DeepWriterID: An End-to-End Online Text-Independent Writer Identification System

Weixin Yang, Lianwen Jin, and Manfei Liu, *South China University of Technology*

The end-to-end handwriting-based writer identification system DeepWriterID employs a deep convolutional neural network and achieves accuracy rates of 95.72 percent for Chinese text and 98.51 percent for English text.

The rapid adoption of touchscreen mobile terminals and pen-based interfaces has increased the demand for handwriting-based writer identification systems, particularly in the areas personal authentication and digital forensics. The writer identification task involves determining a list of candidate writers based on the degree of similarity between their handwriting and a sample of unknown authorship.[1] The wide range of applications for such systems include distinguishing forensic trace evidence, performing mobile bank transactions, and authenticating access to networks. Because most of these applications are closely related to assuring personal and property security, handwriting identification merits more attention from academia and industry.

Identifying a writer's handwriting is a highly challenging problem in the AI and pattern recognition fields. Conventionally, handwriting identification systems follow a sequence of data acquisition, data preprocessing, feature extraction, and classification.[2] Thus far, handwriting identification research has been focused on two categories: offline and online. Offline handwritten materials are considered more general but are harder to identify because they contain merely scanned image information. In contrast, systems dealing with online handwriting are popular and expected to achieve better performance with the development of devices that now make it possible to acquire handwriting with rich information (such as position, velocity, pressure, and altitude).

Handwriting identification systems can also be categorized as either text-dependent or text-independent. Text-dependent methods provide high accuracy but are inapplicable in cases where handwriting text content is absent, whereas text-independent methods are robust against handwriting text contents but require a large amount of data to ensure their generalized applicability. In the evaluation stage, different lengths of source materials (for example, a character, text line, page, and full document) result in varying levels of difficulty in acquiring sufficient information for identification. In addition, multiple languages can be evaluated individually or integrally, leading to different requirements for a system's generalized applicability.

