Letter-level Writer Identification

Zelin Chen[1], Hong-Xing Yu[1], Ancong Wu[2] and Wei-Shi Zheng[1,3,4*]
[1]School of Data and Computer Science, Sun Yat-sen University, China
[2]School of Electronics and Information Technology, Sun Yat-sen University, China
[3]Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, China
[4]Collaborative Innovation Center of High Performance Computing, NUDT, China
chenzelinmail@gmail.com xKoven@gmail.com wuancong@mail2.sysu.edu.cn wszheng@ieee.org

Abstract— **Writer Identification aims to identify a certain writer from a given group of candidates by their handwriting. Although it is very significant in security systems like bank account verification systems, existing works focus on document-level or text-level writer identification. This limits their scalabilities and flexibilities in realistic scenarios as they require complete document or text. To facilitate the realistic applications of writer identification, we propose a novel technology, letter-level writer identification, which requires only a few letters as the identification cue. It is challenging due to large intra-class discrepancy and implicit identifiable writing cues. Considering these challenges, we propose a novel deep model called Multi-Branch Encoding net (Mul-BEnc). To evaluate our model and provide a benchmark for this problem, we have collected a large Letter-stroke sequence Writer identification DataBase (LetWriterDB). The experimental results validate the effectiveness of our model.**

Keywords: **letter-level writer identification, deep model, handwriting database, multi-branch**

I. INTRODUCTION

Writer identification aims to identify a certain writer from a given group of candidates by their handwriting. It is significant in the field of security systems, e.g., in criminal justice systems and bank account verification systems. It is a supplementary approach for biometrics recognition especially when other devices are not available. The popularity of electronic handwriting devices like smartphones promotes its development and makes it more important and meaningful in realistic scenarios.

In the literature of writer identification, existing works [7], [12], [16], [4], [19], [17], [22] focus on document-level or text-level problems. This limits their scalabilities and flexibilities in realistic applications as they require a full document or text. To further facilitate the applications of writer identification, we propose the letter-level writer identification, which only requires a few letters for identification. This problem setting is very meaningful in realistic scenarios because it largely promotes the practicability of writer identification. For example, in a smartphone, writing a few letters is much more applicable than writing a full document. As such, it promotes the popularity of writer identification, so that we can leverage this reliable biometrics for anyone with a touch-screen smartphone conveniently, rather than applying this biometric only in situations with specific devices.

However, letter-level writer identification is very challenging. This is because of the following two aspects. First, the identifiable writing cues are implicitly provided, as compared to the document-level or text-level problem setting. As compared to the document-level or text-level identification that consists of explicit cues about the language style information, the letter-level writer identification mainly implicitly contains identity information in the writing style. Hence, it is much challenging to extract the feature of implicit writing cues. Second, the intra-class discrepancy could be very large, as a certain writer may write a certain letter in different styles as shown in Figure 4.

In consideration of these challenges, we propose a novel deep model called Multi-Branch Encoding net (Mul-BEnc) for the letter-level writer identification problem. Considering that identifying a writer with only a few letters requires rather elaborate encoding, our model adopts a deep framework which takes advantages from both the Recurrent Neural Network (RNN) and the Convolutional Neural Network (CNN) to automatically learn an encoding feature. To address the intra-class discrepancy problem, we propose a novel multi-branch aggregated encoder with a maximum aggregation. The multi-branch encoder encodes any letter by multiple branches, and then aggregates the resulting encodings with maximum aggregation. Such structure can handle the intra-class discrepancy better, as different intra classes are treated in different branches that customize for the specific intra class. Furthermore, we suppose that the writings of the writers will be unreliable sometimes and some letters own their specific characteristic, which is shown in Figure 7 and Figure 6. So in order to enhance model's robustness, we propose a letter-specific aggregation to aggregate the letter-specific feature and increase the model's stability.

To provide a benchmark for this novel problem and evaluate our model, we collect a large Letter-stroke sequence Writer identification DataBase (LetWriterDB [11]). To our best knowledge, this is the first benchmark for the letter-level writer identification problem. It has over 30k samples from over 90 persons. We show some samples from the database in Figure 1.

