Online Writer Identification

A thesis report submitted for EE 499 Project II

by

Sounak Ray

(Roll No : 160102086)

Under the guidance of

Dr. Suresh Sundaram

Department of Electronics and Electrical Engineering

Indian Institute of Technology Guwahati

June 2020

Abstract

Writer identification is an important behavioural biometric. Technological advances has made it possible for people to obtain temporal handwritten trace through hand held devices like tablets. The large amount of information about the writer that can be obtained from the tablet has to be properly leveraged to develop novel writer identification techniques. The dynamic information obtained from the tablet helps us to model both the temporal and the spatial information. Systems that make use of such textual content for identification of an individual are termed 'online'. In particular, the input is a set of strokes, each of which consists of a sequence of points. In this thesis, we aim to develop novel descriptors for text-independent writer identification by leveraging the pressure information obtained from the tip of the stylus. We feel that pressure is the best metric to characterize a writer as opposed to text-dependent features like slope of a stroke or angle of the stroke.

In this thesis, we represent the spatio-temporal trace from a document using a descriptor which is inspired by the Pattern of Local Gravitational Force (PLGF). To begin with, we modify the Pattern of Local Gravitational Force (PLGF) approach, popular in the domain of face recognition, to online writer identification. The Pattern of Local Gravitational Force is modified in such a way that we obtain discriminative descriptors for each sub-stroke. The sub-stroke is then considered as a basic temporal unit of a writer's trace and we use a certain number of sequential sub-strokes to train an end-to-end sequence model using Long Short Term Memory Units (LSTM Units) as the basic building block.

We also explore a novel framework using histogram-based features and a supervised LSTM autoencoder. Our strategy strives to leverage the spatial and temporal information in an online handwritten trace. The spatial descriptors of a sub-stroke are based on histograms that are generated by incorporating the pressure and velocity information encapsulated in the sub-strokes of a writer. By considering a sub-stroke as a basic temporal unit of a writer's trace, we explore the efficacy of a Supervised Long Short Term Memory (LSTM) autoencoder to com-

press the sequence data. The compressed representation is then used to train a Support Vector Machine (SVM) to obtain the final authorship of the handwritten text. The efficacy of our proposed approach is tested on the BIT CASIA Database. We achieve state-of-the art results at the text-line level.

Contents

List of Tables

List of Figures

Nomenclature

PLGF	Pattern of Local Gravitational Force
RNN	Recurrent Neural Network
LSTM	Long Short Term Memory
SVM	Support Vector Machine

Chapter 1

Introduction

Writer Identification techniques are assigned with the task of identifying the author of a handwritten document given that there are numerous candidate authors to choose from. The field of writer identification has grown immensely over the last few decades due to the introduction of machine learning and pattern recognition. The need for development of person identification systems using various biometric traits to strengthen the security associated with various digital accounts of a person has also led to the sudden interest in writer identification techniques. Writer identification has widespread applications in areas such as criminology decision systems and banking sector as it can also remove subjective evaluation, arising due to human intervention, from the process of writer identification and thus can be used in various criminal or civil court cases. However, as handwriting is a behavioral biometric it has high variance when obtained from the same person at different instances of time. Physiological biometrics like iris, fingerprint, face etc. are traditionally less prone to such intra-class variances. This makes the task of writer identification all the more challenging.

Writer identification systems are predominantly of two types, online and offline. Online systems necessarily use a temporal trace or a temporal sequence of x and y co-ordinates to assign authorship of a document. Online data collection methods may also contain additional information like pressure, azimuth, angle, pen up and pen down statuses. This additional information available to online systems usually make them more accurate as compared to offline systems. Offline systems use scanned images of documents to identify the writer. They do not have access to any temporal information as compared to online systems. Each method has it's own set of challenges, online systems usually have greater intra-writer variance, whereas offline documents do not have the temporal data. Online systems are mainly used in access

control applications like in the banking sector.

Writer identification systems can be further segregated into two categories, text-dependent and text-independent depending on the textual content they are used to classify. In the text-dependent approach, the system has knowledge about the content of the data. However, such systems fail in scenarios where text documents containing different text content need to be contrasted. Text-independent techniques, on the other hand, focus on capturing the writing style of a particular writer irrespective of the textual content.

Handwriting possesses very less discriminating power and hence it is considered as a weak biometric. It could be widely used for low security access control applications like accessing someone's personal computer. Writer identification could also be used in two different settings, that is can be used for verification or identification. Verification is usually to check the authenticity of a writing sample, that is, if it belongs to particular person or not. It is usually a binary classification problem wherein the classifier outputs a yes or a no as the answer. Identification however requires that the system assigns the correct label to a writing sample by comparing it with existing samples in the database. Identification tasks are usually more challenging than verification as they require the system to choose among numerous candidate classes.

In this thesis, we aim to develop a text-independent online writer identification system by taking the pressure of the stylus as the characteristic of a person's handwriting. A sub-stroke is considered as basic temporal structure of a handwriting trace and the whole handwritten document is thought to consist of numerous such sequential sub-strokes. We thus aim to capture the spatio-temporal characteristics of a handwritten document by first defining novel spatial descriptors using the common concepts of physics like gravitational force and momentum. The gravitational force and momentum are computed by taking the pressure reading at a sample point as the value of the mass of a point object at that point. Then we capture the temporal information in the sequence of sub-strokes using a sequence modelling algorithm called the Long Short Term Memory (LSTM). We also explore the efficacy histogram-based sub-stroke descriptors which are developed using the concepts of gravitational force and momentum. The proposed descriptors are very intuitive and can be easily computed using the concepts of physics. The temporal information is subsequently modelled using an LSTM autoencoder and the compressed representation thus learnt by the autoencoder is classified using an SVM to establish the final authorship of the document. The experiments have demonstrated that the use of the Supervised LSTM Autoencoder instead of a fully connected layer have helped us in gaining

significant performance improvement as we do not have to train a large number of parameters in the final softmax layer for a complex classification problem. The thesis is organized in the following manner. In Chapter 2, we present a review of the existing literature in the domain of online writer identification and how this has motivated our thinking. In Chapter 3, we present the proposed methodology followed in our writer identification system. In Chapter 4, we discuss the quantitative results of our proposed system and the implementation details. In Chapter 5, we conclude the thesis and present the future scope of our work.

Chapter 2

Related Work

Various applications of writer identification in areas such as forensic science has attracted many researchers towards this exciting field. With the introduction of new electronic devices for acquiring the data, rapid changes are taking place in the methodologies followed for developing such identification systems. Previously, people used to make authentication systems using signature verification systems. However, the main advantage of writer identification over signature identification is the abundance of data for developing machine learning systems from it. Writing samples usually contain few lines of written text per sample. Hence, the number of data points available to us is also substantial. Also, signature verification is a sub-class of writer identification wherein we use a specific trait of person's handwriting, i.e., signature. In this chapter we will present the existing works in the literature of writer identification and how that motivated our idea of using Pattern of Local Gravitational Force (PLGF) which had previously been successfully used for face recognition.

The existing works in the field of online writer identification usually aim to extract features from the spatio-temporal sequence and then classify it using traditional machine learning algorithms like a Support Vector Machine or a Gaussian Mixture Model.

One of the most prominent work in the field of online writer identification was by Schlapbach *et. al.* [4] in the domain of writer identification for smart meeting room applications. This method involved the usage of a Gaussian Mixture Model based approach, wherein they used a GMM-UBM model. They first constructed a Universal Background Model (UBM) with the help of the training data and the Expectation Minimization Algorithm. Thereafter they obtain individual GMMs for each candidate writer. People have also used approaches motivated by the information retrieval domain. Tan *et. al.* [5] segmented the document at the character level.