Although numerous researchers have studied handwriting identification and have

1541-1672/16/$33.00 © 2016 IEEE
Published by the IEEE Computer Society

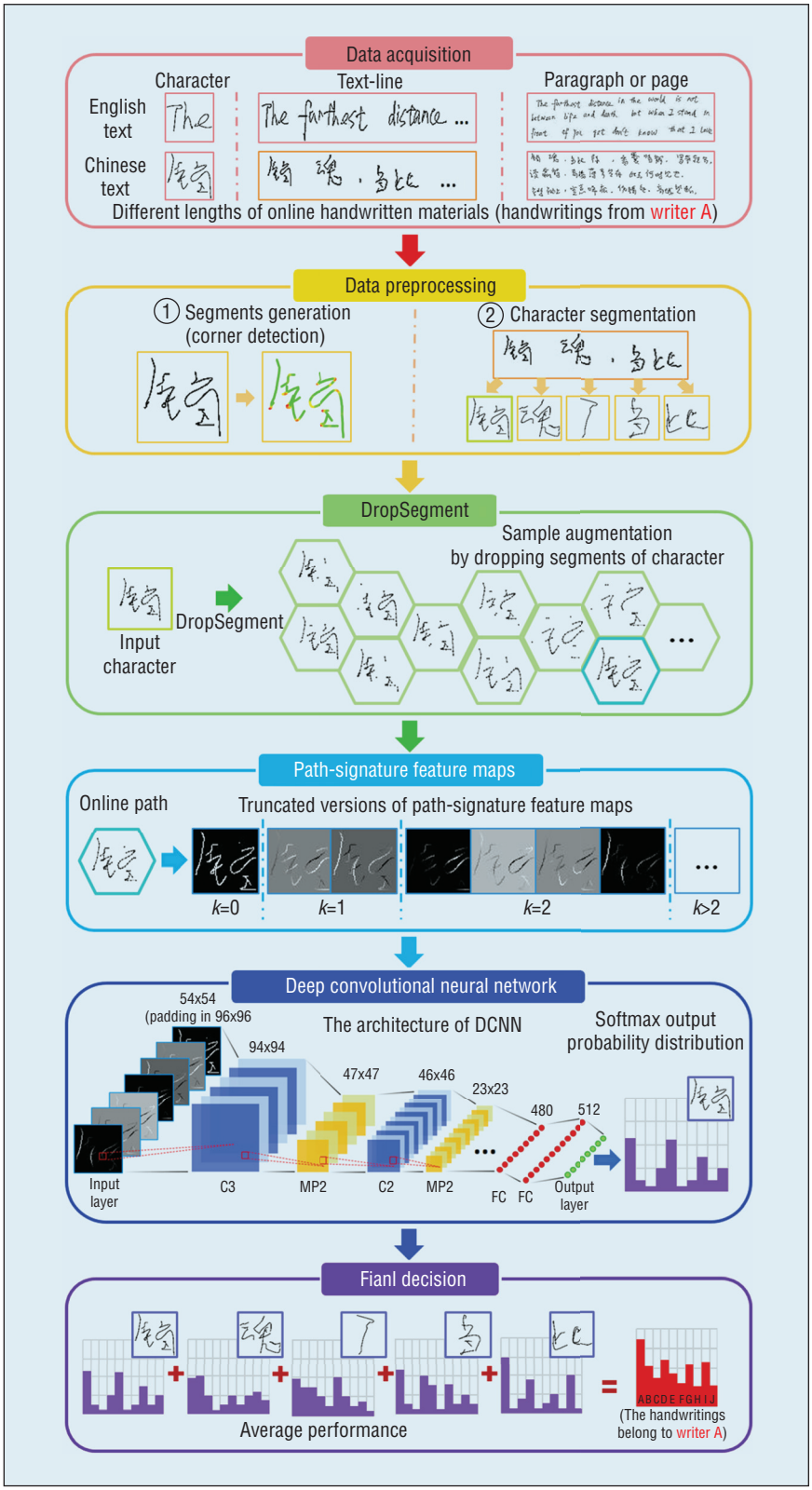

Figure 1. DeepWriterID pipeline. Our proposed online handwriting-based writer identification system incorporates the DropSegment data augmentation technique after data acquisition and preprocessing. This illustration demonstrates the DeepWriterID pipeline for Chinese and English text examples.

achieved tremendous progress,[3] these problems are still unsolved given the variability of real-world conditions, such as insufficient data, source materials of different lengths, and multiple languages in handwritten material. In this article, we propose an end-to-end online text-independent system called DeepWriterID that uses a deep convolutional neural network (CNN) to address these problems. A key feature of DeepWriterID is a new method we're proposing called DropSegment, which randomly removes several segments from the characters of an original handwriting sample while retaining the identity information contained in it. We designed DropSegment to achieve data augmentation, improve the generalized applicability of CNN, and prevent model overfitting. For sufficient feature representation, we further introduce path-signature feature maps to improve performance. We conducted experiments using DeepWriterID on the National Laboratory of Pattern Recognition (NLPR) handwriting database (www.cbsr.ia.ac.cn/english/Databases.asp). Even though we only use pen-down pen-position information for the given handwriting samples, we achieved identification rates of 95.72 percent for Chinese text and 98.51 percent for English text.

Motivation for DeepWriterID

When building the DeepWriterID handwriting identification system (see Figure 1), we encountered the following four problems.

First, training data are scarce. Sufficient handwriting data are necessary not only for text-independent writer identification systems to ensure content-free performance, but also for deep neural network (DNN)-based feature representation models to achieve the best performance. However, collecting them is obtrusive

and tedious for users in the real world, especially when paragraphs or pages of material are required.

Second, the generalization capability is insufficient. This leads to poor system performance, and the use of stroke structure is often blamed. While stroke structure can sometimes be helpful in distinguishing the identity of writers in text-dependent systems, its use limits generalizability when faced with handwriting that has diverse structures (such as natural or multilingual handwriting).

Third, the segmentation problem is vexing. The lengths of identification materials and the sizes of characters differ in practical applications. Either over- or under-segmentation results in different character sizes, which adversely affects identification performance. In addition, when faced with multiple languages, basic units differ in size (for example, Chinese characters versus English words), introducing further difficulties in proper segmentation.

Finally, ensemble models are costly. Ensemble methods provide outstanding results, but the storage requirements are too large for use in practical use, especially for mobile device applications. It's necessary to find a flexible method to achieve similarly successful results while maintaining a constant storage size.

Inspired by Dropout, which randomly omits some of the neuros outputs of a neural network in each training stage in order to prevent overfitting,[5] we developed the DropSegment method for handwriting identification to alleviate these problems.

DropSegment Method

DropSegment is an efficient data augmentation technique in which each original character has at least one stroke, and each stroke contains a certain number of segments, defined by adopting some segmentation methods. For example, we predefine a stroke's corner points as its segmentation points. Suppose that an original character has m strokes and that its ith stroke contains $s_i (s_i \in \mathbb{N}^+)$ segments. If $d_i (0 \le d_i \le s_i, d_i \in \mathbb{N}^+)$ segments are dropped from this stroke, then the number of possible combinations of the remaining segments will be

$$C_{s_i}^{d_i} = s_i ! / d_i !(s_i - d_i)!. \quad (1)$$

In practical handwriting identification, the stroke structure of various source materials (for example, for multiple languages and handwriting ranging from neat to scrawled strokes) is no longer an invariant and could even be harmful to identification.