Our main contributions are three-fold. (I) We propose a novel and meaningful problem, i.e., letter-level writer

*Corresponding author

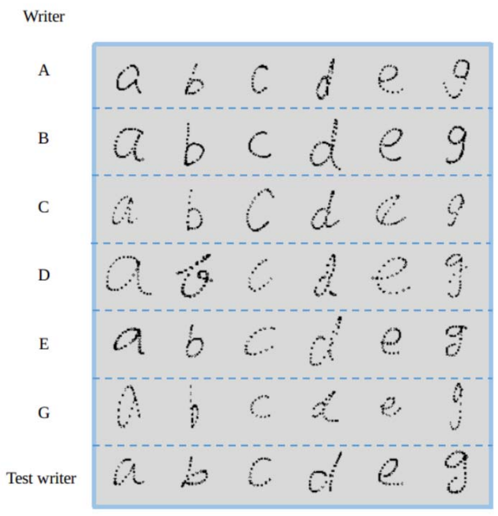

Fig. 1. Examples of LetWriterDB. LetWriterDB is the first benchmark for letter-level writer identification.

identification, which aims to identify a writer by only a few letters. This is significant in the development and deployment of writer identification. (II) We propose a novel deep model Mul-BEnc to address the letter-level writer identification problem. In our model, we propose a novel concept of multi-branch aggregated encoder, where each branch is customized for the specific writing style and propose a letter-specific aggregation to enhance the robustness of our model. (III) We provide a benchmark for the letter-level writer identification problem.

II. Related Work

There is a large amount of literature on writer identification dating back over 30 years and the accuracy of writer identification has achieved promising promotion in recent years.

Conventionally, researches exploit sophisticated handcrafted feature to model the handwriting. For example, the work [12] adopts a shape primitive approach, modeling the handwriting with temporal sequences and shape codes, then it classifies the writers based on diffusion-function [13]. Some works [4], [16] extract a codebook by clustering, using the code in the codebook to represent the handwriting. In [4], the dimensionality of representing a document is reduced through a trick of weighted summation of sparse code. Then the document is modeled with tf-idf feature based on the weights. Some works focus more on spatial characteristic rather than the global one to represent the special writer's handwriting. In [7], point-based feature is extracted first and the document is described by super feature based on the point-based feature above. All the works above achieve promising performances on document-level writer identification but they are still not satisfactory at text level. Although, in the latest work, neural network has been successfully applied for text-level writer identification [22], [23], handcrafted effort is also required. In [22], a handcrafted effort is needed at data augmentation phase called DropSegment, and data processing phase called path-signature feature maps. In [23], handcrafted effort like random hybrid strokes is required too.

Apparently, our model has some advantages compared to the existing methods. (I) Models above depend on handcrafted feature and require some domains knowledge of handwriting. In our model, no handcrafted effort and no domain knowledge of handwriting is needed because our model directly learns from raw data. (II) At letter level, most existing models cannot achieve a high accuracy. The accuracy decreases compared with text or document level. In our model, the performance is remarkable at letter level. The main reason is that both shallow models and deep models do not consider difference of the same writer's writing styles. We exploit a novel architecture namely multi-branch aggregated encoder to take such situation into consideration.

III. Multi-Branch Encoding Net

The traditional handcrafted models of text-independent writer identification can achieve pretty high performance at document level. Therefore we suppose that the hidden discriminative information of different strokes or letters that distinguishes the special writer can be extracted identically. In other words, most useful information for identifying the writer is independent of strokes' types, as writers own their special writing style independent of letters or words. Hence, we extract the feature of different letters with the same architecture, called multi-branch aggregated encoder.

However, when investigating the writing samples, we can find that the same writer writes the same letter differently, shown in Figure 4. Obviously, such different writings should be treated variously because of the various writing styles and font types. Namely, the distribution of the same letter's input is multimodal and there is a large inter-class discrepancy. Hence, in the multi-branch aggregated encoder, we use unshared CNN-RNN (CRNN) encoders and a maximum aggregation to fit such multimodal function and learn a more discriminative feature.

Furthermore, as shown in Figure 7 not all extracted feature are reliable and as shown in Figure 6, it is significant to extract the letter-specific characteristic. Hence, to enlarge the stability and extract the letter-specific feature, we propose a letter-specific aggregation to aggregate the encodings of each letter.

A. Model Architecture

The Mul-BEnc network takes six sequences of letters (a, b, c, d, e, g) as input. The network first processes each letter with the same multi-branch aggregated encoder. In the multi-branch aggregated encoder, the letter would be encoded by several unshared CRNN encoders and the several outputs of different CRNN encoders would be aggregated by maximum aggregation as a feature map. Then the six feature maps would be aggregated by the letter-specific aggregation (max aggregation). Finally, the Fully Connect layer (FC)