They then generated different prototypes of the characters using k-means clustering algorithm. They then used the concept of term frequencies (tf) and inverse document frequencies (idf) from the field of information retrieval to generate distributions on alphabet level. The concept of term of term term frequencies (tf) and inverse document frequencies (idf) were also used by Singh *et. al.* [16]. The authors of this paper made use of the subtractive clustering technique to capture the most frequently occuring writing styles. Thereafter, a tf-idf framework was used for recognition. In [18], the Latent Dirichlet Allocation is proposed to describe the handwritten content of the writers. Researchers have also explored the use of beta-elliptic models [19] to represent the velocity and spatial profiles of sub-strokes

People have also experimented with the concept of applying basic geometric shapes analysis to online writer identification. Li *et. al.* [12] proposed a two stage hierarchical system, wherein they use orientation and pressure analysis to reduce the set of candidate classes. In the second stage, they employ more complex primitives like curvature to perform writer identification. People have also tried to model the temporal information using the temporal sequence modelling and shape coding techniques. Sun *et. al.* [13] employed two types of temporal codes, the stroke temporal sequence code and neighbour temporal sequence code to uniquely encode the speed and pressure respectively. The shape codes proposed by them are used to store information about the direction of the trace.

Document feature based approaches try to develop a universal descriptor for a document. Thumwarin *et. al.* [1] use a barycenter based technique wherein they estimate the velocity of the barycenter. The barycenter is determined using a point of the script and two adjacent pen-point positions at any instant of processing. Tsai *et. al.* present a document descriptor based on the point distribution model. This provides them with a powerful method of modelling the variations in the shape of handwritten trace of a particular writer. Kameya *et. al.* [3] presented a novel writer identification method based on Continuous Dynamic Programming method. A person, who is registered, draws a reference figure which is composed of many parts and the person could be identified with any arbitrarily small region of the figure. Venugopal *et. al.* [11] proposed a codebook-based paragraph descriptor, wherein the codebooks are learnt using sparse coding technique. In recent works, explorations in the direction of sparse representation and code-book based descriptors [11, 15, 20–22] have been utilized to learn representations of a hand-written document

Handwritten characters and words possess a lot of discriminative information, as we could

characterize the writing style of a person using spacing between characters or the connections between various characters in a word. This has already been exploited manually by handwriting experts for a long time. Nakamura *et. al.* [6] proposed a method to extract certain characteristics of characters written in Kanji script by using the character and the spacing between characters. The individual characteristics of a character are then used to determine a set of feature parameters on the basis of shape and writing behavior. The statistical test, like one-way Analysis of Variance (ANOVA), show that the features are discriminative enough to identify and verify a writer.

With the introduction of deep learning techniques, people have now moved from the traditional paradigm of using hand-crafted features to using more deep learning based features. Zhang *et. al.* [7] proposed a recurrent neural network based system. In this system the authors use simple features like the difference between x and y co-ordinates of successive points and then randomly sample data points to represent the handwriting using a Random Hybrid Stroke (RHS). A Long Short Term Memory (LSTM) model is then used to classify these Random Hybrid Strokes. Numerous Convolutional Neural Network based methods also exist in literature.

Most of the existing methods of hand-crafted feature extraction are motivated by existing works from the domain of image recognition, information retrieval, face recognition and speech recognition. Many novel image descriptors exist in literature. Bhattacharjee *et. al.* [8] proposed the Pattern of Local Gravitational Force (PLGF) based descriptor which is inspired by Newton's Laws of Gravitation. The authors of this paper model the pixel intensities as bodies with mass who attract other bodies in it's vicinity using the equation for gravitational force. The gravitational force and angle are then used to construct image descriptors using histogram based descriptors. The novelty of this method motivated us to adopt this work in our framework as we try to develop descriptors by modelling the pressure of the writing at a point as the mass of that point.

The concept of histogram-based descriptors have been used extensively in the field of computer vision. The most popular work in this direction is the Histogram of Oriented Gradients (HOG) proposed by Dalal *et. al.* [17]. The HOG features are computed by counting occurrences of gradient orientation in localized portions of an image by taking various sliding windows across the image. Histogram-based descriptors have also given state of the art results in several online writer identification tasks as well. Venugopal *et. al.* [11] proposed a substroke level histogram-based descriptor by taking inspiration from the delta and acceleration

coefficients popular in the domain of speech processing. Dwivedi *et. al.* [15] also proposed a sub-stroke level histogram-based descriptor and they develop a set of prototypes from these descriptors using the concept of sparse coding based descriptors.

In order to tackle the problem of lack of data LSTM autoencoders can be used to encode the sequence of descriptors from a document or a text-line into a fixed length vector, i. e., we could develop document descriptors using LSTM autoencoders. A common problem encountered is that of over-fitting, where a model is not able to generalize well to unseen data. In order to tackle this, Lei *et. al.* [14] proposed a supervised autoencoder to improve the generalization performance. The supervised autoencoder is designed in such a way that it jointly predicts the true labels and the and the inputs patterns from the compressed representation it has learnt.

Chapter 3

Methodology

Traditional methods in the field of online writer identification use only spatial features like angle, curliness, curvature and writing direction. The classification methods used for these features also do not take into account the temporal information present in the sequence of sample points. The idea of using Recurrent Neural Networks to capture the temporal information in online documents was first proposed by Zhang *et. al.* [7]. They used a recurrent neural network to model the temporal information, but the spatial information was modelled by them in a very simplistic manner. To improve upon their proposed system and use the full potential of the spatio-temporal sequence we propose a method wherein we first extract spatial features from the online document and then classify these sequential spatial features using a sequence modelling technique. We also move from the traditional approach of using only the angle, curliness and curvature of the handwriting and focus more on the pressure of the stylus as we believe that angle, curvature and curliness may depend on the textual content of the document, but the pressure variations is something which do not depend on the textual content.

The stroke of a online handwritten document is considered to be composed of numerous fragments called sub-strokes, which are formed of a group of n temporal points s_i = (x_i, y_i) , where $i \in [1, n]$. The substrokes are considered as our basic temporal units and we construct a feature vector for each sub-stroke using simple yet intuitive concepts of physics. The spatial feature vectors of sequential substrokes are then used to train a temporal sequence model. The temporal sequence model is used to capture the temporal information present in the sequence of sub-strokes.

We first describe the pre-processing steps that we have performed to remove certain anomalies in the data collection procedure. In the subsequent sections we describe the two frameworks

that we have explored in the described setting.

3.1 Pre-processing and Sub-stroke generation

Prior to the sub-stroke generation and the feature extraction processes, we perform certain pre-processing steps to correct anomalies in the data collection procedure.

- **Same time stamp removal** : There might be some data points with different *x* and *y* coordinates but with same time stamp. As a result of which certain features like the momentum of a point could become infinite. To avoid such issues, we average the values of *x* and *y* coordinates and pressure values for points having same time stamp .

- **Removing points with same x and y co-ordinates** : Due to possible issues with the recording instruments it can register numerous readings with the same x and y co-ordinates consecutively. This will create possible issues when we calculate the force between two points in a sub-stroke. To take care of this we remove all the points in a sub-stroke with the same x and y co-ordinates.

- **Removing spurious points** : Certain noisy points may exist in the data when there may be point which does not fall into the current stroke. To remove such points we consider a threshold, such that, if the distance to that point from all the point in the stroke is greater than the threshold, that point is dropped from the data.

After we perform the following pre-processing steps we segment the handwritten trace into strokes using the pen up and pen down status information. This is done to ensure that two successive points which belong to two different strokes do not affect each other while calculating the force at each point. We also segment each stroke further into sub-strokes, which form our basic temporal unit. We consider fixed length sub-strokes by splitting a stroke into smaller fragments containing a fixed number of sample points. Each successive sub-strokes are also allowed to have an overlap that is decided by a stride (r).

In our work, the sub-stroke is considered as a set of N temporal points (x_i, y_i), where $i \in [1, N]$. Moreover, during generation of the sub-strokes, we choose the value of the stride r to be $N/2$.