According to the addition principle in combinatorics, the number of all possible combinations derived from the ith stroke is the sum of Equation 1 over all d_i, yielding

$$\sum_{d_i=0}^{s_i} C_{s_i}^{d_i} = 2^{s_i}. \quad (2)$$

Then, according to the multiplication principle, the number of new characters generated from the prototype character is the product of Equation 2 over all strokes i $(1 \le i \le m,\ i \in \mathbb{N}^+)$, which is expressed as

$$N(m,S) = \left(\prod_{i=1}^{m} \sum_{d_i=0}^{s_i} C_{s_i}^{d_i} \right) - C_{\hat{s}}^{\hat{s}} = 2^{\hat{s}} - 1, \quad (3)$$

where the sequence $S = \{s_1, s_2, \ldots, s_m\}$ contains the number of segments in each stroke of the original character. The sum of the segment counts is $\hat{s} = \sum_{i=1}^{m} s_i$. Because we would never remove all the segments of a character, we exclude the $C_{\hat{s}}^{\hat{s}}$ in Equation 3.

To illustrate our approach, let's look at two examples processed using DropSegment as follows: a three-stroke character with $S = \{2, 3, 4\}$ can generate 511 new characters (calculated by $(2^2)^1 \times (2^3)^1 \times (2^4)^1 - 1$), and an eight-stroke character with all $s_i = 3$ (i, 1, 2 , … 8) can generate more than 1.67×10^7 new characters, calculated by $(2^3)^8 - 1$ based on Equation 3. Therefore, with some number of segments randomly dropped from the characters, the remaining segments recombine to form a massive number of new characters that appear diverse, thereby achieving data augmentation. Figure 1 shows examples of the new characters together with their prototype character.

In practical handwriting identification, the stroke structure of various source materials (for example, for multiple languages and handwriting ranging from neat to scrawled strokes) is no longer an invariant and could even be harmful to identification. By separating strokes, DropSegment can prevent the structural information from being considered and maintain the writer discriminant information, thus improving its generalized applicability. Additionally, DropSegment is robust to over- and under-segmentation problems. When DropSegment is adopted, all the

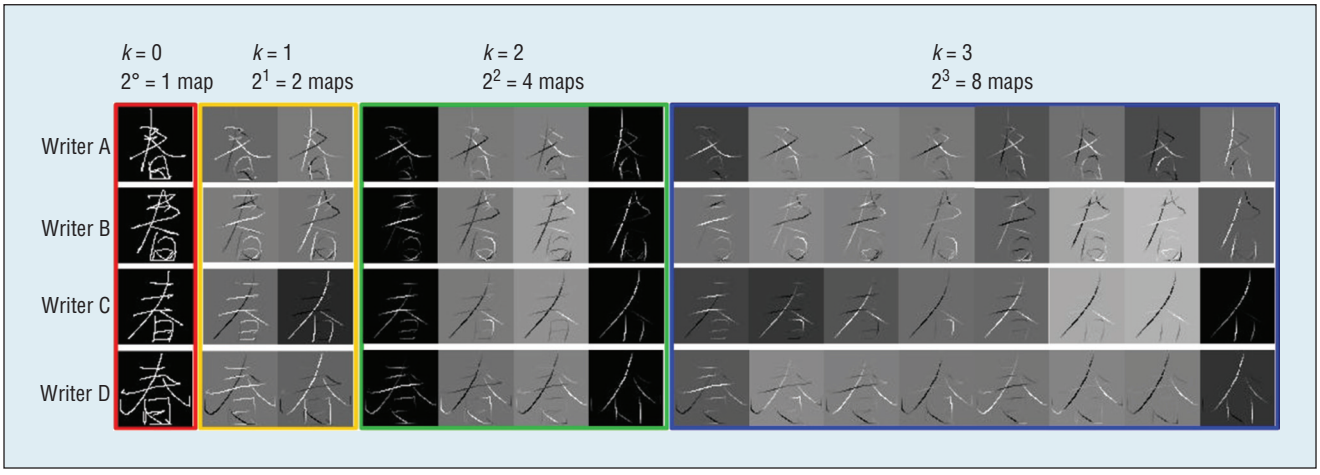

Figure 2. Path-signature feature visualization. The parameter *k* denotes the *k*th level iterated integral of the path signature. The rows denote different writers, and the columns present the maps of different levels of the path-signature feature.

characters will have their segments omitted probabilistically, so the segmentation method can be relatively crude. Moreover, in the testing stage, our approach elastically generates a certain number of new test samples to provide multiple predictions. Averaging these predictions is expected to improve performance with no extra storage consumption.

Compared with the DropStroke method,[7] which removes strokes from a character, DropSegment is more robust for three reasons. First, in fast text-line handwriting or cursive handwritten personal signatures, strokes that are formally separated may be concatenated, so wiping off the whole stroke can lead to too much information loss. Second, the basic units in some languages have scarce strokes, such as English letters, accounting for the limited generalizability of the DropStroke method when faced with multilingual handwriting. Third, because segments are structures that are more detailed than strokes, DropSegment can generate more possible samples than DropStroke. Note that in Equation 3, whenever $d_i = s_i$, the effect of DropSegment is equivalent to that of DropStroke, accounting for the generalized applicability of DropSegment.

