Exploration of Novel Descriptors for Online Writer Identification

A THESIS

submitted for the award of the degree of

DOCTOR OF PHILOSOPHY

By

Vivek Venugopal

DEPARTMENT OF ELECTRONICS AND ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

GUWAHATI - 781 039, INDIA

December 2018

To

My guide **Dr. Suresh Sundaram**

for his guidance and inspiration

&

My **parents and grandparents**

for their blessings, love and support

Certificate

This is to certify that the thesis entitled "**Exploration of novel descriptors for online writer identification**", submitted by **Vivek Venugopal**, a research scholar in the *Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati*, for the award of the degree of **Doctor of Philosophy**, is a record of an original research work carried out by him under my supervision and guidance. The thesis has fulfilled all requirements as per the regulations of the institute and has reached the standard needed for submission. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Dated:

Guwahati.

Dr. Suresh Sundaram
Associate Professor
Dept. of Electronics and Electrical Engg.
Indian Institute of Technology Guwahati
Guwahati - 781 039, Assam, India.

Acknowledgements

The journey of PhD is a long and difficult path and I am deeply indebted to a lot of people for helping me complete this voyage. To begin with, I would like to express my sincere gratitude and appreciation to my supervisor Dr. Suresh Sundaram for his academic guidance and constant encouragement through out the course of my PhD. I'm also highly grateful to him for the efforts he has put in improving the quality of all my manuscripts as well as the thesis. My discussions with him have been insightful and his inputs had definitely improved my research work. The many skills I have learnt from him has provided me a blueprint on how to be an adviser in future.

I would also like to acknowledge my doctoral committee members Prof. P. K. Bora, Prof. S. R. M. Prasanna, and Dr. Prithwijit Guha for their support and inputs to my dissertation. Special thanks to Prof. S.K. Bose and Dr. Amit Sethi for their suggestions in ameliorating the readability of my manuscripts.

I also owe my gratitude to the members of the Multimedia Analytics Lab (Mathew, Dr. Banri, Shikha, Suman, Surbhi, Pranay, Raghvendra, and others) for being there during the crests and troughs of my PhD. My sincere gratitude to Neil, Yogesh, Isht, Swapnil, and Surbhi for the academic collaborations that I had with them as a part of their BTP/MTP. I also acknowledge Dr. Haris B.C and Dr. Sandeep P for their academic inputs during my PhD. I would like to express my thanks to the office staff of the Department of Electronics and Electrical Engineering, IIT Guwahati for all the help offered whenever asked. I also extend my thanks to all my fellow research scholars in the department for their care and support.

I have been fortunate to be surrounded by some great friends in Kaushik, Sarmila, Vijith, Mathew, Sisir, Alex, Parismita, Satishraj, Dr. Jitendra, Sreeram, Uddipana, Kamakshi, Trusna, Niladri and Gargi who have been a pillar of emotional strength throughout my PhD. I am extremely thankful to the Malayalee community at IIT Guwahati for making this place a home away from home for me. In particular, I would like to thank the members of Kameng Ghadakam for their unparalleled support and for all the enriching experiences through the post

dinner walks and trips.

Kapil Dev once said, *"Apart from education, you need good health and for that you need to play sports"*. I had the opportunity to play three different sports at IIT Guwahati namely Badminton, Cricket, and Football. During the course, I met a lot of amazing people in Gaurav, Durga, Srinu, Satya anna, Dr. Charudatt , Dr. Sisir Kumar Nayak, Dr. Suresh Kartha, Prof. Ajay Kalamdhad, Robin, Prof. M.K. Bhuyan, Jitu, Karuna, Tejas, and Mirzaul. Thank you all for the wonderful memories and the efforts to make me physically and mentally fit.

Last but not least, I doff my hat to my parents and grandparents. Without their love, support and sacrifice it wouldn't have been possible for me to complete my PhD. I also thank them for having provided me with the freedom to take this journey and for their wise counsel during its course.

Vivek Venugopal

Abstract

A number of researchers have been attracted to the idea of utilizing handwriting of an individual as a behavioural biometric. Thanks to technological advances, it is possible now to obtain handwriting information of an individual through hand held devices such as tablets. These comprise an electronic pen/stylus, the tip of which helps in capturing the dynamic information present in the handwritten trace. 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. The present thesis focusses on proposing novel descriptors for a text-independent online writer identification system.

In the first part of the thesis, we represent the spatio temporal trace from a document with descriptors derived from a codebook. To begin with, we adapt the Vector of Locally Aggregated Descriptor (VLAD) approach, popular in the field of image retrieval, for online writer identification. We demonstrate that the formulation of VLAD, at times, presents an issue of reduced discrimination between writers. As an alleviation to this drawback, we propose an improved descriptor, that relies on scores of each of the attributes in a feature vector with regards of the proximity to their corresponding values in the assigned codevector. Our approach is formulated in such a way that for a codebook consisting of M codevectors, we concatenate the descriptors from $M-1$ codevectors to obtain the document descriptor.

The codebook used for formulating the codebook descriptors in our first proposal is generated by applying the k-means algorithm. This however comes with a limitation that each feature vector gets assigned to only one prototype. In order to address this, we consider proposing the descriptors with respect to an over-complete

dictionary obtained via a sparse representation framework. Such a strategy ensures that the feature vectors are expressed as a linear combination of atoms that serve as prototypes. Moreover, traditional sparse representation based classification schemes in writer identification consider the pooling of the sparse coefficients by using either of the maximum or average operation. Hence, in the subsequent two parts of the thesis, we derive descriptors that aim to capture additional information from the sparse coefficients, that in a way, surpass the performance of the max /average pooled strategies.

In the second part of the thesis, we propose descriptors constructed from a set of dictionary atoms obtained in a sparse representation framework to characterize the segmented sub-strokes of the handwritten trace. These descriptors attempt to capture the similarity of the attributes of each feature vector with the corresponding value in the subset of dictionary atoms which possess a non-zero sparse coding coefficient.

In the third part of the thesis, we propose descriptors that modify the max /average pooled strategies by incorporating prior knowledge that is learnt from the set of dictionary atoms. More specifically, we associate each of the dictionary atoms with a 'saliency' value, that quantify the degree of its importance with regards to the dynamic characteristics of the enrolled writers. For the computation of the same, we describe two methods that are primarily based on analysis of histograms. Added to these contributions, we also propose an identification framework that makes use of saliency values that are adapted to each writer enrolled in the system.

The efficacy of all proposed descriptors of the thesis are demonstrated on the IAM and IBM-UB1 databases. The results obtained are found to be promising when compared to prior works.

Contents

List of Figures

List of Tables

List of Acronyms

IR	Average Identification Rate
VLAD	Vector of Locally Aggregated Descriptor
SVM	Support Vector Machine
RBF	Radial Basis Function
SPF	Traditional sparse representation system
EN-SL	Modified sparse representation framework with the incorporation of saliency obtained from the entropy approach.
SP-SL	Modified sparse representation framework with the incorporation of saliency obtained from the sum-pooling approach.
SAL-ADP	System where the modified writer descriptor is computed by incorporating writer-specific saliency values to the descriptor obtained via traditional pooling strategies.
SEG-1	Method of splitting the online trace at the local maxima in a stroke to obtain substrokes
SEG-2	Method of generating sub-strokes with a fixed number of points
SPAMS	SPArse Modeling Software
GMM-UBM	Gaussian Mixture Model-Universal Background Model
tf	term frequency
idf	inverse document frequency
LDA	Latent Dirichlet Allocation
RHS	Random Hybrid Strokes
SIFT	Scale Invariant Feature Transform

List of Symbols

$\boldsymbol{\alpha}$	Sparse coefficient vector corresponding to a feature vector $\mathbf{f}$
α_{ij}^{k}	Sparse coefficient value associated with the i^{th} dictionary atom as obtained from the j^{th} sub-stroke of the k^{th} writer
$\mathbf{a_i}$	Acceleration vector at sample point $\mathbf{p}_i$
B	Number of bins for the histogram based features
C	Regularization parameter used in SVM training
$\Delta(\mathbf{p}_i, \mathbf{p}_{i+r})$	Euclidean distance between sample points $\mathbf{p}_i$ and $\mathbf{p}_{i+r}$
$\mathbf{\Delta_i}$	Delta vector at sample point $\mathbf{p}_i$
D	Dimension of feature vector
$\mathbf{D}$	Over-complete dictionary of M atoms.
$d_{\mathbf{\Delta_i}}$	Magnitude of the delta vector $\mathbf{\Delta_i}$
$d_{\mathbf{a_i}}$	Magnitude of the acceleration vector $\mathbf{a_i}$
$d_{\mathbf{p}_i}$	Euclidean distance between $\mathbf{p}_i$ and the centroid $\mathbf{p}_c$ of the sub-stroke S
$d_1(w_1)$	Average pair-wise intra-writer distance between final descriptors belonging to documents written by writer w_1
$d_2(w_1)$	Average pair-wise inter-writer distance between final descriptors belonging to documents written by writer w_1 and all other writers
$e_{ij}(d)$	Reconstruction error computed from the d^{th} attribute of the j^{th} feature vector by utilizing the i^{th} dictionary atom
$\mathbf{f}$	Feature vector extracted at a sample point/sub-stroke
γ	Inverse of the standard deviation of the RBF kernel for SVM
H	Entropy measure