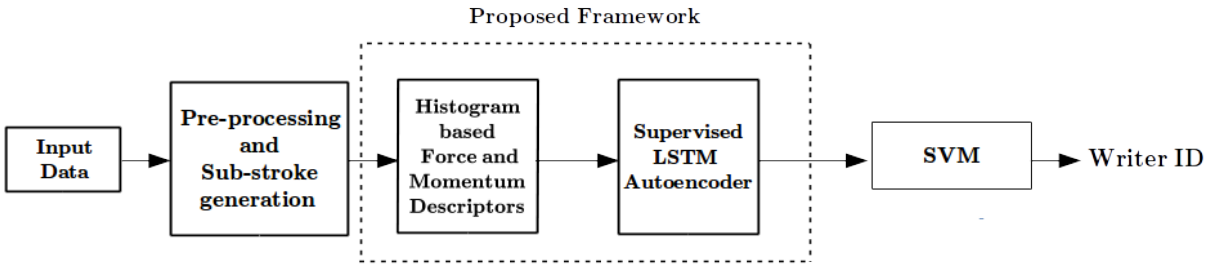

Figure 3.1: Block diagram of the proposed online writer identification system.

3.2 Proposed Framework with Histogram Features and Supervised LSTM Autoencoder

In this part of the thesis, we explore a text-independent online writer identification system by taking the pressure of the stylus as the characteristic of a person's handwriting. We segment the online handwritten trace into sub-strokes and extract histogram based descriptors from them. The generation of the histograms are inspired by the Pattern of Local Gravitational Force (PLGF) from the area of computer vision. In order to get a more discriminative representation, we take inspiration from another common concept in physics known as momentum. This helps us in capturing the temporal dynamics within a sub-stroke. The momentum at a particular point is calculated using the pressure reading and the velocity of that point. Our descriptors are used to capture the spatial information present in an online handwritten trace. The proposed histogram based features have an advantage as they do not depend on the number of points that we consider in a sub-stroke.

Another major advantage of our method is the use of Long Short Term Memory (LSTM) autoencoder to capture the temporal information in the sequence of histogram based descriptors. We have considered a sub-stroke to be the basic temporal unit of a handwriting trace and the whole handwritten document is thought to consist of numerous such sequential sub-strokes. The LSTM autoencoder is used to encode the sequence of sub-stroke descriptors from each line in the corpus into a fixed-length vector for final classification. The fixed-length vector is useful in capturing discriminative temporal information about a writer. This compressed representation is then used to train a Support Vector Machine (SVM) for final classification.

To summarize the preceding discussion, we pictorially highlight the proposed framework in Fig. 3.1.

3.2.1 Histogram-based descriptors

According to the law of universal gravitation, we know that each body creates a gravitational field around it. Analogous to this, we consider that each point in the handwritten trace creates a field, due to which it exerts force on other points from the same stroke. To calculate this force, each sample point in the trace is considered to be a point object with the mass of the object being equal to the pressure value at that point.

At first, we calculate the force ($\mathbf{F_i}$) on a single point, (x_i, y_i), due to all the points which exist within the field of the i^{th} point. The force ($\mathbf{F_i}$) on a single point (i) with pressure reading, p_i is calculated using the formula,

$$\mathbf{F_i} = \sum_{j=-k, j\neq 0}^{k} \frac{G p_i p_{i+j}}{|\mathbf{r_{i,\,i+j}}|^2} \frac{\mathbf{r_{i,\,i+j}}}{|\mathbf{r_{i,\,i+j}}|} \tag{3.1}$$

The value of the gravitational force constant (G) is taken to be 1. We consider that $2k$ is total number of points inside the field of the i^{th} point. In our case we take the value of k=2. We take $\mathbf{r_{i,\,i+j}}$ as the position vector pointing from point i to point $i + j$. The position vector ($\mathbf{r_{i,\,i+j}}$) is defined as follows.

$$\mathbf{r_{i,\,i+j}} = [r_x \quad r_y]^T = [x_{i+j} - x_i \quad y_{i+j} - y_i]^T \tag{3.2}$$

The total force vector ($\mathbf{F_i}$) is then segregated into two components, the x-component and the y-component which we denote as (F_x, F_y) for convenience.

Momentum is defined as the product of mass and velocity of a body. Here, we consider the pressure value at a sample point as an imaginary point mass, similar to the formulation of the force based histogram descriptors. The velocity ($\mathbf{v_i}$) of a point, (x_i, y_i), is calculated considering its immediate neighbour.

$$\mathbf{v_i} = [v_{ix} \quad v_{iy}]^T = \left[\frac{x_{i+1} - x_i}{t_{i+1} - t_i} \quad \frac{y_{i+1} - y_i}{t_{i+1} - t_i}\right]^T \tag{3.3}$$

The momentum of a single point with pressure reading, p_i is then calculated using the formula,

$$\mathbf{P_i} = p_i \mathbf{v_i} \tag{3.4}$$

We compute our histogram based features for the sub-strokes as follows. We calculate the magnitude of the force F_i and the angle of the force α_{fi} at each point inside the sub-stroke as,

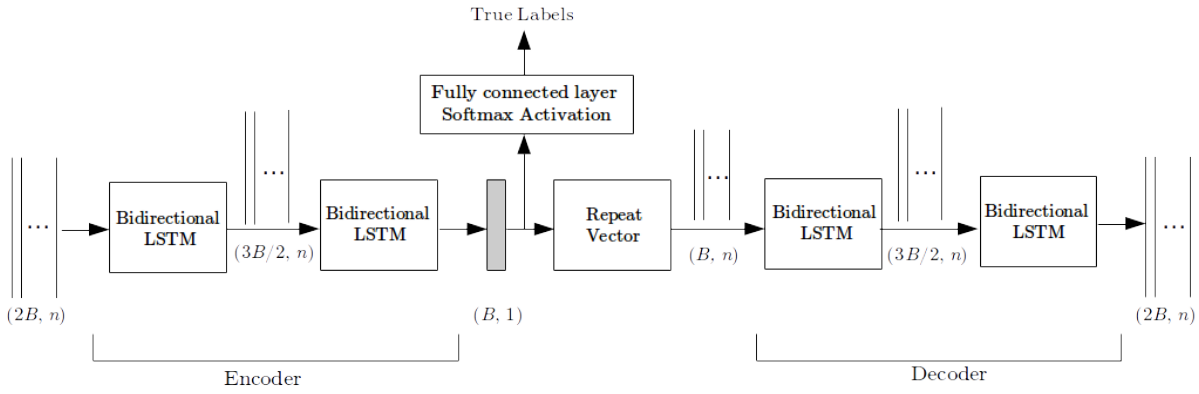

Figure 3.2: Proposed supervised LSTM Autoencoder with the two tuple denoting the size of the output after each layer and where n is the number of sub-strokes in a line and B is number of bins. The shaded block is used to denote the compressed representation learnt.

$$F_i = \sqrt{{F_x}^2 + {F_y}^2}, \qquad \alpha_{fi} = tan^{-1}\frac{F_y}{F_x} \tag{3.5}$$

Subsequently, we divide the whole angle space [0, 2π) into B bins. For each point *i* in the sub-stroke we take its corresponding (F_i, α_{fi}) values and a bin is then determined on the basis of the angle value α_{fi}. The vote of the selected bin is increased by the magnitude value F_i. This accumulation of force values is done for all the points in a sub-stroke under consideration and this gives us a B-dimensional histogram descriptor $\mathbf{x_f}$ for a sub-stroke.