Generating Path-Signature Feature Maps

The path-signature, pioneered by Kuo-Tsai in the form of the collection of iterated integrals and developed in recent years by Terry Lyons and his colleagues to play a fundamental role in rough theory,[9–11] is able to extract a sufficient quantity of information concealed in a path of finite length (such as speech traces or online handwritings) to solve any linear differential equation. For handwriting analysis, the path-signature feature was first introduced by Benjamin Graham to address handwritten character recognition.[4] In DeepWriterID, we sought a way to extract further valuable information for the handwriting identification problem using path-signature feature maps.

Given a pen segment P of finite length writing in the plane $\mathbb{R}^2$, we can denote a continuous mapping $P:[0,T]\to\mathbb{R}^2$. Letting $k\in\mathbb{N}$ and $0<\tau_1<\ldots<\tau_k<T$, the kth level iterated integral of path P can be represented as

$$P_{0,T}^{k}=\int_0^T\int_0^{\tau_k}\ldots\int_0^{\tau_2}dP_{\tau_1}\otimes\cdots\otimes dP_{\tau_k}. \quad (4)$$

The dimension of $P_{0,T}^{k}$ is 2^k. For algebraic reasons, when $k = 0$, the signature is constant at 1, denoting the original input, while $k = 1$ and $k = 2$ represent the path displacement and path curvature, respectively. When P is a straight line, the iterated integrals $P_{0,T}^{k}$ can be calculated iteratively by

$$P_{0,T}^{k}=\begin{cases}(P_{0,T}^{k-1}\otimes\Delta_{0,T})/k & k\geq 1\\ 1 & k=0\end{cases}, \quad (5)$$

where $\Delta_{0,T}$ denotes the path displacement. By increasing the number of integral iterations, higher levels of path information can be revealed, but the dimension of the feature (calculated as 2^k for the kth level of the signature) increases exponentially as well. Therefore, to enrich the path representation while keeping the computational time for features to a minimum, our solution is to create a signature collection, which requires combining different levels of integral iterations. This is expressed by

$$F_{0,T}^{n}=(P_{0,T}^{0},P_{0,T}^{1},P_{0,T}^{2},\ldots,P_{0,T}^{n}), \quad (6)$$

where n ($n\in\mathbb{N}$) is the level at which the signature is truncated. At this stage, each sample point along the segment can generate a set of signature values with the truncated level n. Equation 6's dimension (that is, the number of feature maps) can be calculated as $2^{n+1} - 1$.

To demonstrate feature maps intuitively, we assign the pixels of the pen trajectory with the corresponding

signature image histogram equalization for each feature map. Figure 2 shows the visualization. The rows denote different writers, and the columns present the maps of different levels of the path-signature feature. Note that the maps of the first three iterated integrals seem similar in magnitude among the writers, while the differences between the maps beyond the second level ($k > 2$) are appreciable (for example, the third level). This supports our hypothesis that higher levels of the path signature can contain information that will contribute positively to handwriting identification.

Deep CNN Architecture and Configuration

To achieve integrated optimization in the overall pipeline, we employ a deep CNN (DCNN) model that exploits the spatial sparsity of handwriting, as described in earlier work.[4] The idea of spatial sparsity derives from the online trajectory's scarce foreground pixels compared with the numerous background pixels, which take values of zero so the background computation can be avoided to save time. As Figure 1 shows, our DCNN architecture includes five convolutional layers, each of which is accompanied by a max-pooling (MP) layer. We fix the size of the convolutional filter at 3×3 (denoted by C3) for the first layer and 2×2 (C2) for the others, with a stride of 1 pixel. The MP window size is 2×2 (MP2) with a stride of 2 pixels. The small filter size and pooling size enable the network to retain valuable information inside. Our DCNN renders the input data into a 54×54 bitmap and puts it in the center of a 96×96 grid, where the extra pixels beyond the bitmap are the padding pixel budget for all borders in the convolutional layers. The number of convolutional filter kernels is 80 for the first layer and is incremented by 80 after each MP. At the top of our network, two fully connected (FC) layers with 480 and 512 units in size, respectively, are included in the design to better characterize the complicated biometric information and to provide additional nonlinearity to the network. For activation functions, rectified linear units are used for neurons in the convolutional and FC layers, and softmax is used for the output layer.

> After all the strokes have been separated into segments, the character segmentation follows, using the assumption that we can roughly estimate the width of a character by its height.

Following the representation term found elsewhere,[4] our network architecture can be represented as $M \times 96 \times 96$ Input – 80 C3 – MP2 – 160 C2 – MP2 – 240 C2 – MP2 – 320 C2 – MP2 – 400 C2 – MP2 – 480 FC – 512 FC – Output. Here, M denotes the number of input channels, which is equal to the number of signature feature maps.

