W. Bu, Offline text-independent writer identification based on scale invariant feature transform, IEEE Trans. Inf. Forensics Secur. 9 (3) (2014) 526–536.
[32] W. Yang, L. Jin, M. Liu, DeepWriterID: an end-to-end online text-independent writer identification system, IEEE Intell. Syst. 31 (2) (2016) 45–53.
[33] I. Yoshimura, M. Yoshimura, Writer identification based on the arc pattern transformation, in: Proceedings of the 9th International Conference on Pattern Recognition, 1988, pp. 35–37.
[34] M. Yoshimura, F. Kimura, I. Yoshimura, Experimental comparison of two types of methods of writer identification, IEICE Trans. E65 (6) (1982) 345–352.