$\mathcal{H}$	Histogram
$\mathbf{h_p}$	Histogram constructed by considering $d_{\mathbf{p}_i}$ and the angle made by the line segment joining $\mathbf{p}_i$ and the centroid $\mathbf{p}_c$ of the sub-stroke S with the horizontal
$\mathbf{h_\Delta}$	Histogram constructed by utilising the delta vectors
$\mathbf{h_a}$	Histogram constructed by making use of the acceleration vectors
h_i	The votes obtained for the i^{th} bin of histogram $\mathcal{H}$
$\mathcal{I}()$	Indicator variable
M	Number of codevectors / atoms.
$\boldsymbol{\mu_i}$	i^{th} codevector in the codebook
$\mathbb{N}_\mathbf{p}$	Neighbourhood around sample point $\mathbf{p}$
n_i	Number of feature vectors assigned to the i^{th} codevector
N_S	Number of points in sub-stroke S
N_T	Number of feature vectors in a document.
N_{Tr}	Total number of sub-strokes from all the writers whose data is enrolled for training
p	Probability measure
$\mathbf{p}$	Sample point in online trace
$\mathbf{p}_c$	Centroid of a substroke S
ϕ_i	Curvature at sample point $\mathbf{p}_i$
$\boldsymbol{\phi}_i$	i^{th} dictionary atom
r	Gap parameter
R	Average inter-writer to the intra-writer distance ratio
S	Sub-stroke
$\mathbf{S}_i$	Descriptor corresponding to the i^{th} codevector / dictionary atom
$\mathbf{S}$	Final descriptor for a document
$S_{ij}^{+}(d)$	Score obtained for the d^{th} attribute of the j^{th} feature vector assigned to the i^{th} codevector/atom
$S_{ij}^{-}(d)$	Score obtained for the d^{th} attribute of the j^{th} feature vector assigned to

	the i^{th} codevector/atom
$\tilde{S}_i^+(d)$	Normalized score for the d^{th} attribute corresponding to feature vectors assigned to the i^{th} codevector/atom
$\tilde{S}_i^-(d)$	Normalized score for the d^{th} attribute corresponding to feature vectors assigned to the i^{th} codevector/atom
$\mathbb{S}_k$	Set of feature vectors of sub-strokes segmented from training samples of the k^{th} writer
$\lvert\mathbb{S}_k\rvert$	Number of feature vectors of all sub-strokes segmented from training samples of the k^{th} writer
$\mathbf{S}_i^+$	Descriptor obtained for the i^{th} atom computed by stacking the $\tilde{S}_i^+(d)$ scores across the D attributes
$\mathbf{S}_i^-$	Descriptor obtained for the i^{th} atom computed by stacking the $\tilde{S}_i^-(d)$ scores across the D attributes
s_{ij}^k	Similarity score computed for the j^{th} substroke of the k^{th} writer with regards to the i^{th} dictionary atom
$\bar{s}_{ik}^{norm}$	Normalized similarity score corresponding to the k^{th} writer with regards to the i^{th} dictionary atom
t_i	Time stamp at a sample point $\mathbf{p}_i$
θ_i	Writing direction angle at a sample point $\mathbf{p}_i$
$\theta_{\mathbf{p}_i}$	Angle that the line segment $\mathbf{p}_i - \mathbf{p}_c$ makes with the horizontal
$\theta_{\mathbf{\Delta_i}}$	Orientation of the delta vector $\mathbf{\Delta_i}$
$\theta_{\mathbf{a_i}}$	Orientation of the acceleration vector $\mathbf{a_i}$
v_i	Velocity at a sample point $\mathbf{p}_i$
W	Number of writers in the database
w_i	Saliency value corresponding to the i^{th} dictionary atom
z_i	Writer descriptor obtained for the i^{th} dictionary atom

1

Introduction

Contents

1.1 Introduction

In the recent decade, the concerns of security have led to the necessity of employing technology to identify a person. Person identification through the medium of biometric systems refers to the recognition of individuals by their unique characteristics or traits [2]. A physical or behavioural trait that exhibits the properties of universality, distinctiveness, permanence, acceptability and accessibility may be regarded as a candidate biometric [3].

A survey of works from literature indicate that the biometric systems, in general, can be categorized to either physiological or behavioural [4]. In the former, direct measurements of a part of the human body are used for identification. Examples of such systems include those of fingerprint [5], face [6], iris [7] and hand-scan [8], to mention a few. A behavioural biometric, on the other hand, relies on features that are obtained from data captured from an action made by a user. Some of the systems that belong to this category are those of signature [9], keystroke dynamics [10] and gait [11].

Without loss of generality, the pipeline of steps followed in the design of a biometric system bears semblance to that of a pattern classification framework. To begin with, we require data to be captured from an individual for establishing his / her identity. This is followed by utilizing an appropriate technique to derive discriminative or salient features from the captured data. Subsequent to this, the matching is performed by contrasting the extracted feature sets to those saved in the database. Based on the outcome of the comparison, a decision is made on the authenticity / identity of the data.

Further to the above, it is worth emphasizing that a biometric system can be operated in either of the two modes, namely identification or verification.

Verification mode: In this framework, the biometric system verifies the claim of an individual by comparing his / her captured data to their corresponding templates pre-learnt and saved in the database. Such systems are typically regarded as a one class problem, where the positive samples correspond to data enrolled from the user in question. The decision at the output is to either accept or reject the claim.

Identification mode: In this set-up, the system assigns the identity to the data by matching it against templates of all users stored in the database. This entails a one to many comparison to be made, which is typically accomplished by classifiers in a multi-category setting.

The crux of the present dissertation is in the exploration of strategies for establishing the identity of the user with his / her handwriting.

1.2 Overview of writer identification systems

As a behavioural biometric, the goal of writer identification is to determine the authorship of a handwritten document by contrasting it against a set of enrolled samples with known authorship stored in a database. This area of research falls under the broader field of automatic handwriting recognition. However, there does exist a fine distinction between the two domains. The main focus of handwriting recognition is to rely on invariant representations of the handwritten content, typically achieved by reducing the extent of inter-writer variations. Contrast to this, works on writer identification exploit the writer-specific variations for discriminating the characteristics between the writers [12, 13].

Some of the potential areas of application of writer identification include those of forensics, security to intelligent, adaptive systems and digital libraries [13]. The techniques using statistical models provide a scientific insight for forensic document analysis [14]. In the context of digital libraries, writer identification based systems are useful for indexing and retrieving historical handwritten documents. Another interesting application worth mentioning is that of intelligent environments such as smart meeting rooms, where such systems may be used to annotate the handwritten notes with the writers identity [15, 16]. This information in turn can aid in cross-validating the results obtained from other modalities such as video and/or audio. Added to this, writer identification systems also find utility in automatically generating tags and meta-data in the pipeline of indexing, searching and retrieval.

On the basis of data acquisition, writer identification systems can be classified into either offline and online.

Offline: In offline writer identification, the handwritten patterns are captured on paper, scanned using scanners and subsequently saved in form of digital images [17], [18], [19], [20], [21]. The scanned images provide a passive form of handwritten information that primarily gives the spatial information in the form of image pixels. Accordingly, such systems make use of techniques from the area of image processing for subsequent analysis.

Research works in the literature of offline writer identification suggest that they belong to one of the two approaches, namely textural based and allograph based methods. In the former group, global description from the handwritten image data are utilized for writer identification such as computation of ink width [22] and attributes like Local Binary Pattern [19], Co-occurrence features [23] and Edge structure code [24] to name a few. Conversely, the allograph based techniques consider the feature description computed from the letter parts (allographs). While establishing the authorship of a test document, these entail the use of a dictionary / vocabulary that are pre-learnt during the time of training [21, 25, 26].

Online: The recent trend in technology has enabled the manufacture of hand-held devices that provide a pen based input interface to capture the handwritten data [27]. These primarily make use of an electronic stylus pen to record the data on a pressure-sensitive screen. The pen keeps track of the trajectory information of the handwriting, such as the (x, y) spatial location and time stamp. In the literature of handwriting analysis, the processing of such data is referred to as 'online'. Without loss of generality, the input to such a system is a handwritten document consisting of a sequence of strokes, which in turn contains a sequence of points.

Over the recent years there has been a steady increase in the use of such hand-held devices (Refer Fig. 1.1) which in turn has inspired researchers to evaluate the utility of online handwriting data as a biometric. Added to this, a vast majority of works in the literature of writer identification pertain to an offline setting. Said in another way, there has been limited explorations being made in the domain of online writer identification. This provides us the motivation to pursue

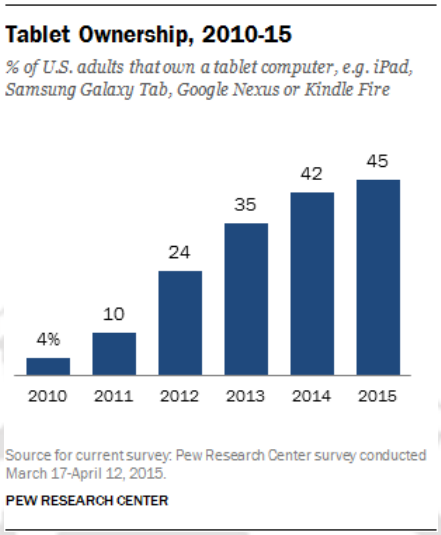


Figure 1.1: Tablet ownership (in %) among US adults 2010-15. Adopted from [1]

research directions in this field.

Another categorization of writer identification system is with respect to the textual content - namely text dependent and text independent approaches [28]. In the former, a specific piece of text is used for the generation of handwriting samples of a writer and the identification process usually involves the use of a recognizer. Though the use of the knowledge pertaining to the content of the data increases the accuracy of text dependent systems, they fail in scenarios where text documents comprising different contents need to be contrasted. For such applications, text independent writer identification systems become more applicable as they capture the style information of handwriting. Such systems are designed to identify the writer irrespective of the textual content.

1.3 Previous works on online writer identification

Over the last decade, there has been considerable research in the area of text-independent online writer identification under the category of allograph-based approaches [1]. One of the prominent works is that of the Gaussian Mixture Model-Universal Background Model (GMM-UBM), developed in the context of research on smart room meeting room systems [16, 29–31].