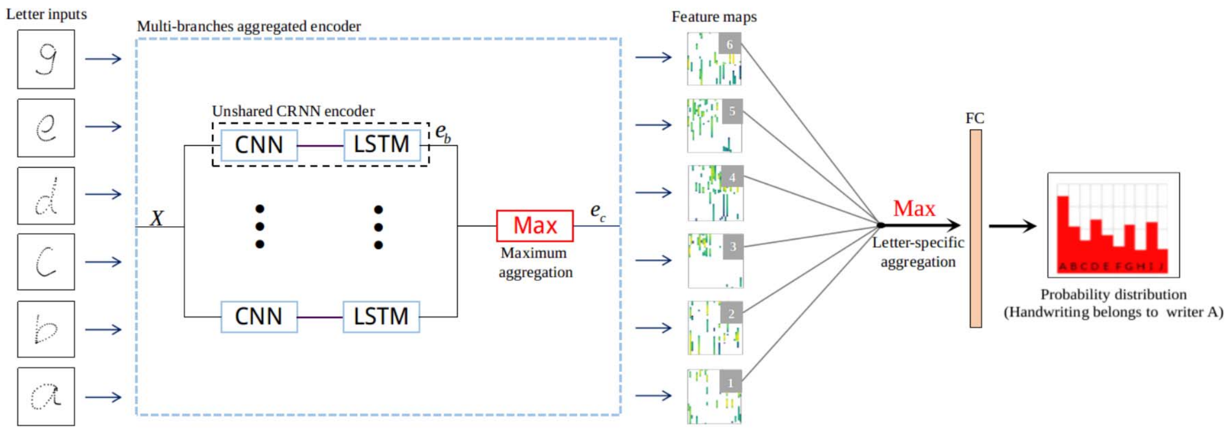
Letter inputs
Multi-branches aggregated encoder
Unshared CRNN encoder
CNN
LSTM
e_b
X
Max
Maximum aggregation
e_c
Feature maps
Max
Letter-specific aggregation
FC
Probability distribution (Handwriting belongs to writer A)

Fig. 2. The illustration of Mul-BEnc Architecture. The input consists of six sequences of letters (a, b, c, d, e, g). Each letter is encoded into a feature map by multi-branch aggregated encoder. The multi-branch aggregated encoder is made up of several unshared CRNN encoders and the maximum aggregation, which can compute the representative feature maps of each letter. Then the letter-specific aggregation (max) aggregates the six feature maps, which can extract the letter-specific feature and increase the stability of the system. Finally, the Fully Connect layer (FC) takes the letter-specific aggregated feature map as input and outputs the probability distribution of each writer.

takes the letter-specific aggregation as input and outputs the probability distribution of each writer.

More detailedly, The multi-branch aggregated encoder is a function $f_e : \mathbb{R}^{T\times P} \to \mathbb{R}^{T\times H}$, where T is the total timesteps of the writing letter trajectory, P is the input dimension of each timestep and H is the output dimension of each timestep. The CRNN encoder is a function $f_b : \mathbb{R}^{T\times P} \to \mathbb{R}^{T\times H}$. In multi-branch aggregated encoder, each letter would be encoded by N CRNN encoders into N encodings (1).

$$[\mathbf{e}_{b,1},\ \cdots,\ \mathbf{e}_{b,N}] = [f_{b,1}(\mathbf{X}),\ \cdots,\ f_{b,N}(\mathbf{X})]. \tag{1}$$

Then it combines the N branches with an aggregated function f_a that aggregates N CRNN encoded results (2).

$$\mathbf{e}_c = f_e(\mathbf{X}) = f_a([\mathbf{e}_{b,1},\ \cdots,\ \mathbf{e}_{b,N}]). \tag{2}$$

Finally, after encoding each input letter $\mathbf{X}$ into encoding $\mathbf{e}_c$, the letter-specific aggregation in Mul-BEnc aggregates all encodings of each letter $[\mathbf{e}_{c,1},\ \cdots,\ \mathbf{e}_{c,L}]$ into a feature $\mathbf{E}$, where L is the number of letters. Then we feed the feature into the classifier. The classifier distinguishes the writers based on the predicted probability distribution. In short, the Mul-BEnc architecture is shown Figure 2. In the following subsection, we will introduce CRNN encoder, multi-branch aggregated encoders and letter-specific aggregation in detail.

B. CRNN Encoder

As the architecture of combining recurrent neural networks and convolutional neural networks achieved promising results both in computer vision as well as natural language processing [3], [18]. So, we leverage the similar architecture in the CRNN encoder, stacking the *LSTM* [9] layer upon the convolutional layer, which is shown in Figure 3. We use convolutional layer to extract the spatial information from the strokes and apply *LSTM* unit to store the strokes' global information.In the following, we will describe the effects and architecture of convolutional layer and *LSTM* layer respectively.

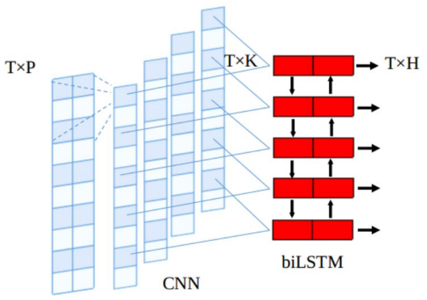

Fig. 3. The architecture of CRNN encoder, the substructure of multi-branch aggregated encoder.