Likewise, we also segregate the momentum into two components P_x and P_y. We calculate the magnitude of the momentum P_i and the angle of the momentum α_{pi} at each point.

$$P_i = \sqrt{{P_x}^2 + {P_y}^2}, \qquad \alpha_{pi} = tan^{-1}\frac{P_y}{P_x} \tag{3.6}$$

A similar voting procedure, as described above, is applied to get the B-dimensional histogram descriptor $\mathbf{x_p}$. Following this, we concatenate the two histograms to form the $2 \times B$ dimensional descriptor

$$\mathbf{x} = [\mathbf{x_f} \quad \mathbf{x_p}] \tag{3.7}$$

A z-score normalization is performed on the histogram features, so as to ensure that they follow the standard normal distribution. The sequence of descriptors obtained from sub-strokes of the hand-written data are fed to a deep learning based LSTM architecture, the details of which are discussed in the following Section.

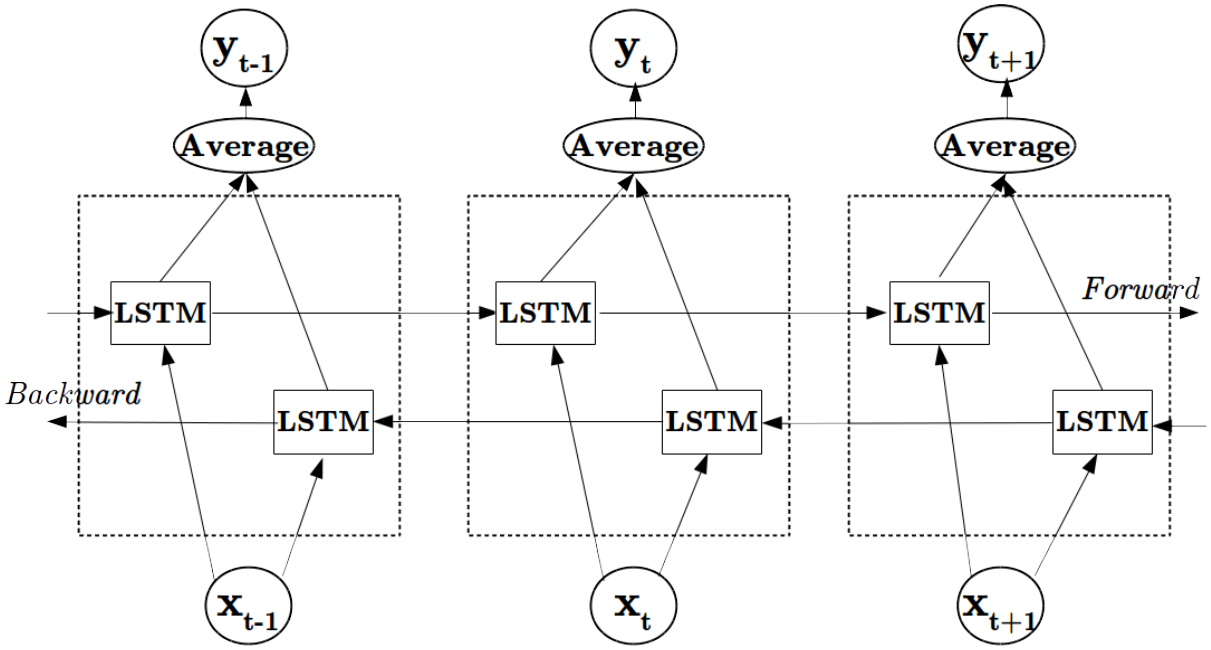

Figure 3.3: Diagram showing the internal representation of a Bidirectional LSTM layer in Fig. 3.2. x_t denotes the feature vector for a particular sub-stroke and y_t denotes the corresponding output.

3.2.2 Supervised Long Short Term Memory (LSTM) Autoencoder Network

Recurrent Neural Networks have proven to be state-of-the-art for sequence classification and encoding tasks. But, such networks maintain an activation for each subsequent time step, that makes them very deep. The issue, as such, arises at the time of training, where we usually encounter the vanishing gradient problem. Another major problem with the use of simple recurrent neural networks is the lack of the ability to remember information for a long period of time. The actual information may be separated from the time step where it is needed by a large amount of data which is irrelevant. In such cases, simple recurrent neural networks fail to give satisfactory results. As an alleviation to these problems, the Long Short Term Memory (LSTM) Networks was proposed for sequence modelling to learn long-term dependencies.

In order to generate some new information, Recurrent Neural Networks used to modify the whole of the existing information by applying some non-linear transfer function. This obscures the concept of important information and non-important information, as the entire information is being modified. Long Short Term Memory (LSTM) layers on the other hand try to make minimal changes to the information in order to generate new information. These minimal changes

are brought about by simple operations like multiplications and addition. The information generated flows from one time-step to the next through cell states. This gives the LSTMs the power to choose what information to remember and what information to forget.

Typically, Long Short Term Memory (LSTM) units consists of a memory unit **c**, a hidden state **h** and three gates (input (**I**)), forget(**F**)), output(**O**)) to control the flow of information. At any time step t, the LSTM unit receives an input $\mathbf{x}_t$, the previous hidden state $\mathbf{h}_{t-1}$ and utilizes the learned weights (W_f, W_i, W_o, W_c, U_f, U_i, U_o and U_c) to calculate the activation of the gates. The computations taking place inside an LSTM layer can be summarized as follows [23]:

$$\mathbf{F}_t = \sigma_f(W_f x_t + U_f h_{t-1} + b_f)$$

$$\mathbf{I}_t = \sigma_i(W_i x_t + U_i h_{t-1} + b_i)$$

$$\mathbf{O}_t = \sigma_o(W_o x_t + U_o h_{t-1} + b_o)$$

$$\mathbf{c}_t = \mathbf{F}_t \odot \mathbf{c}_{t-1} + \mathbf{I}_t \odot \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$

$$\mathbf{h}_t = \mathbf{O}_t \odot \sigma_h(c_t)$$

where $\odot$ denotes the Hadamard product (element-wise product) between two matrices, x_t denotes the input at a particular time step *t*, h_{t-1} denotes the hidden activation of the previous time step, c_t denotes the cell state vector. The matrices W and b denotes the corresponding weights and biases respectively.

While LSTMs capture long-term dependencies, the increased number of weight parameters necessitates that we have a large amount of data at our disposal to train them satisfactorily. A common problem encountered is that of over-fitting, where a model is not able to generalize well to unseen data. In order to tackle this, Lei *et. al.* [14] proposed a supervised autoencoder to improve the generalization performance. Taking inspiration from this work, we develop a supervised LSTM autoencoder to encode the sequence of sub-stroke descriptors and learn a compressed representation for the sequence. This helps us in automatically removing irrelevant and redundant temporal information present in the data.

Specifically, we train the LSTM Autoencoder in a supervised manner by configuring the network in such a way that it jointly predicts the true labels and the inputs sequences from the compressed representation it has learnt. This helped us to greatly reduce the problem of the model overfitting on the training set.

Our LSTM Autoencoder network (Fig. 3.2) is a sequential arrangement of four bidirectional layers (Fig. 3.3). The first half of the structure consisting of two bidirectional layers, forms the encoder architecture for the input sequences while the second half (also comprising of two bidirectional layers) forms the decoder architecture. The output of the encoder forms the bottleneck and it gives us the compressed representation of the input sequences (refer the shaded block in Fig. 2). This learnt representation is repeated for the required number of time steps using a RepeatVector layer before it is used as an input for the decoder. The decoder as such, decodes the representations thus learnt and reconstructs the input.

The compressed representation learnt by the encoder is also simultaneously fed to a fully connected layer which is used to predict the true label of the input sequence using a softmax activation function. The autoencoder is thus trained in a supervised manner in that we provide the network with the true labels of the input sequence as well as train it to reconstruct the input sequence.

3.2.3 Writer Identification

The LSTM Autoencoder thus proposed can deal with variable length sequences and it can be used to learn compressed representation of a sequence of sub-stroke descriptors at both the text-line level and the paragraph level. For, this thesis we only consider experiments at the text-line level. The text-line level compressed representation thus obtained are fed to the Support Vector Machine (SVM) classifier to obtain the authorship of the document. We use SVM in place of a fully connected layer because of the large number of parameters involved in a complex classification problem. Support Vector Machines are also more effective than fully connected layers in learning from small datasets. We use a one-vs-rest strategy to train the SVM, as we train a separate SVM for each writer such that the descriptors corresponding to that writer are positive samples and the rest are negative samples. We used the Radial Basis Function (RBF) kernel to train the SVM. We also used grid-search to determine the optimal values of the regularization parameter and the kernel coefficient.