[1]It may be noted here that textural based methods rely on the bitmap image and hence are not applicable to online writer identification systems.

The idea is to first construct the UBM from the training data corresponding to a set of enrolled writers with the Expectation Maximization algorithm. Thereafter, the adaptation step is considered wherein individual GMMs are obtained for each writer by using only his / her training data. During the testing phase, the document is assigned to the writer, whose corresponding GMM provides the highest log-likelihood score. It is worth emphasizing that the GMM-UBM approach for online writer identification has been inspired, primarily owing to its success in the area of speaker recognition [32].

The development of online writer identification systems have also been motivated from the domain of information retrieval [33, 34]. In these works, an industrial handwriting recognition system is used to perform an automatic segmentation of an online handwritten document at the character level. Different prototypes of the same character are generated by applying the traditional k-means clustering on feature vectors extracted from the online trace of the segments. Following this, a fuzzy c-means approach is employed to estimate their statistical distributions on an alphabet basis. With respect to generating the distribution, the concept of term frequencies (tf) and inverse document frequencies (idf) from the field of information retrieval [35] is proposed. Finally, the writer identity is established by comparing the distribution of tf-idf vector of the test document with those of the reference documents with the minimum distance classifier.

The preliminary exploration leading to the proposal discussed above can be found in [36,37]. In the first work, the authors represent the distribution of character prototypes using the tf vector alone for writer identification. Said in another way, the idf vector was not incorporated in this system. On a similar note, the fuzzy c-means algorithm was not investigated in the paper [37]. This, in turn resulted in assigning each segmented character to only one particular prototype, while building the distribution of handwriting styles.

Further to the work [34], the same authors in a subsequent publication [38] demonstrate that the alphabet knowledge inherent in character prototypes can provide additional writer cues with regards to their styles of writing and their identities. More specifically, an alphabet information coefficient is proposed during the scoring of the handwritten document, that in

turn results in improving the performance of the identification system.

Moving to other explorations, the authors in [39] propose an approach based on dynamic features related to the movements during writing. The extracted feature vectors pertain to the writing pressure, velocity and pen inclination obtained from the trace of twelve primary strokes pre-identified for the Chinese script. Each type of stroke segment is modelled by four Gaussian models - with the resulting mean and variance of each of them employed as templates during identification.

As an improvement to the above proposal, the description of the stroke types are represented in [40] by a set of histograms / probability distribution functions. The four dynamic attributes being considered for histogram generation are pressure, azimuth, velocity and altitude respectively. In addition, the authors propose a weighted sum rule for combining the matching scores obtained from each attribute. However, a potential drawback of both the works [39, 40] is that the stroke types being selected may not generalize well to non-Chinese scripts.

A hierarchical two-stage approach based on shape primitives is employed in [41] for online writer identification of Chinese and English texts. In the first stage, the statistics of the orientation and the pressure information in a shape primitive (captured as a probability distribution) are employed to reduce the list of probable writers. In the second stage, the curvature of the shape primitive of the selected writers is quantized to a fixed number of bins. Thereafter, the mean and variance of four dynamic features (pressure, azimuth, altitude and velocity) in each bin is employed for identification.

The utility of employing temporal sequence and shape codes to encode online handwriting has been studied in [42]. Two temporal sequence codes, namely the stroke temporal sequence code and neighbour temporal sequence code are employed to characterize the trajectory with regards to speed and pressure respectively. The shape codes, on the other hand, serve to provide information about the direction of the trace. This method of encoding, in particular, was found to be effective for cross-language (English and Chinese) writer identification. The idea behind using sequence codes has been inspired primarily from the descriptor - Local Binary Pattern used in image processing [43].

In the work [44], the authors employ an unsupervised learning scheme called subtractive clustering. This algorithm locates the high density regions in the feature space to capture the frequent writing styles / prototypes of the author. Different to the k-means and fuzzy c-means, it does not rely on the initial choice of seed points. The discovered prototypes are subsequently employed to score the authorship of an unknown handwritten text.

In the work [45], the idea of multi-fractals are used to model the segmented graphemes / sub-strokes. Subsequent to it, the authors utilize a weighting based on tf-idf framework. The scoring of the tf and idf term is based on a frequentist approach, that in a way, characterises the number of segmented grapheme patterns assigned to a given codevector in a codebook. In addition, based on the grouping of the graphemes for Persian / Arabic script, separate codebooks are constructed for each of them.

A framework that follows a global approach based on utilizing the words as primitives is investigated in [46]. Different sets of statistic and dynamic features are extracted from the stroke, the space between strokes and the whole word. In the decision phase, the scores from the Dynamic Time Warping algorithm and Support Vector Machine (SVM) classifier are used. Furthermore, the influence of the number of writers as well as the number of words per writer on the performance of the system is also studied.

An identification system using the Bayesian methodology is proposed in [28], wherein the authors first identify the consistent primitives of the handwriting by using the clustering technique of k-means. Thereafter, the between-writer variations within consistent primitives are exploited to determine the authorship of the handwritten text. This is accomplished by employing classifiers for each of the consistent primitives (clusters) and then combining the obtained results to retrieve the most likely writer. The classifiers are designed using the labelled training samples of writers that falls in each of the clusters. Through experiments in this work, the authors demonstrate the utility of their approach across five different scripts.

A set of static and dynamic features adapted for embedded writer identification systems with restricted memory and processing power is proposed in [47]. This set is reduced to a subset by employing a feature selection approach. The resulting optimal number of features is then

passed through a minimum distance classifier, Bayes classifier and their serial combination. The same authors in a subsequent work [48] realize a method for dynamic writer identification by exploiting the relation between static and dynamic information in a handwritten text. Here, the correlation between the length and the direction of the handwritten segments between sample points are used, together with pressure, altitude and azimuth.

A point distribution model is proposed for obtaining features (using projection) to discriminate between the writers [49]. The objective of such a model is to learn the eigen-structure for individual writers that, in a way captures their unique writing styles. During the identification step, the sum of strengths of major eigen-modes (resulting from the projection) is employed as a similarity metric.

The authors in [50–52] explore the utility of Beta-elliptic model for online writer identification. This model provides a description that takes into consideration, the aspects of velocity and trace of the handwriting trajectory. The efficacy of the obtained features are tested on online Arabic words. On a similar note, in the system proposed in [53, 54], a multi-fractal representation obtained from offline images of handwritten words is used in conjunction with the online trace.

In the works [13, 55], the authors assume that the writing styles of users are a shared component of an individual's handwriting. Keeping this view, a theoretical framework for the same is proposed by utilizing the idea of Latent Dirichlet Allocation (LDA), popular in the area of topic models. In a sense, each writer's data is modelled as a distribution over finite styles that are shared between the different writers. In a following exploration [56, 57], the same authors extend the LDA to bring out a novel perspective for writer identification by introducing the concept of accents. These refer to the distinctive writing characteristics unique to a group of people, who share a common native script. With regards to the proposed methodology, the accent of the writer is predicted first. Thereafter, based on the selected accent, the authorship of the handwritten document is determined. This approach reduces the complexity of the writer identification task post accent-prediction step.

An empirical study on the the amount of data required to build the underlying models in the

design of a writer identification system is presented in [58]. Here, the authors systematically examine data sufficiency bounds for the approaches of feature space models and writer-style space models respectively. Based on extensive experimentation, they provide evidence to demonstrate that the writer-style space model gives higher identification performance with lesser data costs, as compared to those of feature space models.

Of late, deep learning based approaches have recently captured the interest of the online writer identification research community. These methods learn the feature representation from the handwritten data, thus alleviating the need of hand-crafted features. A Convolutional Neural Network based approach was presented in [59,60] where the authors employ a drop segment method (influenced from the drop-out training strategy of [61]) to improve the generalization capability. Further to this, for sufficient feature representation, a path signature feature map is constructed from the pen-position information along the online trace of the handwriting.

Further to the above works, a Recurrent Neural Network model with bi-directional Long Short Term Memory based approach has also been proposed in the work [62]. In this research, the authors represent the handwritten data of a given writer by a set of Random Hybrid Strokes (RHSs). Each RHS corresponds to a randomly sampled short sequence comprising the pen tip movements. These are then converted to a fixed-length vector for final classification by the deep-learning framework. The posterior probabilities computed across the set of RHSs are finally averaged to establish the identity of the writer.

For sake of convenience, we also summarize in Table 1.1, the aforementioned explorations in the literature of online writer identification.

1.4 Contributions of the thesis

In this thesis, we propose a number of novel writer descriptors for text independent online writer identification systems. Different to previous explorations, our focus is in utilizing additional information from a pre-learnt codebook or sparse dictionary, that can provide cues in deriving these descriptors. Our different proposals have been laid out in three main contributing chapters and we elaborate on them in the following subsections.

Author	Approach	Language
Schlapbach et.al [16, 29–31]	GMM-UBM	English
Tan et.al [33, 34, 36–38]	Character prototypes	English
Yu et.al [39],Li et.al [40]	Histogram of dynamic features	Chinese
Li et.al [41]	Shape primitive and hierarchical classification	Chinese and English
Li et.al [42]	Temporal sequence and shape codes	Chinese and English
Gautam Singh and Suresh Sundaram [44]	Subtractive clustering	English
Chaabouni et.al [45, 53, 54]	Multi-fractals	Arabic/Persian
Anoop Namboodiri and Sachin Gupta [28]	Shape-based features+Bayesian methodology	Devanagari, Roman, Cyrillic, Arabic and Hebrew
Gargouri et.al [46]	Stroke based features+DTW+SVM	Arabic
Ming-Yen Tsai and Leu-Shing Lan [49]	Point distribution model	Chinese
Dhieb et.al [50–52]	Beta-elliptic model	Arabic
Shivram et.al [13, 55–58]	Latent Dirichlet Allocation	English
Zhang et.al [62]	Recurrent Neural Network	Chinese and English

Table 1.1: Summary of online writer identification systems in literature

1.4.1 Chapter 2: Exploration of codebook descriptors

In this part of the thesis, we derive a strategy that encodes the sequence of feature vectors extracted at sample points of the temporal trace with descriptors obtained from a codebook. These descriptors aim to capture the relative location of the feature vectors corresponding to the handwriting samples of a writer with respect to their nearest codevector in the feature space. The idea of the same comes from the intuition that the relative location of feature vectors of the same writer are more or less aligned in near-by proximity in the feature space. However, such a trend may not be prevalent when considering the feature vectors across different writers.