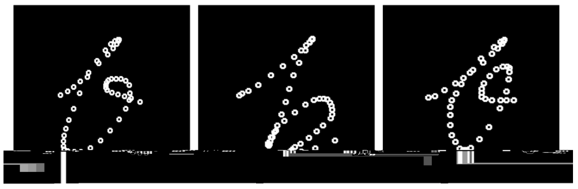

Fig. 4. The writer writes letter 'b' differently. It shows that the intra-class discrepancy is large.

1) Convolutional Layer Learns Higher Level Feature: Intuitively and traditionally, in the researches of trajectory recognition, the velocity, the accelerated velocity, the angular velocity and the accelerated angular velocity are often aggregated together to generate a new feature [20], [14], [12]. In the following description, we show that convolutional layer can learn the features above. Firstly, we illustrate the computation of convolutional layer and then explain that the transformation from absolute position to the velocity has resemblances to the computation in the 1 dimensional convolution.

Following the convolutional layer architecture in [6], we give an example of a kernel computation output to illustrate its resemblances to the velocity. We define $\mathbf{O} = Conv1D(\mathbf{W},\mathbf{X})$, where $\mathbf{O} \in \mathbb{R}^{T-S+1}$, $\mathbf{W} \in \mathbb{R}^{S\times P}$, $\mathbf{X} \in \mathbb{R}^{T\times P}$, T is the total timesteps of the sequence, P is the input dimension of each timestep, S is the kernel size. The output of $Conv1D$ is (3).

$$o_k = \sum_{i=1}^{S}\sum_{j=1}^{P} w_{i,j}x_{i+k-1,j},\ k \in \{1,\cdots,T-S+1\}. \tag{3}$$

To compare the computation between the velocity and $Conv1D$, we assume that $P = 1$, $S = 2$. The velocity computation is (4).

$$v_t = 1\times x_{t+1,1} - 1\times x_{t,1},\ t \in \{1,2,\cdots,T-1\}. \tag{4}$$

If we let $w_{1,1} = 1$ and $w_{2,1} = -1$, then $Conv1D$ (3) is the same as the velocity (4). Similarly, if we let $P = 1$, $S = 3$, $w_{1,1} = 1$, $w_{2,1} = -2$, $w_{3,1} = 1$ then $Conv1D$ (3) is the same as the accelerated velocity (5).

$$a_t = 1\times x_{t-1,1} - 2\times x_{t,1} + 1\times x_{t+1,1}. \tag{5}$$

Namely, the model capacity of convolutional layer is larger.

The analysis above is the reason why we apply convolutional layer to extract higher level feature rather than leveraging the handcrafted feature or raw data feeding the *LSTM* layer. Besides, when identifying a writers, the convolutional computation is good at extracting partial feature which provides a lot of discriminative information.

2) LSTM: Recurrent neural network has been extensively used and achieved remarkable result in the area of processing sequences.

One of the popular variation of recurrent neural network is *LSTM*. One of the advantages of *LSTM* is that it can store long-distance history information, handling the information based on the state. In our work, we leverage all outputs of each timestep to extract global distinguished information. Following the *LSTM* unit in [9], we define a function $[s_t,\ y_t] = LSTM(x_t,\ s_{t-1},\ y_{t-1})$ (6).

$$\begin{aligned}
f_t &= \sigma(\mathbf{W_f}\cdot[y_{t-1}^T, x_t^T]^T + b_f),\\
i_t &= \sigma(\mathbf{W_i}\cdot[y_{t-1}^T, x_t^T]^T + b_i),\\
\tilde{s}_t &= \mathbf{W_s}\cdot[y_{t-1}^T, x_t^T]^T + b_s,\\
s_t &= f_t * s_{t-1} + i_t * \sigma(\tilde{s}_t),\\
o_t &= \sigma(\mathbf{W_o}\cdot[y_{t-1}^T, x_t^T]^T + b_o),\\
y_t &= o_t \odot tanh(s_t).
\end{aligned} \tag{6}$$

Where $\mathbf{W}_* \in \mathbb{R}^{H\times(H+K)}$ and $b_* \in \mathbb{R}^H$. K is the kernel number in the convolutional layer. In specific time step t, $x_t \in \mathbb{R}^K$ is the input, $s_t \in \mathbb{R}^H$ is cell state, $y_t \in \mathbb{R}^H$ is the unit output, f_t is the output of unit forget gate, i_t is the output of unit input gate and o_t is the output of unit output gate. σ is the sigmoid function, $\odot$ means element-wise product.