3.3 Proposed Framework with PLGF features and End-to-End LSTM Network

3.3.1 Pattern of Local Gravitational Force (PLGF)

The handwriting of a person is composed of numerous sequential points. The location of these points primarily depends on the language in which the person is writing the text. Any variations of the location of the points, due to the influence of the writer, is very minimal. Hence, the identification systems developed using features extracted from the x and y co-ordinates are primarily text-dependent in nature.

To overcome this problem, we propose a novel descriptor based on the Pattern of Local Gravitational Force (PLGF), proposed in [8]. The authors of this paper have been influenced by the law of gravitation and they have proposed a descriptor which encodes the direction and magnitude of gravitation for each pixel by considering the pixels as point masses. They have a proposed a novel neighbors to center difference binary pattern (NCDBP) encoding scheme to encode the force magnitude at each pixel. The force direction is encoded using a threshold value based encoding which is inspired by the Local Binary Pattern (LBP).

Inspired from the success of the PLGF descriptor in face recognition tasks, we try to incorporate this methodology into the framework of writer identification by making simple, yet intuitive changes to their proposed method. Just like an image contains numerous pixels, a handwritten trace consists of numerous sample points. But, unlike an image the points do not have a pixel intensity associated with it. We thus make use of the pressure value at a point and treat the pressure values as point masses. This helps us in obtaining meaningful discriminative features as the pressure of a handwriting is more characteristic of the writer than the angle, slope or curliness of the trace. The methodology of obtaining the descriptors of a sub-stroke, which is the basic temporal unit considered in our works, are discussed next.

Force on a point

According to the law of universal gravitation, we know that each body creates a gravitational field around it and whenever any other body comes into that field, it is attracted by the first body. Analogous to this, we consider that each point in the handwritten trace has a field around it, where it exerts its force on other point masses.

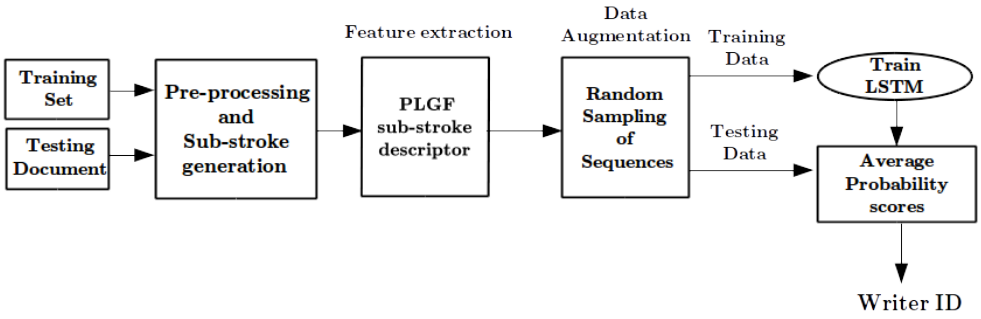

Figure 3.4: Proposed online writer identification system.

At first, we calculate the force ($\mathbf{F_i}$) on a single point, (x_i, y_i), due to all the points which exist within the field of that single point. The force on a single point with pressure reading, p_i is calculated using the formula,

$$\mathbf{F_i} = \sum_{j=-k, j\neq 0}^{k} \frac{G p_i p_{i+j}}{|\mathbf{r_{i,\,i+j}}|^2} \frac{\mathbf{r_{i,\,i+j}}}{|\mathbf{r_{i,\,i+j}}|} \tag{3.8}$$

here $\mathbf{F_i}$ is the total force on the i^{th} point, G is the gravitational force constant, p_i is the pressure reading at the i^{th} point, 2k is total number of points inside the field of the i^{th} point, $\mathbf{r_{i,\,i+j}}$ is the position vector pointing from point i to point i+j and $|\mathbf{r_{i,\,i+j}}|$ denotes the magnitude of the vector $\mathbf{r_{i,\,i+j}}$. The total force $\mathbf{F_i}$ vector is then segregated into two components, the force magnitude and the force angle which we denote as (f_i, a_i) for convenience.

Pattern of Local Gravitational Force (PLGF) descriptor

After calculating the total gravitational force at each point we now want to encode this force magnitude and force angle to formulate the pattern of local gravitational force. We thus could represent the original handwriting in the transformed domain as a two tuple, force magnitude and force angle at each point. The sub-stroke descriptor is then calculated from this transformed domain. We consider that a sub-stroke consists of n points, and take the corresponding (f_i, a_i) for those n points. We then find the point which is at the centre of the sub-stroke. To ease this process we consider n to be always an odd number. Now we encode the force and the angle values separately in the following manner:

- The force magnitude is encoded using a differential encoding strategy. We use two different thresholds to encode the force magnitude values. The first threshold value is the

mean of the positive differences and the second threshold value is the mean of the negative differences. At first we form a set of difference values, which stores the differences of the force magnitudes between various points in the sub-stroke and the centre point (c) which is represented as d_f,

$$d_f = [f_1 - f_c, f_2 - f_c, ..f_{c-1} - f_c, f_{c+1} - f_c, ..., f_{n-1} - f_c, f_n - f_c]$$

The positive threshold value (d_f^+) is obtained by taking the mean of the positive values obtained from the set of differences. The negative threshold value (d_f^-) is obtained by taking the mean of the negative values obtained from the set of differences.

$$d_f^+ = \frac{\sum_{i=1}^{pc}(d_f)_i}{pc} \qquad\qquad d_f^- = \frac{\sum_{i=1}^{nc}(d_f)_i}{nc}$$

here *pc* is the number of elements in the set, d_f having elements $(d_f)_i \geq 0$ and *nc* is the number of elements in the set d_f having elements $(d_f)_i < 0$. We then form a histogram which consists of four bins. The first bin denotes the number of points with negative difference and having their values lesser than the negative threshold (d_f^-). The second bin denotes the number of points with negative difference and having their values greater than the negative threshold (d_f^-). The third bin denotes the number of points with positive difference and having their values lesser than the positive threshold (d_f^+) and the fourth bin denotes the number of points with positive difference and having their values greater than the positive threshold (d_f^+). We thus encode the force magnitude of the k^{th} sub-stroke using this four dimensional vector ($\mathbf{\Omega_k}$).

- The angle value is encoded in a much easier manner. We calculate the angle difference vector by taking angle difference between various points in the sub-stroke and the centre point (c) which if represented as d_a,

$$d_a = [a_1 - a_c, a_2 - a_c, ...a_{c-1} - a_c, a_{c+1} - a_c, ..., a_{n-1} - a_c, a_n - a_c]$$

The angle differences are then encoded using a fixed threshold value *thr* = $\frac{\pi}{8}$. The value of *thr* is adopted from the experiments performed by [8]. A histogram with two bins is formed where the first bin represents the number of points where angle difference is less than *thr* and the second bin represents the number of points where the difference value is greater than *thr*. So, we have a two dimensional vector ($\boldsymbol{\alpha_k}$) to encode the angle value of the k^{th} sub-stroke.

We thus formed a descriptor of a sub-stroke by concatenating the $\mathbf{\Omega_k}$ and $\boldsymbol{\alpha}_\mathbf{k}$. This six dimensional vector is created for each subsequent sub-strokes and this sequence of vectors form our training set.