We begin by investigating on the merit of the Vector of Locally Aggregated Descriptor (VLAD) approach, a codebook based descriptor popularly used in the field of image retrieval [63], for online writer identification. However, we demonstrate that at times, the formulation of the VLAD is confronted with an issue, that can reduce the discrimination between writers. This can occur when the point based feature vectors corresponding to the different writers are not linearly separable in the Voronoi cell of a codevector - thereby affecting the identification rate. To alleviate this drawback, we propose a set of descriptors with a modified formulation aimed at improving writer discrimination beyond VLAD.

In our proposal, we first assign each feature vector belonging to a document to a specific codevector based on distance criterion. The proposed descriptors take into consideration, the scores of each of the attributes in a feature vector with regards of the proximity to their corresponding value in the assigned codevector. We show in this Chapter that such explicit scoring can provide useful cues for better discriminating handwritten samples of the different writers enrolled to the system, when compared to the VLAD. We formulate our strategy in a way that, for a given codebook size M, we employ the descriptors of only $M-1$ codevectors to construct the final descriptor by concatenation.

The performance of the VLAD and our proposed descriptor are evaluated on two publicly available online handwriting datasets, namely the IAM and IBM-UB1 databases. The results obtained demonstrate an improvement in writer identification rate over the system that uses the conventional VLAD formulation. Moreover, since our devised descriptor has a dimension

twice that of the VLAD, we consider reducing it to half. Despite this reduction, we report improved results.

Apart from the above contribution, for constructing the codebook, and subsequently the descriptor, we derive features by incorporating a gap parameter. This aids in capturing the information from the sample points in the neighbourhood / vicinity of the point under consideration. An empirical study is also conducted on the variation of the writer identification rates for different values of the gap parameter.

1.4.2 Chapter 3: Exploration of sparse coding based descriptors

In the first part of the thesis presented in Chapter 2, the codebook used for formulating the writer descriptor is generated by applying the k-means algorithm. This however comes with a limitation that each feature vector gets assigned to only one prototype. As an alleviation to this issue, we consider proposing the descriptors with respect to an over-complete dictionary obtained via a sparse representation framework. Such a strategy ensures that the feature vectors are expressed as a linear combination of atoms that serve as prototypes. To the best of our knowledge, the use of sparse coding approaches for online writer identification have not been addressed till date in the literature.

The traditional pipeline typically followed in works on sparse coding involve pooling the obtained coefficients with either maximum or average operation. Though these methods do work satisfactorily, we do believe that there can still be room for improvement. In the remaining two Chapters, we demonstrate that exploring additional information from the sparse coefficients can provide us with useful cues that can model the dynamic characteristics of the writer. The incorporation of such information in turn leads to a better performance of the online writer identification system.

Keeping in line with the above idea, in Chapter 3, we move on to proposing a methodology to encode the sub-strokes of the online handwritten trace with descriptors derived from the set of dictionary atoms obtained in a sparse coding framework. The usage of sparse representation stems from the flexibility it offers over the hard clustering algorithms such as k-means in terms

of representing the segmented handwritten sub-stroke of a writer as a combination of more than one prototype / atom. The sub-strokes are segmented from the online trace of the handwritten data in a pre-processing step.

Our work attempts at capturing the similarity of the attributes of each feature vector of a sub-stroke to the corresponding value in the subset of dictionary atoms that contribute with a non-zero sparse coding coefficient. More specifically, we derive descriptors from each of the dictionary atoms, which incorporate the similarity scores of the attributes in a feature vector. The scores are in a way, indicative of the reconstruction error obtained while employing a particular dictionary atom alone for reconstruction. The idea of computing similarity scores for each attribute separately leads to a writer descriptor that captures the dynamic characteristics at a finer level there by leading to performance improvement in the IAM and IBM-UB1 databases.

In addition to the above contribution, we consider several sets of histogram based attributes / features at the sub-stroke level for constructing the dictionary atoms and subsequently the descriptors. We provide a thorough and comprehensive analysis of the histogram based attributes by addressing the issue of the selection of the bin size to be used and their relation to the writer identification rate. More specifically, we propose an entropy based strategy for determining the appropriate bin size to be selected for the generation of these histograms. Considering that the descriptors of the sparse coding framework are constructed from the resulting histogram feature sets, the selection of the bin size using our method ensures a good discrimination between the writers. This also gets reflected in the experimental results, where we show a dependence between the bin size chosen by the entropy based analysis and the corresponding writer identification performance.

1.4.3 Chapter 4: Exploration of saliency information in the sparse framework

In this Chapter, we once again utilize the sparse coding strategy for identifying the authorship of online handwritten documents. The contribution is on the idea of exploring additional information, that is related to quantifying the degree of importance of each of the dictionary

atoms with regards to the dynamic characteristics of the writers. In this context, we define in this work, the term "saliency" [2] for an dictionary atom - the value of which is learnt from the sparse coefficients corresponding to the sub-strokes of the handwritten training data. In this Chapter, we propose two different approaches at obtaining the saliency values.

(i) In the first approach, we make use of the entropy measure computed over histograms generated from the sparse coefficients as a measure of saliency.

(ii) The saliency factor in the second approach is computed as a function of the sum pooled sparse coefficients obtained from the feature vectors corresponding to the enrolled documents of the writers in the database. Our formulation used bears resemblance to the inverse document frequency (idf) popular in the field of information retrieval.

The pre-learnt saliency values are incorporated on the traditional schemes of max and average pooling of sparse codes, thereby resulting in a modified writer descriptor. It has been shown through experiments on the IAM and IBM-UB1 databases that the modified writer descriptor obtains improved performance when compared to the traditional max or average pooled writer descriptor.

Further to the above, we also consider incorporating writer-specific adaptation of saliency values, that quantifies how important a dictionary atom is for a given writer. We employ the reconstruction error on the sub-stroke based feature vectors to derive a similarity score for each dictionary atom with regards to a writer using only his / her handwriting. The obtained scores across all the dictionary atoms are subsequently fused with their respective saliency values to generate adapted values for the purpose of identification. In particular, we formulate an ensemble of SVMs, wherein the descriptor to the SVM trained for a writer is based on the saliency values adapted for that writer. The final decision on the authorship is proposed as the maximum of the prediction score obtained from the SVMs.

[2] In this thesis on online writer identification, this term is not to be interpreted to that used by researchers of the image processing and computer vision community.

1.5 Conclusion

In this Chapter, we first provided an overview of biometric systems. Thereafter, we moved on to a detailed discussion of online writer identification together with an elaborate survey of prior works in the literature. The contributions of each of the working Chapters was also highlighted, thus setting the roadmap to the present thesis.

2

Exploration of codebook based descriptors

Contents

2.1 Introduction

It may be recalled from Section 1.3 that the research ideas being proposed in the literature of online writer identification have been motivated largely from areas of speaker identification [29, 46] and information retrieval [13, 33, 34, 55]. A contemporary area of research in image processing is the problem of object retrieval. Techniques proposed in the recent years have considered the use of descriptors obtained from a codebook - with the Vector of Locally Aggregated Descriptor (VLAD) [1] being considered as a popular technique [63, 64]. Being an image based approach, the codebook comprises a set of code vectors with the corresponding Voronoi cells computed from a clustering algorithm on a set of interest point feature vectors such as Scale Invariant Feature Transform (SIFT) [65]. In this Chapter, we attempt at adapting the framework of codebook descriptors for online writer identification.

In the context of the present work, a codebook comprises a set of codevectors that represent the frequently occurring writing patterns among writers in an average sense [28, 34, 41]. We begin by investigating on the merit of the VLAD approach for online writer identification. Specifically, we replace the SIFT feature vectors with the set of point based feature vectors along the online trace of the handwritten data. However, we demonstrate that at times, the formulation of the VLAD is confronted with an issue, that can reduce the discrimination between writers. This can occur when the point based feature vectors corresponding to the different writers are not linearly separable in the Voronoi cell of a codevector - thereby affecting the identification rate. To alleviate this drawback, we propose a set of descriptors with a modified formulation aimed at improving writer discrimination beyond VLAD.

Without loss of generality, the codebook descriptors (including the VLAD and our modified proposal) are aimed to capture the relative location of the feature vectors corresponding to the handwriting samples of a writer with respect to their nearest codevector in the feature space. The idea of the same comes from the intuition that the relative location of feature vectors of the same writer are more or less aligned in near-by proximity in the feature space. However, such a trend may not be prevalent when considering the feature vectors across different writers.

[1] Further details regarding this method will be outlined in subsection 2.4.1.

As a reiteration to the preceding discussion, we propose in this Chapter, a modified strategy of encoding the sequence of feature vectors along the online trace of a document with descriptors from a codebook. Based on a distance criterion, each feature vector of the document is assigned to a specific codevector in the codebook. As we shall see in Section 2.4.2, the proposed descriptors take into consideration, the scores of each of the attributes [2] in a feature vector with regards of the proximity to their corresponding value in the assigned codevector. We show that such explicit scoring can provide useful cues for better discriminating handwritten samples of the different writers enrolled to the system, when compared to the Vector of Locally Aggregated Descriptor (VLAD).