Single direction recurrent network suffers a weakness of not taking advantage of future information. Bidirectional recurrent network [15] utilizes both the previous and future feature by processing the sequence on two directions, and then concatenates the two independent sequences, i.e., $y_t = \overrightarrow{y_t}||\overleftarrow{y_t}$. As a result, we use bidirectional *LSTM* (biLSTM). Finally, we concatenate the output of each time step, generating the encoding $\mathbf{e}_c = [\overrightarrow{y_1},\ \cdots,\overrightarrow{y_T},\ \overleftarrow{y_1},\ \cdots,\ \overleftarrow{y_T}]$.

C. Multi-Branch Aggregated Encoder

We propose a novel idea of modeling handwriting, namely multi-branch aggregated encoder. There are two components in the multi-branch aggregated encoder, the multiple branches of CRNN encoder and the maximum aggregation. In the following description, we will introduce the multiple branches of CRNN encoder and the maximum aggregation respectively.

1) Multi-branch CRNN Encoders: In reality, a writer can write the same letter differently. For example, the writer would write the letter sometime squiggly and sometime very formally, or the writer would write the letter with different font types (e.g. sometime cursive type and sometime italic type). Figure 4 shows the different writing styles of the same writer. In brief, it needs various encoders to handle various situations as the large intra-class discrepancy. Hence, treating the different situations with multi-branch encoder can alleviate the influence of the diversity of different writing styles. The unshared CRNN encoders own their weights respectively. The different branches of CRNN encoder are customized for different writing styles.

2) Maximum Aggregation: As analyzing at the beginning of section III, each letter would be encoded through N branches of CRNN encoder (1) and would be aggregated in the shared multi-branch aggregated encoder by f_a (2). We describe the aggregated function following.

The same letter has been encoded by N branches in order to alleviate the influence of the diversity of different writing styles. Furthermore, it is significant to find an effective way to aggregate the encodings. It can represents the writing better. Inspired by aggregation in [21] and maxout activation in [8], we aggregate the $[\mathbf{e}_{b,1},\cdots,\mathbf{e}_{b,N}]$ with a maximum function. As our problem is a multimodal problem, the inputs

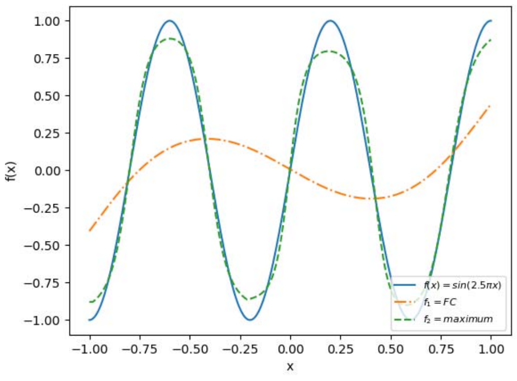

Fig. 5. Figure shows the trained results of networks with a few parameters. The network f_1 contains 2 hidden layers, 4 hidden units in each layer. The network f_2 is the maximum network of two sub dense networks. Each subnetwork contains 2 hidden layers, 2 hidden units each layer. Apparently, maximum network performs better.

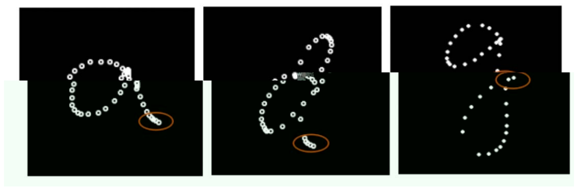

Fig. 6. Letter 'g' owns the specific characteristic different from other letters (surrounding by red circle). The stroke's tail writing from bottom left to upper right only appears in letter 'g'. But the stroke's tail writing from upper left to bottom right appears in both letter 'a' and letter 'd'.

maximum network performs better.

In short, we adopt the maximum aggregation for $[\mathbf{e}_{b,1}, \cdots, \mathbf{e}_{b,N}]$ (7).

$$f_a(x) = maximum([f_{b,1}(\mathbf{X}),\ \cdots,\ f_{b,N}(\mathbf{X})]). \quad (7)$$

D. Letter-Specific Aggregation

There are two effects of letter-specific aggregation, (I) extracting letter-specific characteristics and (II) increasing the stability of the model. The maximum function can achieve both effects. In the following description, we will explain the details of the effects above and how the maximum function achieves the both effects.

(I) Some letters own specific characteristic as shown in Figure 6. In the multi-branch aggregated encoder, all strokes are encoded in an identical way. So the same place of different letter's feature maps absolutely shares the same meaning. It means that the specific characteristic is corresponding to the specific position in the feature maps. The value of such position should be much larger than others if the letter owns the specific characteristic while other letters not. Apparently, the maximum function can extract such characteristic among the feature maps.