3.3.2 End-to-end Long Short Term Memory (LSTM) Networks

The data captured for the purpose of online writer identification is in the form of temporal sequences. Recently, recurrent neural networks (RNNs) have given state-of-the-art results in classification or recognition tasks for time-series data. However, despite the stunning results, RNNs have a shortcoming as they cannot capture long term dependencies. Hence a more powerful version of Recurrent Neural Networks was proposed, which are Long Short Term Memory (LSTM) units. LSTM networks are a special type of RNNs which are capable of learning long-term dependencies from temporal data. They are now widely used as they have shown to work well on a large variety of problems. The computations taking place inside a LSTM cell can be summarized as follows [23]:

$$\mathbf{F}_t = \sigma_f(W_f x_t + U_f h_{t-1} + b_f)$$

$$\mathbf{I}_t = \sigma_i(W_i x_t + U_i h_{t-1} + b_i)$$

$$\mathbf{O}_t = \sigma_o(W_o x_t + U_o h_{t-1} + b_o)$$

$$\mathbf{c}_t = \mathbf{F}_t \odot \mathbf{c}_{t-1} + \mathbf{I}_t \odot \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$

$$\mathbf{h}_t = \mathbf{O}_t \odot \sigma_h(c_t)$$

where $\odot$ denotes the Hadamard product (element-wise product) between two matrices, x_t denotes the input at a particular time step t, h_{t-1} denotes the hidden activation of the previous time step, c_t denotes the cell state vector. The matrices W and b denotes the corresponding weights and biases respectively.

The basic temporal unit that we have considered in our problem is a sub-stroke and hence we developed descriptors for each sequential sub-stroke using the method described above. This temporal sequences of feature vectors extracted by the descriptor is used to train a shallow 3 layer LSTM network. The first 2 layers of the network are LSTM layers, followed by a dense layer at the end. The dense layer has a softmax activation function since it is used to predict the writer id to which the sample belongs. The scarcity of data prevented us from training a much

deeper model. In order, to tackle the problem of overfitting we also use a dropout of 20%. The loss function used by us is the categorical cross entropy and the optimizer used is Adam [9]. In order to perform data augmentation we use a novel training technique. We do not pass the whole sequence of descriptors as a single training sample. Instead, we use randomly sampled contiguous sequence of descriptors, which consists of 100 sequential sub-stroke descriptors. After sampling 100 sequential descriptors we use a random time step to sample another 100 sequential descriptors. This greatly boosted the performance of our model.

Chapter 4

Experiments and Results

4.1 Dataset

All our experiments are done on the BIT CASIA dataset. The BIT CASIA dataset [10] contains data of 187 writers in the Chinese language and 134 writers in the English language dataset. Each writer has written a total of three pages of English text and four pages of Chinese text. The x-coordinate, y-coordinate, time stamp information, pressure of the stylus, pen up or pen down status, azimuth and altitude of the pen trace are recorded as a part of the online handwriting data.

For the single-language experiments we chose both the English and the Chinese language dataset. For each writer in both the datasets we randomly select two pages as a part of the training set and the third page is kept for testing. The writer identification experiments are done at the text-line level.

For the cross-language experiments we chose the Chinese-English language dataset. The Chinese-English dataset consists of three pages written in Chinese and three pages in English. For each writer, we randomly select four pages for the training set while reserving the remaining two pages for evaluation.

The process of generating the training and testing set, extracting the histogram descriptors, training the LSTM Autoencoder and SVM, is performed ten times. The average percentage of correctly identified text-lines are reported.

Table 4.1: Performance of our proposed methodology at the Text-Line Level using different values of Number of Bins B and Size of sub-stroke N for English language dataset.

Size of	Number of bins, B					
sub-stroke(N)	5	10	15	20	25	30
10	55.41	65.38	74.06	77.33	77.65	74.81
15	58.22	68.78	72.44	78.67	80.03	77.93
20	61.17	72.28	74.54	80.31	**81.75**	79.26
25	60.32	70.21	73.68	80.93	80.83	74.55
30	62.45	71.01	70.54	74.56	77.28	68.44

4.2 Proposed Framework with Histogram Features and Supervised LSTM Autoencoder

4.2.1 Implementation Details

We use bidirectional LSTM units to ensure that our model captures both forward and backward temporal information. The time sequences obtained from both forward and backward processing of the input sequences are combined together by averaging the sequences. As indicated in Fig 2, each input sequence is encoded into a B dimensional vector by the auto-encoder, where B is the number of bins used in generating the histograms $\mathbf{x_f}$ and $\mathbf{x_p}$ (Equation 7). The optimizer used for training our model is RMSprop and we use a mini-batch size of 16. The initial learning rate is set to be 0.01. We set the number of epochs to be 1000.

We employ two different loss functions to train our model, namely the mean square error and the categorical cross entropy. The former function aims to minimize the reconstruction error while the later deals with the error incurred in the prediction of true labels. We also use Model Checkpoint and Early Stopping to prevent over-fitting of our model due to excessive number of epochs. To monitor the problem of over-fitting, we consider 10% of the training set as the validation set.

4.2.2 Efficacy of the proposed method on the English language database

As a first experiment, we test our proposed method on the English language dataset. The task of single-language writer identification is easier than cross-language writer identification. Since, we have taken the sub-stroke as the basic temporal unit of our model we evaluate our proposal by varying the size (N) of the sub-stroke. We also vary subsequently the number of bins (B) for the histogram-based descriptor. The size of sub-stroke determines the number of temporal units for which we could consider the temporal characteristics of the writer to be approximately similar and the number of bins (B) determines the dimensionality of the descriptor of each sub-stroke. Table 4.1 can be used to observe the results the text-line level. We could observe that for a fixed size of sub-stroke the performance of our proposed methodology first increases and then decreases. The optimal number of bins for our proposed methodology is 25 (B=25). This is because with increase in the number of bins the number of features increases and we know that with the increase in the number of features the machine learning models always face the problem of overfitting. So, we should always use an optimal number of features.

On keeping the number of bins constant and increasing the sub-stroke size, we observe that the performance of the model again decreases after an optimal value (N=20). This is because the sub-stroke is considered as the basic temporal unit of our methodology and on incorporating more points in the sub-stroke we lose the basic temporal structure already present in the data. This helps us in concluding that the local dynamics of handwritten trace is more discriminative than the global dynamics as a smaller sub-stroke size means we are forming descriptors for local variations in a writer's style. At the configuration of B=25 and N=20 our proposed methodology is able to achieve the best accuracy of 81.75% at the text-line level.

4.2.3 Efficacy of the proposed method on the Chinese language database

We also test our proposed methodology on the Chinese database. The Chinese database is more complex than the English database as it contains a large number of strokes which are very small in length. To test our model we follow a similar protocol as followed in the case of English database. We vary the values of number of bins B and the size of the sub-stroke N to find the optimal number of bins and the optimal size of a sub-stroke. We observe from Table 4.2, that the best performance is obtained at the configuration B=20 and N=20 for which we obtain an accuracy of 80.97% at the text-line level.

Table 4.2: Performance of our proposed methodology at the Text-Line Level using different values of Number of Bins B and Size of sub-stroke N for Chinese language dataset.

Size of	Number of bins, B					
sub-stroke(N)	5	10	15	20	25	30
10	65.14	69.22	73.44	76.23	77.54	77.23
15	63.23	69.97	72.58	76.94	79.65	78.16
20	66.83	72.65	78.12	**80.97**	80.04	77.25
25	67.94	71.23	75.86	78.71	75.74	76.84
30	65.33	68.96	69.73	77.82	73.59	72.34

Table 4.3: Performance of our proposed methodology at the Text-Line Level using different values of Number of Bins (B) and Size of sub-stroke (N) for Chinese-English dataset.

Size of	Number of bins, B					
sub-stroke(N)	5	10	15	20	25	30
10	45.34	54.76	58.93	68.73	73.94	69.20
15	48.92	59.68	62.87	70.05	72.40	71.91
20	56.99	58.52	67.18	75.23	**76.84**	73.29
25	48.08	53.71	62.89	71.22	75.64	74.74
30	50.72	57.46	60.93	65.57	68.23	70.22

Table 4.4: Performance comparison of proposed writer identification system for different combinations of histogram feature sets.