Experimental Results

To evaluate DeepWriterID's performance, we used the NLPR handwriting database (www.cbsr.ia.ac.cn/english/Databases.asp), which contains four pages of Chinese text and four pages of English text from each writer. We performed experiments on two subsets: dataset I (DB I), which was contributed by 187 writers, includes two free-content Chinese pages for training and one fixed-content Chinese page for testing, and dataset II (DB II), which was contributed by 134 writers, includes two free-content English pages for training and one fixed-content English page for testing.

The samples in the NLPR database were collected by a Wacom Intuos2 tablet and include rich sequential information including pen-position, pen-down and pen-up states, azimuth, altitude, and pressure. In general, the only consistently available information on most touchscreen mobile devices is the pen-down handwriting position, so we deliberately ignored the other handwriting information and conducted our experiments using the pen-down pen position.

Data Preprocessing

The samples in the NLPR database are presented on pages, and the handwriting can be regular or cursive, so we use segmentation as a data preprocessing step in DeepWriterID to unify the various input data. The segmentation is twofold: segment generation and pseudo-character segmentation.

Segment generation. For each stroke on a page, we use a fast and efficient corner detection algorithm[6] for segment generation. To detect corners, this algorithm assumes that the directions of the forward and backward vectors of a noncorner point will cancel each other. A bending value is defined as

$$\beta = \max \begin{pmatrix} |x_{i+k} + x_{i-k} - 2x_i|, \\ |y_{i+k} + y_{i-k} - 2y_i| \end{pmatrix} / 2k, \quad (7)$$

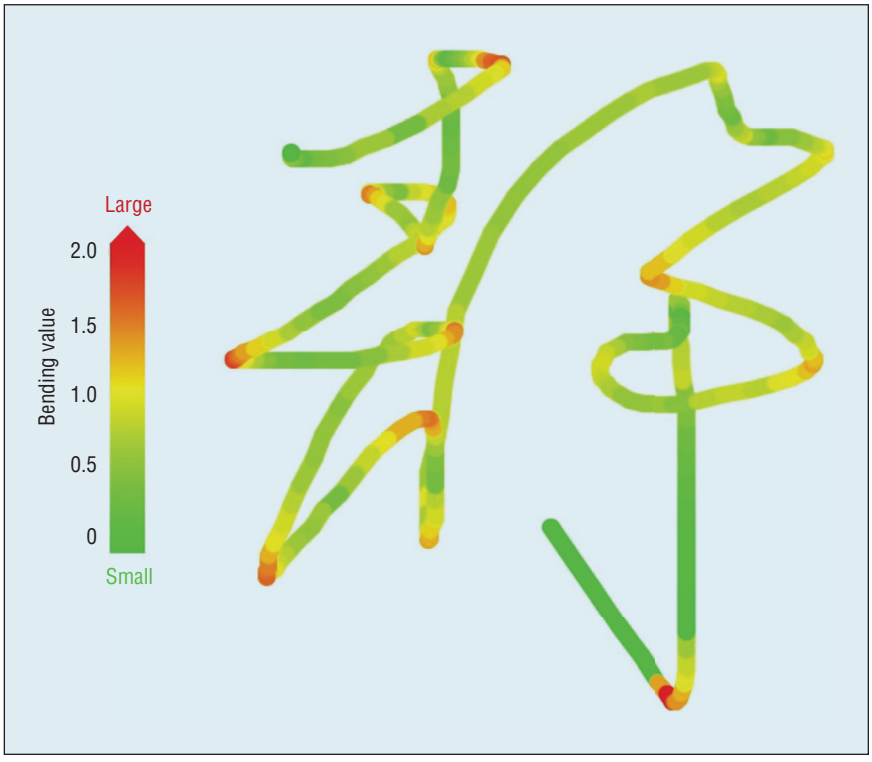

Figure 3. A heat map of bending values. The color red on the trajectory indicates a large bending value, whereas green denotes a relatively small value. The points with local maximum bending values are considered as potential segmentation points.

where (x_i, y_i) are the trajectory points after interpolation and (x_{i+k}, y_{i+k}) and (x_{i-k}, y_{i-k}) are the corresponding kth forward and backward points, respectively. For this work, we set the value of k to 2. Bending values assess the degree of curvature and are calculated for each point on the trajectory, and the local maximum bending values are defined to be the corners. Then, each stroke is divided into different segments according to these corners. Figure 3 illustrates the heat map of bending values for a sample pen trajectory.

Pseudo-character segmentation. After all the strokes have been separated into segments, the character segmentation follows, using the assumption that we can roughly estimate the width of a character by its height. In fact, we don't need an accurate segmentation of a character for the writer identification task, so we apply a pseudo-character segmentation process without losing the discriminant information for different writers.

To unify the size of CNN inputs, the height-to-width ratio should be the same. We thus determined this ratio to be 1 for simplicity. From our experiments, we found that a ratio value between [1, 1.3] will also produce similar results.