Apart from the above contribution, for constructing the codebook, and subsequently the descriptor, we derive features / attributes by incorporating a gap parameter. This aids in capturing the information from the sample points in the neighborhood / vicinity of the point under consideration. An empirical study is also conducted on the variation of the writer identification rates for different values of the gap parameter. We show that the use of the gap parameter helps in reducing the degree of skew in the histograms of the feature thus enabling better performance of writer identification system.

The performance of the VLAD and our proposed descriptor are evaluated on two publicly available online handwriting datasets, namely the IAM and IBM-UB1 databases. The results obtained demonstrate an improvement in writer identification rate over the system that uses the conventional VLAD formulation. Moreover, since our devised descriptor has a dimension twice that of the VLAD, we consider reducing it to half. Despite this reduction, we report improved results.

We conclude the Section by enumerating the main research contributions of this Chapter:

- Proposal of a gap parameter for feature extraction.
- Adaptation of the codebook based VLAD framework to the online writer identification

[2] We refer to the feature values in a feature vector as 'attributes'. The attributes computed at each sample point of the online trace are stacked to form a feature vector. Accordingly, elsewhere in the work, we interchangeably use 'features' or 'attributes'.

scenario.

- Demonstration of a potential drawback with the VLAD framework.
- Proposal of a novel descriptor that alleviates the drawback of the VLAD.
- An empirical evaluation of the effectiveness of our proposal with several recent variants of VLAD.
- Exploration of a reduced dimension variant of the descriptor and demonstration of its performance in writer identification.

2.1.1 A block schematic of our proposal

Figure 2.1 presents a block diagram of the proposed writer identification system. The temporal (x, y) sequence from a test document of unknown authorship is first passed through pre-processing module. Thereafter, by incorporating a gap parameter, a set of attributes / features are constructed at each sample point of the online trace in the feature extractor module. The pre-learnt codebook is obtained by applying the k-means algorithm on feature vectors pooled from several samples of handwritten documents of writers enrolled to the system. By employing the codevectors, we compute the proposed descriptor that encapsulates the writer specific characteristic of the handwritten test document. Finally, a SVM classifier is used to retrieve the identity of the writer. The SVM has been trained using the descriptor corresponding to the enrolled handwritten documents of the writers present in the database.

The rest of the Chapter is organized as follows: We discuss the details of the preprocessing and the feature extraction modules in Section 2.2 and 2.3 respectively. The application of codebook descriptors is discussed in 2.4. To start with, the adaptation of the VLAD for writer representation is elaborated at length in sub-section 2.4.1 with its limitation. In order to resolve the issue, we propose a modified codebook descriptor in sub-section 2.4.2. A brief overview of the SVM classifier, that will be used for the experiments in this thesis is presented in Section 2.5. With regards to the results on the different experiments, we outline them systematically in

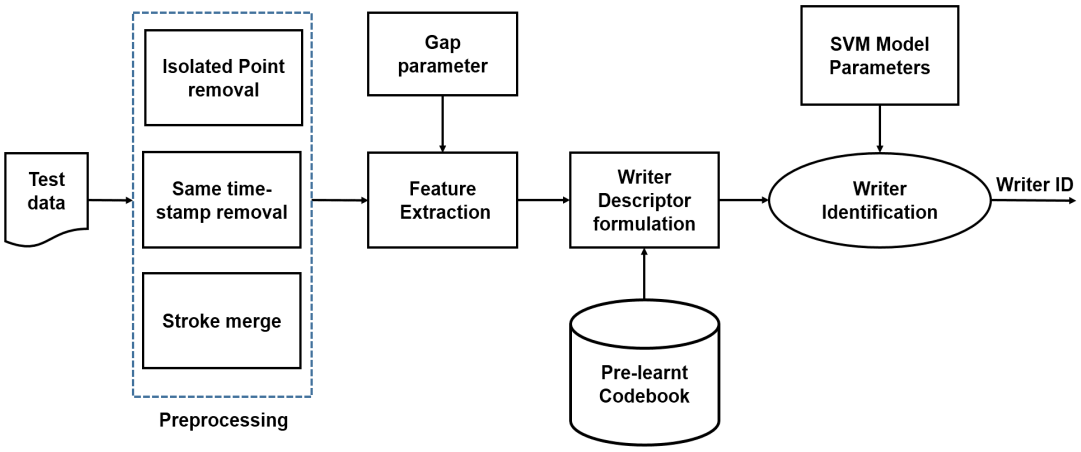


Figure 2.1: Block diagram of the proposed text independent writer identification methodology, that employs the descriptors from a codebook. The test data refer to paragraphs or text lines written by the enrolled writers.

Section 2.7, subsequent to discussing the databases with their respective training and testing protocols in Section 2.6.

2.2 Preprocessing

The preprocessing steps employed in this study are as follows.

- **Isolated point removal:** In order to eliminate such points, we consider a neighbourhood $\mathbb{N}_{\mathbf{p}}$ around a sample point (say $\mathbf{p}$) in the same stroke. Thereafter, we compute the Euclidean distances [3] between $\mathbf{p}$ and all the other points in $\mathbb{N}_{\mathbf{p}}$. If all the distances are greater than a threshold, then it is regarded as an isolated point and subsequently eliminated from the online handwritten trace. The threshold is computed empirically for each document by considering the mean m and standard deviation σ of the Euclidean distances calculated between consecutive sample points of a stroke. In our work, the value of threshold is set to $(m + 3\sigma)$.

- **Same time stamp removal:** Due to possible timing issues in data capture, there could be points with different x and y coordinates but with the same time stamp. As a result

[3] In lieu of Euclidean distance, City Block or Mahalanobis distance can also be used.

of this, features dependent on the time stamp difference like speed could turn out to be infinite. To avoid such values, we consider the x and y coordinates amongst the set of points having same time stamp and average their values.

- **Stroke merge:** While analysing the online data, at times, we encountered that a handwritten trace (which can be represented as a single stroke) is broken down to more than one stroke. In such cases, we consider the time difference between the last point of a stroke and the first point of the succeeding stroke. If the obtained value lies within a threshold, the corresponding strokes are merged into a single stroke.

2.3 Feature Extraction

In this work, we consider a set of attributes / features, that are proposed at each sample point of the online trace of the handwritten input. As we shall see in this section, a subset of them are computed between a pair of points $(\mathbf{p}_i, \mathbf{p}_{i+r})$ in a stroke, where $\mathbf{p}_i = (x_i, y_i)$, $r > 0$. Said in another way, these features are derived by incorporating the information of sample points whose indices are spaced by a gap r. Accordingly, in this work, we hereinafter refer to this value as a 'gap parameter'. The use of $r = 1$ implies that we use consecutive sample points for deriving the attributes - the values of which at times may be influenced by the unintentional trembling of hand / jitter during the writing process [66]. Therefore we consider a higher value of r for feature extraction. The choice for the higher value of r is justified in Section 2.7.2 by an empirical study that demonstrates its influence on the performance of the identification system.

In addition to the above, we also consider the parameter r to capture the information in the neighborhood around the sample point $\mathbf{p}_i$ by deriving vicinity based features. Accordingly, based on the preceding notion, we enumerate the proposed attributes / features below:

- **Speed**: computed at the sample point $\mathbf{p}_i$ as:

$$v_i = \frac{\Delta(\mathbf{p}_i, \mathbf{p}_{i+r})}{t_{i+r} - t_i} \tag{2.1}$$

Here $\Delta(\mathbf{p}_i, \mathbf{p}_{i+r})$ is the Euclidean distance between the sample points (x_i, y_i) and (x_{i+r}, y_{i+r}) that are spaced with a gap r. The denominator $t_{i+r} - t_i$ is the difference between their corresponding time stamps.

- **Writing Direction**: The writing direction at $\mathbf{p}_i$ is defined by two features, that denote the the cosine and sine of the angle θ_i made by the line segment joining $\mathbf{p}_i$ and $\mathbf{p}_{i+r}$ with the horizontal axis.

$$cos(\theta_i) = \frac{x_{i+r} - x_i}{\Delta(\mathbf{p}_i, \mathbf{p}_{i+r})} \qquad sin(\theta_i) = \frac{y_{i+r} - y_i}{\Delta(\mathbf{p}_i, \mathbf{p}_{i+r})} \tag{2.2}$$

- **Curvature**: These two features / attributes relate to the cosine and the sine of the angle ϕ_i, computed using the vectors $\mathbf{p}_{i+r} - \mathbf{p}_i$ and $\mathbf{p}_i - \mathbf{p}_{i-r}$ respectively.

- **Vicinity aspect** : is the height to width ratio of the bounding box encompassing the $2r + 1$ sample points $\{\mathbf{p}_{i-r}, ..., \mathbf{p}_i, ..., \mathbf{p}_{i+r}\}$ in the vicinity of $\mathbf{p}_i$.

- **Vicinity Curliness**: is defined as the ratio of the length of the trajectory to the maximum amongst the width and height of the bounding box in the vicinity of $\mathbf{p}_i$.

Each of the above features is normalized to zero mean and unit variance. Accordingly, we represent the set of N_T normalized feature vectors for a document by $\{\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_{N_T}\} = \{\mathbf{f}_j\}_{j=1}^{N_T}$. Here $\mathbf{f}_j$ is the D dimensional feature vector whose attributes / features can be written as : $\mathbf{f}_j = [f_j(1), \cdots, f_j(d), \cdots, f_j(D)]$. It may be noted that the value of D is seven in our work.