Adopted	Dataset		
Methodology	English	Chinese-English	Chinese
$\mathbf{x_f}$	75.34	67.93	74.65
$\mathbf{x_p}$	73.29	70.73	73.86
$[\mathbf{x_f}, \mathbf{x_p}]$	**81.75**	**76.84**	**80.97**

4.2.4 Efficacy of the proposed method in a cross-language setting

Since the features extracted using pressure is independent of the content and language of the handwritten trace and more characteristic of the writer, we could use it to perform cross-language writer identification. Hence in this section we consider the efficacy of our proposed features in a cross-language setting by considering the Chinese-English dataset. We train the model on a mixed dataset of Chinese and English documents and test our models again on a mixture of the documents without knowing the language in which a particular document is written.

Similar, to the previous two experimental settings, we vary the values of number of bins B and the size of the sub-stroke N to find the optimal values for best performance. We observe from Table 4.3, that the proposal with the configuration B=25 and N=20 gives the highest performance accuracy of 76.84% at the text-line level.

We also evaluate the performance of the sub-stroke descriptor with various combinations of the histogram-based feature vectors ($\mathbf{x_f}$ and $\mathbf{x_p}$) extracted from the substrokes. The results with various combination of descriptors used can be seen in Table 4.4. From the table, we observe that the augmented feature set ([$\mathbf{x_f}$, $\mathbf{x_p}$]) helps us in improving the identification rate as compared to the feature sets $\mathbf{x_f}$ and $\mathbf{x_p}$ alone. This is due to the fact, that the momentum histogram features captures the information about the temporal dynamics within a sub-stroke, which helps us in getting a more discriminative representation.

4.2.5 Comparison with prior works

The use of the raw x and y co-ordinate values, pen up and pen down statuses along with an end-to-end Bidirectional LSTM network has led to the state-of-the-art result at the paragraph level in [7]. In this subsection, we implement various learning architectures listed in Table 4.5 at the text-line level and provide a quantitative comparison of the above raw features with our proposed descriptors. We also provide a quantitative comparison between our proposed supervised LSTM autoencoder based approach and an end-to-end deep learning framework of [7] at the text-line level.

As proposed in [7], we take the difference between the x co-ordinates and y co-ordinates of successive points to form our feature vector. However, in order to present a fair comparison, we do not use the product of pen up and pen down status as mentioned in [7]. Instead we use

Table 4.5: Comparison of the proposed framework with the existing features as proposed in [7].

Adopted Methodology	Dataset		
	English	Chinese-English	Chinese
Existing Features+LSTM	63.28	60.55	65.19
Existing Features+LSTM Autoencoder	67.74	61.28	66.63
Histogram Features+LSTM	69.93	64.68	69.72
Histogram Features+LSTM Autoencoder	73.83	72.94	76.95
Existing Features+BiLSTM	65.41	63.75	69.54
Existing Features+BiLSTM Autoencoder	73.36	70.13	73.44
Histogram Features+BiLSTM	77.29	72.20	75.33
Proposed Framework	**81.75**	**76.84**	**80.97**

the product of pressure values of two successive points and append it to the feature vector. We then perform z-score normalization to obtain our final sequence of feature vectors for the deep learning systems.

The feature vectors of each text-line are then either classified by using a end-to-end network or encoded to form a compressed representation using an autoencoder. The results of all the implemented systems are compiled in Table 4.5. An important observation from this table is the significant improvement of our proposed framework when compared with the system comprising of basic features proposed in [7] and a BiLSTM network. The results also show an improvement when we use BiLSTM blocks instead of LSTM blocks. This helps us in concluding that in handwriting both forward and backward temporal processing is essential to capture the underlying characteristics. We could also observe that the autoencoder network helps us in alleviating the requirement of a large amount of training data required to train an end-to-end deep learning network.

The work by Venugopal *et. al.* [11] report an accuracy of 80.85% at the text-line level. We could see that our proposed approach outperforms this method. The authors of this paper also mention that their work on the CASIA database outperforms the existing methods at the text-line level.

4.3 Proposed Framework with PLGF features and End-to-End LSTM Network

4.3.1 Implementation Details

The methodology described for extracting the sub-stroke is descriptors is performed for all the document files after performing the necessary pre-processing steps. To extract our features we consider the field of a point to be effective upto 5 points on either side of the point, i. e., we take the value of k = 5. The sub-stroke size that we consider here is 11, i.e., n = 11. The model is not trained by taking the whole sequence of sub-stroke descriptors as a single training sample as recurrent neural networks are not designed to be trained from a small amount of data. The unavailability of pre-trained recurrent neural networks in the domain of handwriting recognition meant that we also could not use transfer learning. We overcame this shortcoming through intelligent data augmentation. We use randomly sampled contiguous sequence of descriptors, which consists of 100 sequential sub-stroke descriptors as a single training sample. After sampling 100 sequential descriptors we use a random time step to sample another 100 sequential descriptors. This greatly boosted the performance of our model. To prevent overfitting we used dropout, with a dropout rate of 0.2. We used an initial learning rate of 0.01. The LSTM layers are designed to have a *tanh* activation function. We trained our neural network model for 1500 epochs as we have used random initialization to initialize the parameters of our model. To prevent overfitting we were also monitoring the validation loss of our model and used the *ModelCheckpoint* function in *Keras* to store our best model on the basis of the minimum validation loss achieved. We also reduced the learning rate (α) of our training scheme by a factor of 0.2 whenever the validation loss did not improve for 15 epochs. This allowed our model to not get stuck in plateau regions of the loss function. The number of parameters in our model are also not fixed and are varied according to the amount of training data available to us.

During testing time, we again randomly sample sequences consisting of 100 sub-stroke descriptors from a document ad generate the prediction probabilities from the softmax layer. This is repeated with all such sequence of descriptors in the document. The prediction probability vectors obtained across a document are averaged and the class having the highest probability is assigned the authorship of that particular document. This implementation is motivated by the success of ensemble classifiers where numerous weak classifiers together provide much better

classification results.

We use bidirectional LSTM units to ensure that our model captures both forward and backward temporal information. The time sequences obtained from both forward and backward processing of the input sequences are combined together by averaging the sequences. As indicated in Fig 2, each input sequence is encoded into a B dimensional vector by the auto-encoder, where B is the number of bins used in generating the histograms $\mathbf{x_f}$ and $\mathbf{x_p}$ (Equation 7). The optimizer used for training our model is RMSprop and we use a mini-batch size of 16. The initial learning rate is set to be 0.01. We set the number of epochs to be 1000.

We employ two different loss functions to train our model, namely the mean square error and the categorical cross entropy. The former function aims to minimize the reconstruction error while the later deals with the error incurred in the prediction of true labels. We also use Model Checkpoint and Early Stopping to prevent over-fitting of our model due to excessive number of epochs. To monitor the problem of over-fitting, we consider 10% of the training set as the validation set.

4.3.2 Results

In this section, we present the accuracy that our descriptor and LSTM model has been able to achieve. We tested our model on the English database and on the Chinese English dataset to see how our descriptor performed in a cross-language setting. We at first trained the neural network on the whole English writer dataset. But, we observed that after certain number of epochs the training loss continued to decrease but the validation loss remained constant. This led to the conclusion that our model was overfitting on the training set due to limited amount of training data and it was not able to generalize well enough to the training set. We also observed that after sometime, although the validation loss slightly increased due to overfitting, the validation accuracy was still increasing. This could be due to two possible reasons. Firstly, some data samples with probabilities very near the maximum probability class get predicted better and so their output class changes. This leads to an improvement in the accuracy. Some data points with very less probability for the true class keeps getting worse. This leads to an increase in the validation loss while the validation accuracy remains same.