During the pseudo-character segmentation, the average height is measured first. Then, each segment will be sequentially assigned to form a pseudo-character. When a character's width exceeds the average height after adding a segment, this segment is regarded as the beginning of a new character. Pseudo-characters formed this way aren't always meaningful, but they're uniformly similar to a square in shape. Because the input samples required for the writer identification task are independent of their meanings, it isn't essential to employ accurate segmentation according to meaningful characters. The samples generated this way can contain non-characters or consist of two to three letters (for English text) instead of a single letter or a word. Still, we found that such samples don't prevent them from being useful and effective in CNN training data for writer identification.

DropSegment Implementation

After data preprocessing, all the data on pages have been separated into characters with segments. In our implementation, for each iteration, we first randomly select a mini batch of these characters. If a character contains m ($m \in \mathbb{N}^+$) segments, a random number r ($r \in [0, \lfloor m/2 \rfloor]$, $r \in \mathbb{N}$) of the m segments will be removed. The removed segments are also randomly chosen. Using this setting, DropSegment not only generates a considerable number of samples but also prevents too much information loss. For the two examples in the previous section, the number of generated characters is reduced to 256 and 9.74×10^6, respectively.

The remaining segments, however, appear fragmented and could destroy the writing styles, so combining the segments is necessary. For each character, after segment removal, the remaining segments that are concatenated with each other in the prototype character will be recombined to form longer segments. After that, each isolated segment will individually become a new stroke. Hence, DropSegment can maintain most of the identity information while achieving data augmentation.

Investigating the Path-Signature Feature

In our experiment, we employed the single DCNN we described earlier. During the training stage, before

extracting path-signature feature maps, we adopted a random mix of affine transformations (translation, rotation, and scaling) as the elastic distortion to achieve further data augmentation and generalized applicability. The mini-batch size was set to 100, and the Dropout rates for the last four layers were empirically set to 0.3, 0.4, 0.5, and 0.5, respectively. We spent a week training the DCNN on a PC with a GTX 980 GPU. The DCNN was trained on character-level samples, and noting that the network's softmax output can be treated as a probability distribution of all classes, we thus averaged the outputs of all the characters from each page to give the final prediction at the page level, as Figure 1 illustrates.

The baseline method (which we denote as Bitmap) in our experiments renders the online handwritten characters as offline bitmaps to train the DCNN. We found that a DCNN with additional path-signature features can incorporate prior knowledge in the representation and significantly improve performance. Thus, we evaluated the effect of the path-signature feature on handwriting identification using the Chinese text in DB I. Figure 4 shows the curves of the page-level test error rate. The path-signature feature at truncated level n is denoted by Sign n. We discovered that the path-signature feature together with Bitmap can greatly enhance the network representation and beat the baseline by a large margin. The higher truncated signature levels provide more subtle handwriting information, accounting for their better results, but the improvements become more negligible with the exponentially increasing feature dimensions.

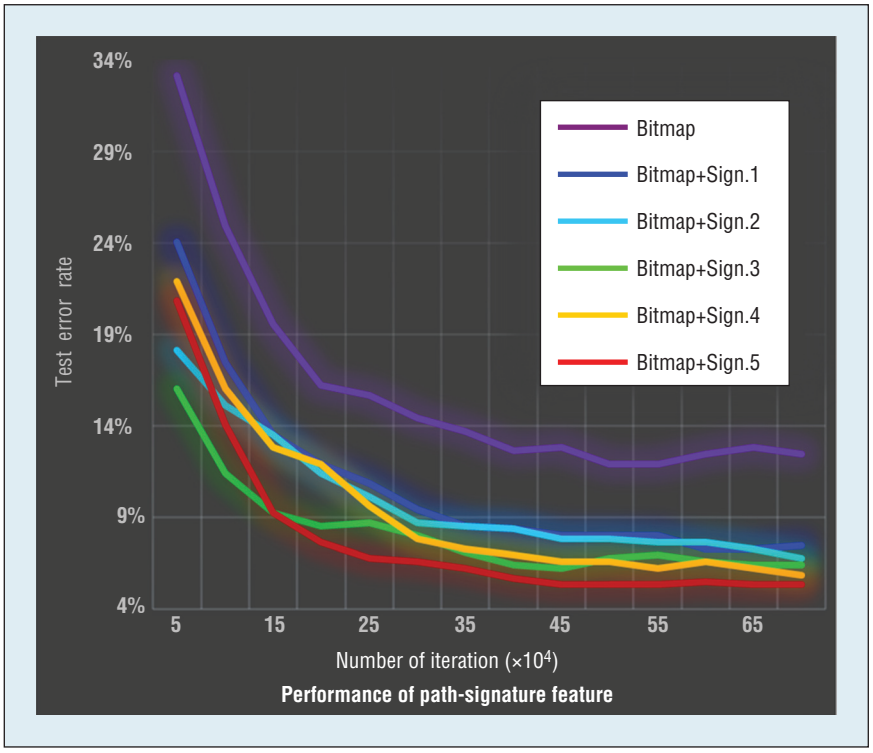

Figure 4. Performance of different truncated levels of the path-signature feature. The page-level results shown are the top test error rates on the Chinese dataset (DB I).