2.4 Codebook description

Subsequent to the feature normalization, we generate a codebook comprising M codevectors. For this, we concatenate the D dimensional normalized feature vector sequences corresponding to the documents of randomly picked subset of writers. The k-means algorithm [4] is then applied on this sequence resulting in the codevectors $\{\boldsymbol{\mu_i}\}_{i=1}^{M}$ with $\boldsymbol{\mu_i} = \begin{bmatrix} \mu_i(1) & \mu_i(2) & \cdots \mu_i(D) \end{bmatrix}^T$.

[4] In our implementation, we set the value of k to the size of the codebook M.

We begin by outlining the formulation of the VLAD with its limitation - thereafter motivating our proposed descriptor in subsection 2.4.2.

2.4.1 VLAD

Given a codebook $\{\boldsymbol{\mu_i}\}_{i=1}^{M}$, each normalized feature vector $\mathbf{f}_j$ is assigned to the nearest codevector based on the minimum Euclidean distance criterion. Let the subset of normalized feature vectors in $\{\mathbf{f}_j\}_{j=1}^{N_T}$ assigned to codevector $\boldsymbol{\mu_i}$ be denoted as $\{\mathbf{f}_{ij}\}_{j=1}^{n_i}$ where $n_i < N_T$ and $\sum_{i=1}^{M} n_i = N_T$.

The idea of the VLAD is to aggregate the residuals / distortions (as defined in Equation 2.3) for each codevector with regards to all the feature vectors assigned to it [63]. Accordingly, the descriptor $\mathbf{S}_i$ for the codevector $\boldsymbol{\mu_i}$ is:

$$\mathbf{S}_i = \sum_{j=1}^{n_i} (\mathbf{f}_{ij} - \boldsymbol{\mu_i}) \qquad 1 \leq i \leq M \tag{2.3}$$

The vectors $\{\mathbf{S}_i\}_{i=1}^{M}$ are concatenated to give rise to the final descriptor $\mathbf{S}$ of dimension $D \times M$, which is subsequently L_2 normalized.

In the VLAD formulation, each of the components $f_{ij}(d) - \mu_i(d)$, $1 \leq d \leq D$ in the residual vector $\mathbf{f}_{ij} - \boldsymbol{\mu_i}$ contribute equally to the aggregation process - with both high and low distortions being given equal preference (refer Equation 2.3). This may be regarded as a limitation that can affect the accuracy for the writer identification problem.

As a demonstration to the above drawback, we consider a toy-example schematic of a Voronoi cell in Figure 2.2. This cell is shown to comprise two dimensional feature vectors of four writers (denoted by different symbols). Clearly, the feature vectors corresponding to writers 3 and 4 are linearly separable while those from writers 1 and 2 are not. The normalized sum of residuals of feature vectors (from the VLAD framework) of writers 1 and 2 are depicted by the blue and brown arrows [5]. It is apparent that owing to their similar spatial orientation, the separability between these writers is reduced. However, it can be noted that some of the

[5]For the present discussion, owing to the non-linear separable nature of writers 1 and 2, we focus on improving their discrimination with regards to the identification problem.

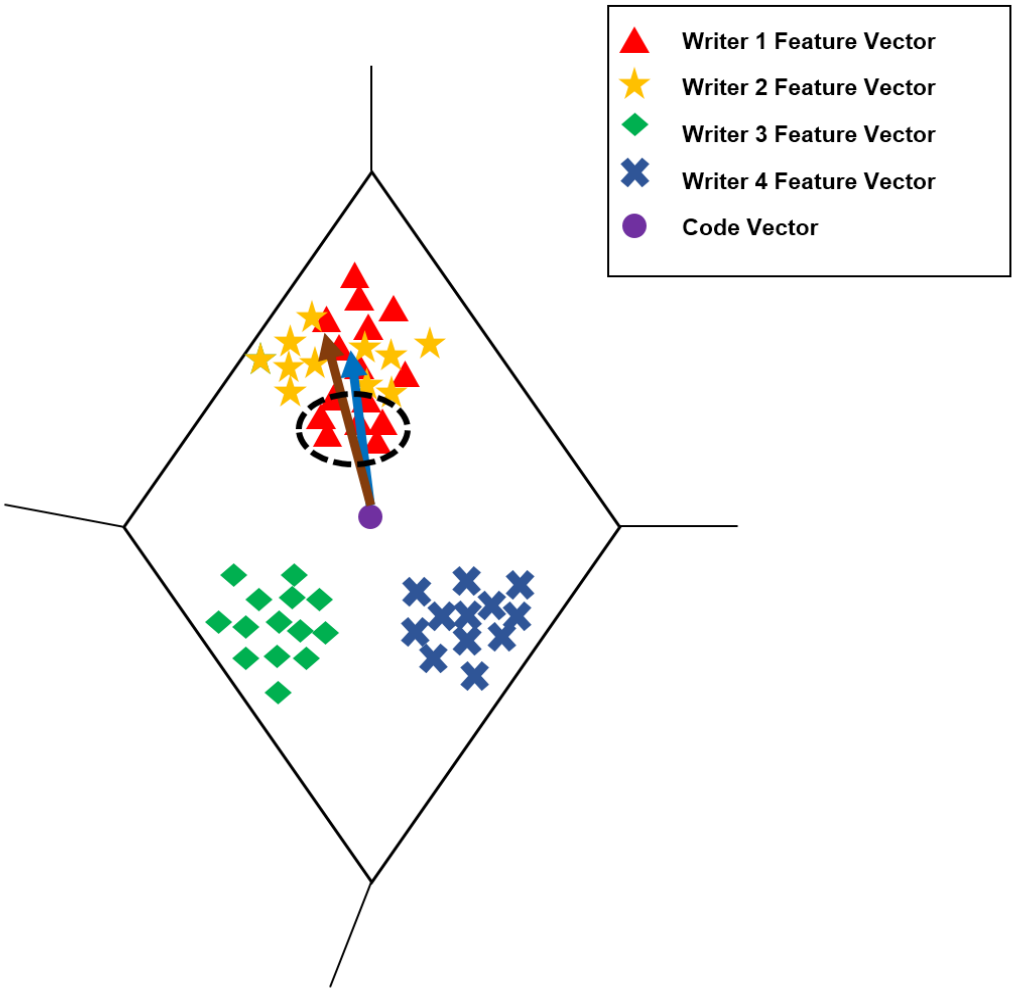


Figure 2.2: An illustrative toy-example for demonstrating the drawback of the VLAD. For more details, kindly refer to the textual content in subsection 2.4.1.

feature vectors of writer 1 (highlighted by the area in the black ellipse) are located closer to the representative two dimensional codevector of the cell (denoted by a violet circle) as compared to those of writer 2 . Hence it is worth assigning a higher score to such vectors - in a way so as to increase the discrimination of these two writers. This is the crux of our proposal discussed in the following subsection.

The VLAD framework aggregates the signed difference by considering how the feature vectors are located spatially relative to their assigned codevector. This idea is extended further in our proposal as follows: we separately score each of the distortion components by taking into perspective whether they are positive or negative.

2.4.2 Proposed Descriptor

Given a document comprising N_T points, we assign each of the feature vectors to their nearest codevector. Let $\{\mathbf{f}_{ij}\}_{j=1}^{n_i}$ denote the feature vectors contained in the Voronoi region of codevector $\boldsymbol{\mu_i}$. By definition, n_i represents the number of feature vectors and satisfies $n_i < N_T$ and $\sum_{i=1}^{M} n_i = N_T$.

We define two scores $S_{ij}^{+}(d)$ and $S_{ij}^{-}(d)$ for the d^{th} attribute of the j^{th} feature vector assigned to the codevector $\boldsymbol{\mu_i}$.

$$S_{ij}^{+}(d) = \begin{cases} \frac{1}{1+|f_{ij}(d)-\mu_i(d)|} & f_{ij}(d) \geq \mu_i(d) \\ 0 & otherwise \end{cases}$$

$$S_{ij}^{-}(d) = \begin{cases} \frac{-1}{1+|f_{ij}(d)-\mu_i(d)|} & f_{ij}(d) < \mu_i(d) \\ 0 & otherwise \end{cases} \tag{2.4}$$

$$1 \leq i \leq M, \quad 1 \leq j \leq n_i, \quad 1 \leq d \leq D$$

By notation, $\mu_i(d)$ represents the value of the d^{th} attribute in the codevector $\boldsymbol{\mu_i}$. The negative sign in the formulation of $S_{ij}^{-}(d)$ is an implication of the value of $f_{ij}(d)$ being less than $\mu_i(d)$. Further to this, it may be observed from Equation 2.4, that we are essentially capturing in a way the proximity of each of the attributes in a feature vector to their corresponding values in the assigned codevector. In particular, the term $|f_{ij}(d) - \mu_i(d)|$ denotes the absolute value of

the residual / distortion value between $f_{ij}(d)$ and $\mu_i(d)$ [6]. Note that, the more proximal the feature attribute $f_{ij}(d)$ is to $\mu_i(d)$, the higher is the absolute value of the score $S_{ij}^{-}(d)$ or $S_{ij}^{+}(d)$.

With regards to obtaining the descriptor of the codevector $\boldsymbol{\mu_i}$, the scores obtained for the d^{th} feature attribute of the feature vectors $\{\mathbf{f}_{ij}\}_{j=1}^{n_i}$ are accumulated and normalized with regards to the values obtained from the entire document as follows:

$$\tilde{S}_i^{+}(d) = \frac{\sum_{j=1}^{n_i} S_{ij}^{+}(d)}{\sum_{k=1}^{M}\sum_{j=1}^{n_k} S_{kj}^{+}(d)}$$
$$\tilde{S}_i^{-}(d) = \frac{\sum_{j=1}^{n_i} S_{ij}^{-}(d)}{\sum_{k=1}^{M}\sum_{j=1}^{n_k} S_{kj}^{-}(d)} \qquad (2.5)$$
$$1 \leq i \leq M, \qquad 1 \leq d \leq D$$

The numerator terms $\sum_{j=1}^{n_i} S_{ij}^{+}(d)$ and $\sum_{j=1}^{n_i} S_{ij}^{-}(d)$ with $1 \leq d \leq D$ represent respectively, the aggregation of scores from the positive and negative distortion values corresponding to a codevector $\boldsymbol{\mu_i}$.