In order to tackle these problems and establish the efficacy of our proposed method we decided to train the neural network with a much simpler problem at hand. To do this we trained the neural network with a subset of the whole dataset, i.e., we first tested the neural network

Number of classes	Accuracy (%)
10	100.00
15	93.33
20	85.00
25	80.00
30	76.67

Table 4.6: Accuracy of our model when trained on the English dataset at the paragraph level.

Number of classes	Accuracy (%)
10	80.00
15	73.33
20	72.50
25	68.00
30	63.33

Table 4.7: Accuracy of our model when trained on the Chinese and English dataset at the paragraph level.

with 10 classes and then with 15 classes and then with 20 classes and so on. We observed that when we train the neural network with 10 classes we obtain very high accuracy and as we increase the number of classes, the accuracy keeps on decreasing. This led us to conclude that the neural network can be trained with a limited amount of data if the classification task is simple one. The amount of data required is directly proportional to the complexity of the classification problem and the complexity of the classification problem is decided by the number of classes. The accuracy obtained for various size of the training and testing set is presented in Table 4.6.

In Table 4.7 we present the performance of our proposed system in text-independent system. The results shown here are obtained are shown on two documents per writer. We see that although our model performs poorly than the case where we train and test on same language scripts, it is still giving good results.

Chapter 5

Conclusion

In this thesis, we thus explore a novel technique to capture the spatial-temporal information present in an online handwritten trace. Majority of the existing approaches in this domain use only spatial descriptors to classify documents without considering the temporal information. The novelty of our approach was to use a spatial descriptor and then model the temporal information using various deep learning-based sequence modelling techniques. The spatial descriptors also use the pressure readings obtained using the recording instruments thus giving more discriminative power to our features. This is because we all know that the pressure of the writing encapsulates a lot of information about the writer. Thus we feel that the temporal variations of pressure are an important feature which was not explored in any of the previous works in this domain.

The main challenge faced while working on this system was the dearth of data. Neural networks need a lot of data to generalize well when we train the neural networks through random initialization of its parameters. The amount of data needed also depends on the complexity of the problem at hand. Since, the problem of classification becomes more complex with a large number of classes, the amount of data needed is not sufficient. In order to avoid the process of tuning the large number of parameters in the final softmax layer satisfactorily we do not use an end-to-end deep learning based framework. Through our experiments, presented in Section 4.2.5, we present that our Supervised LSTM Autoencoder based classification framework outperforms the end-to-end LSTM network based approach.

The ability of the LSTM autoencoder to learn discriminative representations is reinforced by the use of the Supervised LSTM Autoencoder. The training of the LSTM autoencoder in a framework where it jointly predicts the true labels and tries to reconstruct the input sequences

strengthens the ability of the LSTM autoencoder to learn more discriminative representations while capturing the temporal information present in the sequence of sub-stroke descriptors.

We also develop a novel descriptor which is more simple and intuitive as it is inspired by simple concepts in physics. The total force on a point consists of two components, the force magnitude (f_i) and the force angle (a_i). The angle space [0, 2π] is divided into uniform sized bins and we then construct a histogram where the bin is determined based on the angle a_i and it is assigned the corresponding vote of magnitude f_i. This histogram based descriptors is very intuitive and is more easy to calculate and does not require any complex encoding mechanism unlike the PLGF-based features. Moreover, the proposal is the first of its kind to investigate the utility of LSTM autoencoders in the field of online writer identification.

The present state-of-the art on the BIT CASIA database is 80.85% at the text-line level as reported in [11]. We thus report an improvement on the text-line level performance where our method has obtained an accuracy of 81.75%.

Our work thus leaves scope for future explorations. More sophisticated LSTM Autoencoder structures could be used to capture the temporal information in a better way. The use of both online and offline features could be explored during the development of spatial features. The use of various data augmentation techniques could assist to help improve the performance of deep learning frameworks. We believe that the power of our developed descriptors and sequence modelling techniques will give better results when we scale the system to real life scenarios.

Bibliography

[1] Thumwarin P., Matsuura T. “On-line writer recognition for Thai based on velocity of barycenter of pen-point movement,” *IEEE International Conference on Image Processing*, vol. Singapore, pp. 889–892, , Oct. 2004.

[2] Tsai L. M. Y. (2005) ”Online writer identification using the point distribution model”, in International Conference on System, Man and Cybernetics, vol. 2, pp. 1264–1268.

[3] Kameya, Hiroshi et al. “Figure-based writer verification by matching between an arbitrary part of registered sequence and an input sequence extracted from on-line handwritten figures.” Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings. (2003): 985-989.

[4] Schlapbach, A., Liwicki, M., Bunke, H. (2008). A writer identification system for on-line whiteboard data. Pattern Recognition, 41, 2381-2397.

[5] G. X. Tan, C. Viard-Gaudin, and A. C. Kot, “A stochastic nearest neighbor character prototype approach for online writer identification,” in Pattern Recognition, 2008. 19th International Conference on, 2008, pp. 1–4

[6] Nakamura, Y., Kidode, M. (2005). Individuality analysis of online Kanji handwriting. Eighth International Conference on Document Analysis and Recognition (ICDAR’05), 620-624 Vol. 2.

[7] Zhang, X., Xie, G., Liu, C., Bengio, Y. (2017). End-to-End Online Writer Identification With Recurrent Neural Network. IEEE Transactions on Human-Machine Systems, 47, 285-292.

[8] Bhattacharjee, D., Roy, H. (2019). Pattern of Local Gravitational Force(PLGF): A novel Local Image Descriptor. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[9] Kingma, D.P., Ba, J. (2014). Adam: A Method for Stochastic Optimization. CoRR, abs/1412.6980.

[10] BIT handwriting database. (2007). [Online]. Available: http://biometrics.idealtest.org

[11] V. Venugopal and S. Sundaram, "Online Writer Identification With Sparse Coding-Based Descriptors," in IEEE Transactions on Information Forensics and Security, vol. 13, no. 10, pp. 2538-2552, Oct. 2018. doi: 10.1109/TIFS.2018.2823276

[12] Li, Bangy and Sun, Zhenan and Tan, Tieniu, "Hierarchical shape primitive features for online text-independent writer identification," in Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on. IEEE, 2009, pp. 986–990

[13] 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. IEEE, 2009, pp. 931–935.

[14] L. Le, "Supervised autoencoders : Improving generalization performance with unsupervised regularizers," 2018

[15] Dwivedi, Isht et al. "Online Writer Identification Using Sparse Coding and Histogram Based Descriptors." 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR) (2016): 572-577.

[16] G. Singh and S. Sundaram, "A subtractive clustering scheme for text-independent online writer identification,"2015 13th International Conf. on Document Analysis and Recognit. (ICDAR), pp. 311–315, 2015.

[17] Dalal, Navneet and Bill Triggs. "Histograms of oriented gradients for human detection." 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) 1 (2005): 886-893 vol. 1.

[18] A. Shivram, C. Ramaiah, and V. Govindaraju, "A hierarchical bayesian approach to online writer identification,"Biometrics, IET, vol. 2, 2013.

[19] T. Dhieb, W. Ouarda, H. Boubaker, M. Ben Halima, and A. M. Alimi,"Online arabic writer identification based on beta-elliptic model," in2015 15th International Conference on Intelligent Systems Design and Applications (ISDA), pp. 74–79, Dec 2015

[20] V. Venugopal and S. Sundaram, "Modified sparse representation classification framework for online writer identification," IEEE Transactions on Systems, Man, and Cybernetics: Systems, pp. 1–12, 2018.

[21] S. Sundaram and V. Venugopal, "An online writer identification system using adaptive sparse representation framework,"IET Biometrics, 2020.

[22] V. Venugopal and S. Sundaram, “An improved online writer identification framework using codebook descriptors,” Pattern Recognit., 2018

[23] S. Hochreiter and J. Schmidhuber, “Long short-term memory,”Neuralcomputation, vol. 9, pp. 1735–80, 12 1997.