Table 1. Average writer identification rates (%) of different methods with and without DropSegment for the Chinese text (DB I).

Methods	Without DropSegment (1 test)	With DropSegment (1 test)	With DropSegment (20 tests)
Bitmap baseline	87.70	89.30	89.84
Bitmap + Sign 1	91.98	93.05	93.05
Bitmap + Sign 2	92.51	93.05	93.58
Bitmap + Sign 3	93.05	94.12	94.65
Bitmap + Sign 4	94.12	94.65	94.12
Bitmap + Sign 5	94.65	95.19	95.72

Investigating the DropSegment Method

Next, we conducted experiments on the Chinese text in DB I to evaluate the proposed DropSegment method. DropSegment is flexible and produces varying results as it randomly generates a certain number of new test samples from the prototypes. Therefore, it makes sense to combine these results to achieve better performance without retraining new models. We averaged 20 of these predictions at the test stage in our experiments. Table 1 gives the page-level performance, showing that all the identification rates are markedly improved over those without DropSegment.

Figure 5 compares three previous methods with our proposed method. The first method, introduced by Marcus Liwicki and his colleagues, combines features with point- and stroke-level information.[12] The second method, proposed by Marius Bulacu and Lambert Schomaker, uses both texture- and allograph-level features and evaluates offline samples.[13] The third method, which previously

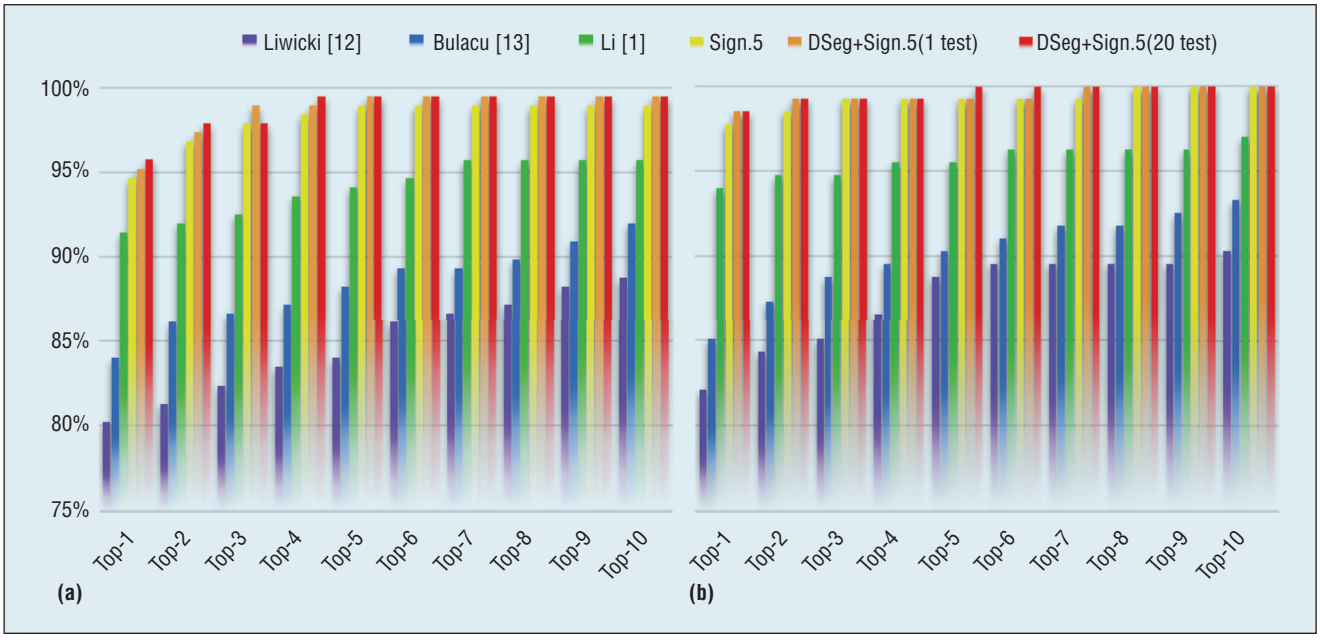

Figure 5. The Top-10 average writer identification rates of different methods: (a) Chinese text (DB I) and (b) English text (DB II).

THE AUTHORS

Weixin Yang is a PhD student in information and communication engineering at the South China University of Technology. His research interests include machine learning, handwriting analysis and recognition, computer vision, and intelligent systems. Yang received a BS in electronics and information engineering from South China University of Technology. Contact him at wxy1290@163.com.