(II) Besides extracting the letter-specific characteristics, maximum function can increase the model's stability. We

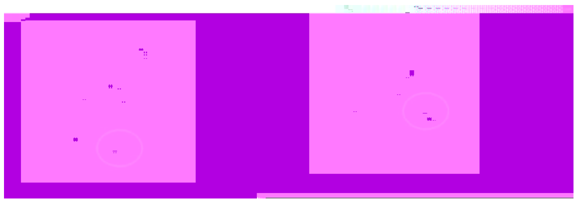

Fig. 7. Letter 'a' and 'd' share the similar writing at the tail of stroke (surrounding by red circle), but currently the writer writes letter 'd' squiggly so that the special characteristic of letter 'd' may be unreliable.

suppose that not all characteristics in the feature maps are reliable. For example in intuition, both letter 'a' and 'd' share the similar writing at the tail of stroke, but which is shown in Figure 7 (surrounding by red circle) is that the writer writes the tail of letter 'd' squiggly. In other words, the characteristic of the stroke's tail of letter 'd' is unreliable currently. It is probably unstable to fuse the stroke's tail characteristic of such writing of letter 'd'. Therefore, if letter 'a' and letter 'd' stroke's tail characteristic is corresponding to the specific position of the feature maps, we greedily select largest one among the L feature maps as it is the most reliable. In brief, larger the value is, more reliable the characteristic is. Hence the maximum can extract the most informative and reliable value in the specific position that contributes most to identify the writer and increase the model's stability by filtering the less reliable or unreliable response.

In brief, we fuse the $[\mathbf{e}_{c,1}, \cdots, \mathbf{e}_{c,L}]$ into $\mathbf{E}$ with a maximum function (8).

$$E_{i,j} = \max_{l \in \{1,\cdots,L\}} e_{c,l,i,j}. \quad (8)$$

Then we feed the output into a fully connect layer followed by a logistic regression layer with softmax activation to predict the probability of each writer.

IV. LetWriterDB Database

A. Comparison with Other Databases

In our model, we need the labels' of letters, but it lacks database to investigate. Though there are many databases like [5], [1], [10] can be applied to identify the writers, these databases are usually applied to text-level or document-level writer identification. It is not suitable for letter-level

type	a	b	c	d	e	g	total
person#	91	85	83	81	77	77	91
letter#	5977	5050	4962	4700	4682	4723	30094

TABLE I
STATISTICS OF LETWRITERDB

	letter#	person#
training set	20687	60
test set	8867	60

TABLE II
THE SPLIT OF TRAINING SET AND TEST SET

writer identification. These databases only give some labels for types of texts or documents but lack the labels of letter's types, while the labels of letter's types are significant in the challenge. Furthermore, it is a challenging work to label the letters as most ligatured words exist in the existing databases.

B. Details of the Database

Therefore, to solve the challenge, we propose a large letter-stroke sequences writer identification database, LetWriterDB. Some examples of LetWriterDB are shown in Figure 1. LetWriterDB contains a large number of online letters collecting from smartphones, where online means the handwriting are described by sequence. We sample the online handwriting in mobile phones by a website, which is very convenient for writers to offer their handwriting. While collecting the online handwriting, we choose the types of letters which can be written in just only one stroke. The writers are asked to write six letters (a, b, c, d, e, g) respectively. The reason of choosing the single-stroke letters is following. The problem that we study is identifying writers by requiring them to write several pre-specified letters. For convenience, the number of letters and strokes should be as few as possible. Table I shows some statistics of our database.

C. Evaluate Protocol

We randomly split the data into training set and test set with the ratio 2:1 which is shown in Table II. We choose the writers whose letter's number is in the top 60. The reason is that only the top-60 writers providing all required letters are used, while the remaining writers lacking letters (e.g. e, g) are invalid under our setting. We offer the other samples as they can be applied in other situation like when we only ask the writer to offer a, b, and c letters.

V. Experiments

To evaluate the effectiveness of our model, we do some experiments on the LetWriterDB and IAM [10]. We modified IAM for letter-level writer identification by selecting six kinds of letters from texts, including both single-stroke letters (e, h, y) and multi-stroke letters (f, i, t). There are 90 writers in the IAM database after our modification.

A. Compared with State-of-the-Art Methods

We compare our model with other state-of-the-art methods including two shallow models and two deep models.

*a) **Point Based** [7]:* The shallow model describes the stroke directly using N point-based feature like writing direction angle and curvature angle. Besides the partial information, a little global information from whole stroke is extracted too, like stroke speed and word-based feature.

*b) **Histogram Based** [4]:* In the model, the document is modeled with a weighted summation of K cardinalities of sparse coding. The sparse coding are learned from the all training strokes from the whole database. After then a tf-idf feature is extracted, using the weights information. In addition, each extracted stroke in [4] is described by histograms.

	LetWriterDB	IAM
Point Based [7]	19.97%	67.99%
Histogram Based [4]	11.23%	55.96%
DeepWriterID [22]	35.19%(±0.1%)	49.35%(±0.1%)
DeepRNN Based [23]	7.53%(±0.1%)	7.83%(±0.1%)
Ours	**0.29**%(±0.1%)	**2.97**%(±0.1%)

TABLE III

Comparison with Other State-of-the-Art Methods on LetWriterDB and IAM Database.