In our work, the normalized scores in Equation 2.5 for each of the D feature attributes are concatenated as the descriptor $\mathbf{S}_i$ for codevector $\boldsymbol{\mu_i}$.

$$\mathbf{S}_i = \begin{bmatrix} \tilde{S}_i^{+}(1) & \tilde{S}_i^{-}(1) & \cdots & \tilde{S}_i^{+}(d) & \tilde{S}_i^{-}(d) & \cdots & \tilde{S}_i^{+}(D) & \tilde{S}_i^{-}(D) \end{bmatrix}^T \qquad (2.6)$$

It is interesting to note from Equation 2.5 that the scores $\tilde{S}_i^{+}(d)$ and $\tilde{S}_i^{-}(d)$ for the d^{th} feature attribute obtained across the M codevectors sum to one. Owing to this constraint, the document can be represented by concatenating descriptors corresponding to any of the $M-1$ codevectors. In our work, we consider the descriptors of the first $M-1$ codevectors. Put in another way, the final descriptor $\mathbf{S}$ is the concatenation of $\{\mathbf{S}_i\}_{i=1}^{M-1}$

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_1 & \mathbf{S}_2 & \cdots & \mathbf{S}_{M-1} \end{bmatrix}^T \qquad (2.7)$$

[6] Without loss of generality, the distortion / residual value can be positive or negative, depending on the sign of the difference $f_{ij}(d) - \mu_i(d)$.

We see that the dimension of $\mathbf{S}$ is $2 \times D \times (M-1)$. On a passing note, in the scenario when none of the points in the document are assigned to a codevector $\boldsymbol{\mu_i}$, the $2 \times D$ entries of the corresponding descriptor $\mathbf{S}_i$ are set to zero.

For sake of completeness, we provide the intuition behind deriving scores for positive and negative distortions separately with respect to the feature attributes in our proposal (refer Equation 2.4). To substantiate our scoring approach, we consider the scenario wherein feature vectors of two different writers, namely writer 1 and 2 are spatially oriented as shown in Figure 2.3. With regards to this illustration, we consider upon applying the scoring function of the form of Equation 2.8 to all the attributes - irrespective of their sign of distortion to the assigned codevector. It is quite straightforward to see that the descriptors for both these two writers resulting from the scores would be very similar, thereby reducing the discrimination between them.

$$S_{ij}(d) = \frac{1}{1 + |f_{ij}(d) - \mu_i(d)|} \tag{2.8}$$
$$1 \leq i \leq M, \quad 1 \leq j \leq n_i, \quad 1 \leq d \leq D$$

A better separation, however can be obtained when only one of the scoring functions $S_{ij}^{-}(d)$ and $S_{ij}^{+}(d)$ are chosen for $f_{ij}(d)$, depending on the sign of the distortion $f_{ij}(d) - \mu_i(d)$. This ensures the necessity of the scoring framework of Equations 2.4 and 2.5, that is used in our proposed descriptor formulation.

2.4.3 Computational Complexity

Based on the preceding discussion on the VLAD and our proposed descriptor, we mention in this subsection, the computation time complexities of each of them in Table 2.1. We present our analysis with regards to each step of the two algorithms for a document of a writer. We see that in contrast to VLAD, our strategy requires a few additional operations. The 'n/a' entries in the Table indicate that the corresponding operation is not relevant.

The calculation of the Euclidean distance of each of the N_T feature vectors to the M codevectors $\{\boldsymbol{\mu_i}\}_{i=1}^{M}$ and their assignment to the closest codevector is common for both the

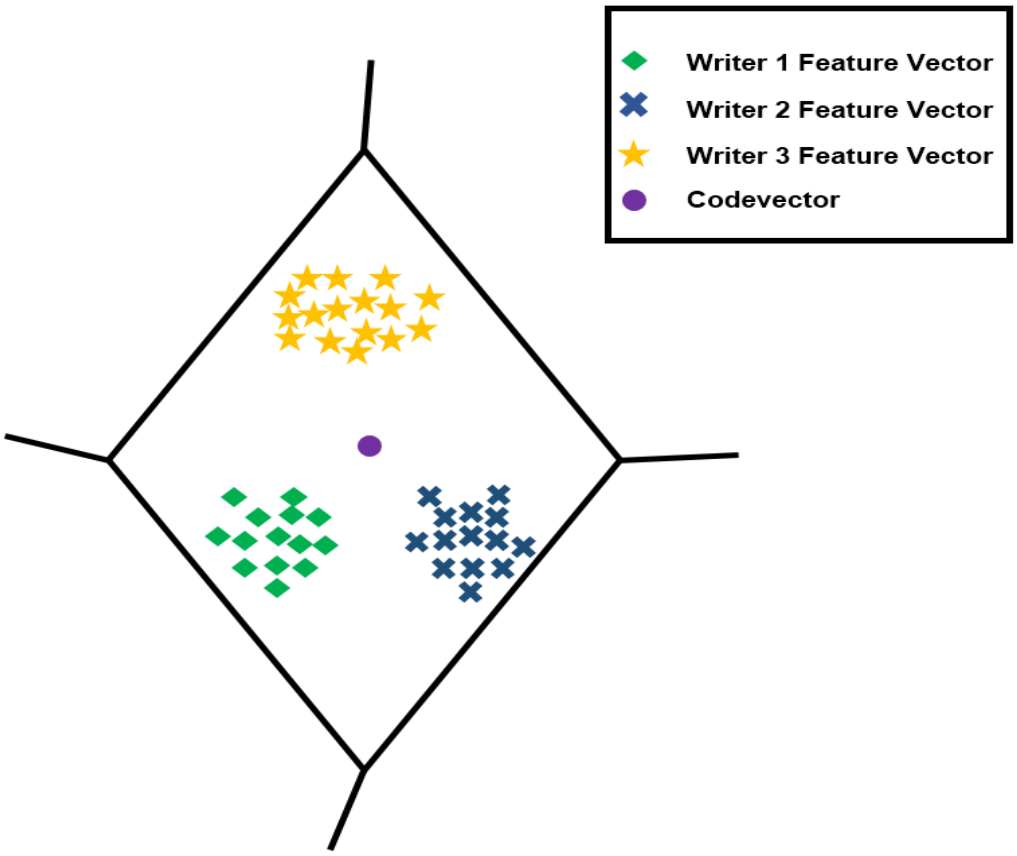


Figure 2.3: An illustrative toy example for motivating the need of two scores for each attribute in our proposal. For more details, kindly refer to the textual content

Table 2.1: Comparison of the Computation Complexity of the proposed algorithm for a document of a writer having N_T feature vectors with the VLAD. Here n/a stands for "Not Applicable", indicating that the corresponding operation is not relevant for the VLAD. For the description of notation, the reader may refer to the text.

Description	Proposed Methodology	VLAD
Distance Computation to each codevector	$\mathcal{O}(N_T \times D \times M)$	$\mathcal{O}(N_T \times D \times M)$
Assignment to nearest codevector	$\mathcal{O}(N_T \times M)$	$\mathcal{O}(N_T \times M)$
`Computation of:`		
Scores in Eqn 2.4	$\mathcal{O}(D \times \max_{1 \le i \le M} n_i)$	n/a
Scores in Eqn 2.5	$\mathcal{O}(D \times \max_{1 \le i \le M} n_i)$	n/a
$\mathbf{S}_i$	$\mathcal{O}(D)$	$\mathcal{O}(D \times \max_{1 \le i \le M} n_i)$
$\mathbf{S}$	$\mathcal{O}(M)$	$\mathcal{O}(M)$

algorithms and possess a complexity of $\mathcal{O}(N_T \times D \times M)$ and $\mathcal{O}(N_T \times M)$ respectively. The computation of the $S_{ij}^{+}(d)$ and $S_{ij}^{-}(d)$ scores in Equation 2.4 across each feature attribute and codevector for the proposed methodology incurs a time complexity of $\mathcal{O}(D \times \max_{1 \le i \le M} n_i)$. A similar comment does hold good for the calculation of $\tilde{S}_i^{+}(d)$ and $\tilde{S}_i^{-}(d)$ scores in Equation 2.5. It may be noted that these calculations are not applicable to the VLAD. Following this, the computation of the descriptor $\mathbf{S}_i$ for the codevector $\boldsymbol{\mu}_i$ require a complexity of $\mathcal{O}(D \times \max_{1 \le i \le M} n_i)$ and $\mathcal{O}(D)$ for the VLAD and proposed methodology respectively. Finally, the descriptor $\mathbf{S}$ obtained as a concatenation of $\{\mathbf{S}_i\}_{i=1}^{M}$ has a complexity $\mathcal{O}(k)$.

Lastly, for better clarity, the pseudo-code of our proposed codebook descriptor derived from the first $M-1$ codevectors is shown in Algorithm 1.

2.5 Writer Identification

The codebook descriptors extracted from each enrolled document written by the W writers are trained using the Support Vector Machine (SVM) [67] for obtaining the model parameters.

Algorithm 1 Proposed Codebook descriptor generation

- **Input:**
  - Document containing N_T points.
  - Codebook with a set of M codevectors $\{\boldsymbol{\mu_i}\}_{i=1}^{M}$.
- **Output:**
  - Codebook descriptor $\mathbf{S}$.
- **Algorithm:**

For a given gap parameter value r, compute the D dimensional feature vector sequence $\{\mathbf{f}_j\}_{j=1}^{N_T}$.