Lianwen Jin is a professor in the College of Electronic and Information Engineering at the South China University of Technology. His research interests include handwriting analysis and recognition, image processing, machine learning, computer vision, and intelligent systems. Jin received a PhD in information and communication engineering from the South China University of Technology. He has received the New Century Excellent Talent Program of MOE Award and the Guangdong Pearl River Distinguished Professor Award, and is a member of the IEEE Computational Intelligence Society, IEEE Signal Processing Society, and IEEE Computer Society. Contact him at lianwen.jin@gmail.com.

Manfei Liu is a master's student in circuits and systems at the South China University of Technology. Her research interests include machine learning and computer vision. Liu received a BS in electronics and information engineering from South China University of Technology. Contact her at liu.mf@foxmail.com.

produced the best results for the NLPR database, uses hierarchy models with features extracted by the shape primitive probability distribution function.[1]

The best results previously reported were 91.50 percent for DB I (Chinese text) and 93.60 percent for DB II (English text), respectively.[1] These results were mostly achieved by combining sufficient features extracted from rich online information including pressure, altitude, azimuth, velocity, and pen-position in both the pen-down and pen-up states. However, our approach attained higher writer identification rates of 95.72 percent for the Chinese text and 98.51 percent for the English text, indicating relative error reduction rates of 49.6 percent and 76.5 percent, respectively, and we achieved this even though we only used pen-position information.

The proposed DropSegment technology makes it possible to train excellent CNNs even with inadequate data, as is often the case with writer identification. Furthermore, it achieves promising ensemble performance without training or needing to store additional network models.

It's worth noting that the proposed method doesn't consider rejection of unknown writers, which is an important issue for further study. One possible way to reject unknown writers is to analyze the confidence measurement learned by the CNN. Moreover, DeepWriterID should also allow the registration of new writers who aren't included in the training database. This would provide a flexible way to absorb new characteristics and offer a means of online or incremental learning, which is also worth studying in the future. ■

Acknowledgments

This research is supported in part by Natural Science Foundation of China (NSFC grant 61472144), Guangdong Science and Technology Plan (GDSTP,grants 2014A010103012 and 2015B010101004), and Guangdong University Pearl Scholar foundation (GDUPS 2011).

References

1. B. Li, Z. Sun and T.N. Tan, "Hierarchical Shape Primitive Features for Online Text-Independent Writer Identification," *Proc. 10th Int'l Conf. Document Analysis and Recognition (ICDAR)*, 2009, pp. 986–990.
2. H.E. Said, T.N. Tan, and K.D. Baker, "Personal Identification Based on Handwriting," *Pattern Recognition*, vol. 33, no 1, 2000, pp. 149–160.
3. M. Sreeraj and S.M. Idicula, "A Survey on Writer Identification Schemes," *Int'l J. Computer Applications*, vol. 26, no. 2, 2011, pp. 23–33.
4. B. Graham, "Sparse Arrays of Signatures for Online Character Recognition," preprint, Cornell Univ. Library, 2013; http://arxiv.org/pdf/1308.0371.pdf.
5. G.E. Hinton et al., "Improving Neural Networks by Preventing Co-adaptation of Feature Detectors," preprint, Cornell Univ. Library, 2012; http://arxiv.org/pdf/1207.0580.pdf.
6. M.J.J. Wang et al., "Corner Detection Using Bending Value," *Pattern Recognition Letters*, vol. 16, no. 6, 1995, pp. 575–583.
7. W.X. Yang, L.W. Jin, and M.F. Liu, "Chinese Character-Level Writer Identification with Path Signature Feature, DropStroke and Deep CNN," *Proc. 13th Int'l Conf. Document Analysis and Recognition (ICDAR)*, 2015, pp. 546–550.
8. K.-T. Chen, "Integration of Paths: A Faithful Representation of Paths by Noncommutative Formal Power Series," *Trans. Am. Mathematical Soc.*, 1958, pp. 395–407.
9. B. Hambly and T. Lyons, "Uniqueness for the Signature of a Path of Bounded Variation and the Reduced Path Group," *Annals of Mathematics*, vol. 171, no. 1, 2010, pp. 109–167.
10. T. Lyons and Z. Qian, *System Control and Rough Paths*, Oxford Univ. Press, 2002.
11. T. Lyons, "Rough Paths, Signatures and the Modelling of Functions on Streams," preprint, Cornell Univ. Library, 2014; http://arxiv.org/pdf/1405.4537v1.pdf.
12. M. Liwicki et al., "Writer Identification for Smart Meeting Room Systems," *Document Analysis Systems VII*, LNCS 3872, Springer, 2006, pp. 186–195.
13. M. Bulacu and L. Schomaker, "Text-Independent Writer Identification and Verification Using Textural and Allographic Features," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 4, 2007, pp. 701–717.