	Top-1 Test Error
Average	0.66%(±0.1%)
Weighted Sum	0.73%(±0.1%)
Maximum	**0.29**%(±0.1%)

TABLE IV

Effect of Aggregation in Multi-Branch Aggregated Encoder

*c) **DeepWriterID** [22]:* The paper propose a data augmentation method called DropSegment. The letter's online path is extracted into a path-signature [2] feature maps. Then the feature maps are fed into convolutional neural network.

*d) **DeepRNN Based** [23]:* The paper propose a simple RHS data augmentation method. The letter's online path is fed into bidirectional recurrent neural network.

The top-1 test errors on different baselines are shown in Table III.

1) Comparison with Hand-crafted Models: The performance of our model is better than traditional hand-crafted feature-based models. It illustrates that in the aspect of writer identification, especially at letter level, feature engineering is not guaranteed to be an optimal solution. In other words, our model with deep-learning-based method can learn more discriminative feature from several letters and such method is more suitable for the problem we proposed. More appealingly, our model directly learns from raw data and extracts better representation of handwriting, which requires little domain knowledge of handwriting.

2) Comparison with Other Deep Models: Our model outperforms the other deep models as we propose some novel structures in our model. We take the intra-class discrepancy into consideration by multi-branch aggregated encoder while other deep models not. Besides, we aggregate the letter's feature maps by letter-specific aggregation while others only combine the results of probability distributions.

B. Effects of Different Components in Mul-BEnc

The following evaluations of effects of different components in Mul-BEnc are based on LetwriterDB.

1) Effect of Aggregation in Multi-Branch Aggregated Encoder: To evaluate the effect of the maximum aggregation in multi-branch aggregated encoder, we replace the maximum aggregation with other conventional functions. In Table IV, it is shown that maximum aggregation achieves the best performance. This shows that by applying the maximum aggregation, the multi-branch aggregated encoder can learn better representation.

	Top-1 Test Error
Average	0.37%(±0.1%)
Weighted Sum	0.82%(±0.1%)
Maximum	**0.29**%(±0.1%)

TABLE V

EFFECT OF LETTER-SPECIFIC AGGREGATION

	RNN	RNN stacks on CNN
$N = 1$	0.89(±0.1%)	0.79%(±0.1%)
$N = 2$	0.66(±0.1%)	0.49%(±0.1%)
$N = 3$	0.53%(±0.1%)	0.29%(±0.1%)

TABLE VI

THE EFFECT OF CRNN

2) *Effect of Letter-Specific Aggregation:* To evaluate the effect of the maximum aggregation in letter-specific aggregation, we also replace the maximum function with other conventional functions. In Table V, it is shown that maximum aggregation achieves the best performance too.

3) *Effect of Branch Number N:* For evaluating the exact impact of branches of CRNN encoders, we compare the performance of our model with various N. In Figure 8, it is shown that the model with a single CRNN encoder gains much higher error rate. It is likely that the representation of different writing styles should be customized by different branches. However, as the different written styles of the same letter is finite, applying overmuch branches cannot gain better performance. Adversely, such complex networks may be hard to be trained as shown by model $N = 5$ and $N = 6$. To be specific, the similar test errors between $N = 3$ and $N = 4$ show that when the number of branches is enough, larger N cannot achieve better performance, while the network may be hard to train.

4) *Effect of Convolutional Layer:* We evaluate the effect of higher level extraction learned by convolutional layer by comparing the performance of Mul-BEnc and Mul-BEnc (no CNN). In Table VI, errors decrease by employing convolutional layer before recurrent neural network. It is evident that feeding higher level feature into recurrent neural network can achieve better performance.

C. *Performances of Using Fewer Letters*

Recalled the motivation of our work, we wish to make the writer identification easier to be deployed. In our model, we can further cut down number of letter's types while maintaining the performance. To evaluate the performance of our model at the situation of fewer letters, we randomly abandon one to three letter's types and train the model again. We repeat the same situation six times. For example, for the situation of abandoning one letter's type, we do six experiments without letter 'a', 'b', 'c', 'd', 'e' and 'g' respectively. Table VII shows the mean top-1 accuracy of various number of letter's types. Such results are competitive among the state-of-the-art methods. In short, with fewer letters, our model can still identify writers with remarkable performance. The practicability of the model is high.

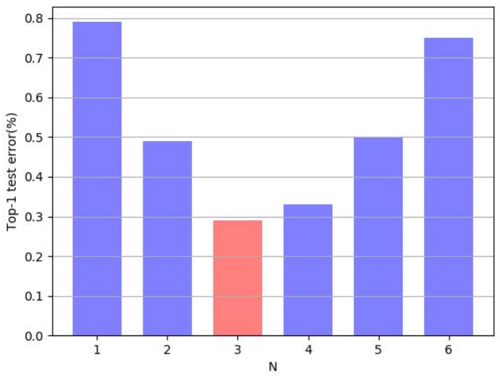

Fig. 8. Performances of Various Branch Number N.

letter type#	6	5	4	3
mean top-1 accuracy	99.71%	99.51%	98.75%	98.23%

TABLE VII

MEAN TOP-1 ACCURACIES OF COMBINING DIFFERENT LETTER'S TYPES

VI. CONCLUSION

The meaningful problem we address, letter-level writer identification, aims to identify a writer by only several letters he/she writes. There are two main challenges, the intra-class discrepancy and implicit identifiable writing cues. To solve these challenges, we provide a benchmark LetWriterDB as the inceptive work and propose a state-of-the-art model called Mul-BEnc with accuracy 99.71% on the LetWriterDB. In Mul-Enc, not only do we propose a novel multi-branch aggregated encoder to alleviate the influence of large intra-class discrepancy but also propose letter-specific aggregation to aggregate the letter-specific characteristic and increase the model's stability, enhancing the robustness. At last, we show that our model is useful in reality as the performance does not decrease a lot with fewer letters.

ACKNOWLEDGEMENT

This work was supported partially by the NSFC(61522115, 61472456, 61661130157,61628212), the Royal Society Newton Advanced Fellowship (NA150459), Guangdong Province Science and Technology Innovation Leading Talents (2016TX03X157).

REFERENCES

[1] CASIA Handwriting Database. `http://biometrics.idealtest.org`.

[2] K.-T. Chen. Integration of paths–a faithful representation of paths by noncommutative formal power series. *Transactions of the American Mathematical Society*, 89(2):395–407, 1958.

[3] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.

[4] I. Dwivedi, S. Gupta, V. Venugopal, and S. Sundaram. Online writer identification using sparse coding and histogram based descriptors. In *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*, pages 572–577. IEEE, 2016.

[5] H. El Abed, V. Märgner, M. Kherallah, and A. M. Alimi. Icdar 2009 online arabic handwriting recognition competition. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 1388–1392. IEEE, 2009.

[6] M. Feng, B. Xiang, M. R. Glass, L. Wang, and B. Zhou. Applying deep learning to answer selection: A study and an open task. In *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*, pages 813–820. IEEE, 2015.

[7] M. Gargouri, S. Kanoun, and J.-M. Ogier. Text-independent writer identification on online arabic handwriting. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 428–432. IEEE, 2013.

[8] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. *arXiv preprint arXiv:1302.4389*, 2013.

[9] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[10] IAM On-Line Handwriting Database. `http://www.fki.inf.unibe.ch/databases`.

[11] Letter-stroke Sequence Writer Identification DataBase. `https://www.dropbox.com/s/k94cma7iabaugvs/gesture.db`.

[12] B. Li and T. Tan. Online text-independent writer identification based on temporal sequence and shape codes. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 931–935. IEEE, 2009.

[13] H. Ling and K. Okada. Diffusion distance for histogram comparison. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 246–253. IEEE, 2006.

[14] A. Namboodiri and S. Gupta. Text independent writer identification from online handwriting. In *Tenth International Workshop on Frontiers in Handwriting Recognition*. Suvisoft, 2006.

[15] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[16] G. Singh and S. Sundaram. A subtractive clustering scheme for text-independent online writer identification. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 311–315. IEEE, 2015.

[17] G. X. Tan, C. Viard-Gaudin, and A. C. Kot. Automatic writer identification framework for online handwritten documents using character prototypes. *Pattern Recognition*, 42(12):3313–3323, 2009.

[18] M. Tan, C. d. Santos, B. Xiang, and B. Zhou. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*, 2015.

[19] M.-Y. Tsai and L.-S. Lan. Online writer identification using the point distribution model. In *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, volume 2, pages 1264–1268. IEEE, 2005.

[20] X. Wang, M. Xia, H. Cai, Y. Gao, and C. Cattani. Hidden-markov-models-based dynamic hand gesture recognition. *Mathematical Problems in Engineering*, 2012, 2012.

[21] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. *arXiv preprint arXiv:1611.05431*, 2016.

[22] W. Yang, L. Jin, and M. Liu. Deepwriterid: an end-to-end online text-independent writer identification system. *IEEE Intelligent Systems*, 31(2):45–53, 2016.

[23] X.-Y. Zhang, G.-S. Xie, C.-L. Liu, and Y. Bengio. End-to-end online writer identification with recurrent neural network. *IEEE Transactions on Human-Machine Systems*, 47(2):285–292, 2017.