Assign each feature vector $\mathbf{f}_j$ to the nearest codevector in the codebook.

% `Descriptor Computation`:

For every codevector in $\{\boldsymbol{\mu_i}\}_{i=1}^{M-1}$
Compute n_i - the number of feature vectors assigned to codevector $\boldsymbol{\mu_i}$.
For every feature vector assigned to $\boldsymbol{\mu_i}$
Compute $S_{ij}^{+}(d)$ and $S_{ij}^{-}(d)$ scores across the D attributes.
End For
End For

For every codevector in $\{\boldsymbol{\mu_i}\}_{i=1}^{M-1}$
Compute $\tilde{S}_{i}^{+}(d)$ and $\tilde{S}_{i}^{-}(d)$ scores across the D attributes.
End For

Stack the $\tilde{S}_{i}^{+}(d)$ and $\tilde{S}_{i}^{-}(d)$ scores across the D attributes to get $\mathbf{S}_i$.

Concatenate the descriptors $\{\mathbf{S}_i\}_{i=1}^{M-1}$ to obtain the final descriptor $\mathbf{S}$.

The SVM is a supervised binary classifier, that belongs to the class of maximum margin classifier. Basically, the aim is to obtain the decision surface that maximizes the separation between the positive and negative classes. In the following, we provide a brief description of the same in the general pattern recognition setting.

Consider a training data set consisting of pairs $(\mathbf{x}_i, l_i), 1 \leq i \leq N_{Tr}$, where each input feature vector $\mathbf{x}_i$ is labelled as l_i. The value of l_i corresponds to one of the binary labels $\{-1, +1\}$. When the feature vectors of the two categories are linearly separable, the SVM minimizes the cost function defined by:

$$J(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} \tag{2.9}$$

subject to the constraints

$$l_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \qquad 1 \leq i \leq N_{Tr} \tag{2.10}$$

Here $\mathbf{w}$ is the weight vector and b is the bias term. However, the more general setting is that the categories to be recognized are not linearly separable. In such cases, the above cost function is reformulated by introducing slack variables $\xi_i \geq 0 \;\; i = 1, 2, ...N_{Tr}$. The SVM now finds $\mathbf{w}$ to minimize

$$J(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N_{Tr}} \xi_i \tag{2.11}$$

subject to

$$l_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \tag{2.12}$$

The constant C is a regularization parameter. With regards to efficient implementation, the SVM maps the training data to a higher dimensional feature space, via a mapping function. This is the so-called 'kernel-trick' wherein, a kernel function is defined to obtain the optimal hyperplane without explicitly using the mapping function.

With respect to multi-category classification, many approaches have been proposed in literature - the most popular being the one vs one and one vs all. In the former, for C_1 categories, a hyperplane is constructed for each pair, thus resulting in $\frac{C_1(C_1-1)}{2}$ binary classifiers. During testing of new data, each classifier gives a vote of one to the winning category. Thereafter, the

assignment of the category is made with respect to the one with the highest number of accumulated votes. Contrary to this, in the one vs all scheme, a classifier is constructed by treating all training data belonging to one category as positive and remaining as negative. This in turn results in C_1 two-category SVMs. The test data is assigned to the category that provides the highest discriminant function.

For the experiments discussed in the thesis, we have employed a one vs all multi-class SVM approach with the Radial Basis Function (RBF) kernel. The choice of this kernel is attributed to its encouraging results with our strategy, amongst those available in LIBSVM Toolkit. As far as the training is concerned, the parameters C and γ of the kernel are selected using grid search.

2.6 Experimental set-up

In this section, we present an outline of the two databases used, together with their enrolment and evaluation protocols adopted in our experiments.

2.6.1 Database Description

The IAM Online handwriting database consists of samples of 217 writers acquired from a whiteboard [68]. To acquire a database of handwritten sentences contained in the corpus, the texts in the corpus are split into fragments of about 50 words each. These fragments were then copied onto forms on paper and each writer was asked to write down the text contained in eight forms on the whiteboard. Figure 2.4 shows a sample paragraph of a writer from this database. The online handwriting data collected contains x-coordinate, y-coordinate and time stamp information.

The IBM-UB1 handwriting dataset contains online handwriting data from 43 writers, collected on the IBM Crosspad [69]. For each document written by a specific writer, a summary text - a one to two page essay on a particular topic, and a corresponding 'query' text was generated. The 'query' text is a collection of approximately 25 words that attempt to distill the essence of the summary text. Figure 2.5 shows a snapshot of a sample summary and query

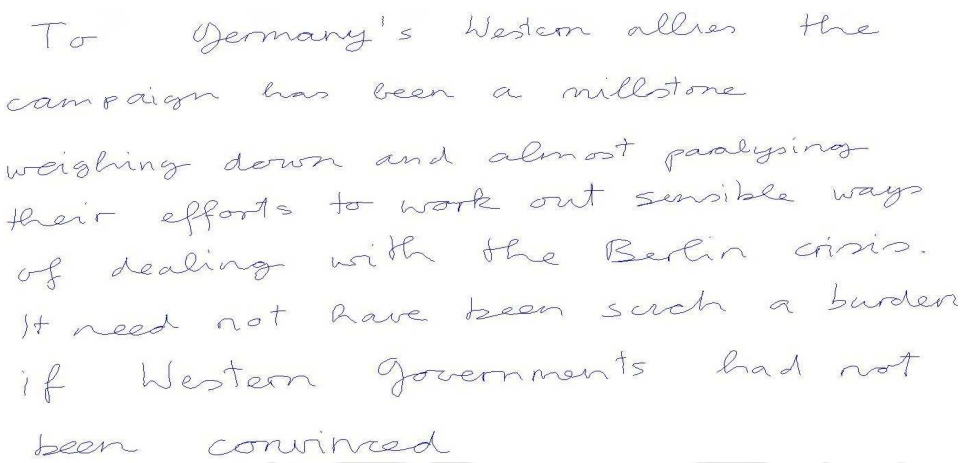
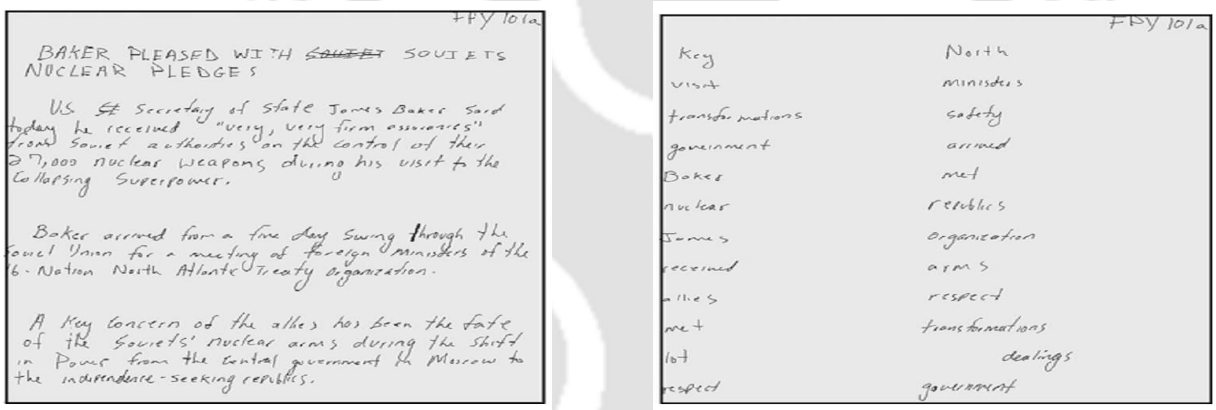


Figure 2.4: Sample paragraph of a user from the IAM Online Database.



(*a*) (*b*)

Figure 2.5: A snapshot of a summary and query text from the IBM-UB1 handwriting database

text. The basic attributes collected comprise the spatial x-coordinate and y-coordinate.

2.6.2 Training and testing protocol

In most experiments reported in this study, the following protocol is used for the IAM Online Database. Out of the eight paragraphs written by each writer, four are chosen at random for enrolment and the remaining are used for performance evaluation. In addition, training paragraphs corresponding to a subset of 50 writers were picked at random for the codebook generation.

For the IBM-UB1 handwriting data-set, we are using the protocol as suggested in [55] with 80% of the summary document paragraphs selected at random from each writer for enrolment

and the rest for testing (including the query pages). With regards to the generation of the codebook in the training phase, a set of four paragraphs from a subset of 15 writers are randomly chosen.

Two types of training strategies are explored in this work, namely paragraph and text-line level. For paragraph level training, a codebook descriptor is derived separately for each paragraph of the online handwritten data where as for the text line level, the paragraphs are split into individual text lines and thereof, a descriptor is obtained for each text line.

The selection of documents for enrolment, codebook generation, construction of descriptor, followed by classification referred to as a trial is performed ten times. The average writer identification rate (obtained over the trials) are reported for each experiment.

2.7 Performance Evaluation

In this section, we demonstrate results of experiments that present the utility of our proposal in identifying the authorship of online handwritten documents.

2.7.1 Comparison with VLAD for varying codebook size M

In this subsection, we show the trends of the VLAD and our proposed framework with regards to writer identification rates for varying sizes of codebook M. The average writer identification rates for the two techniques is depicted in Figures 2.6 (a)-(d) for codebook sizes ranging from 5 to 100 in steps of five. Note that the blue and red curves denote the performance of VLAD and our proposal respectively. The sub-figures in the first column (a), and (c) represent the paragraph level identification rates for the IAM and IBM-UB1 respectively. Likewise, the text line level performance for these databases are plotted on the remaining sub-figures of (b), and (d). For simplicity, we abbreviate the average writer identification rate as IR.

Owing to the fact that the above plots are aimed at demonstrating, in general the trend with different values of M, we employ the gap parameter $r = 1$ to derive the features for obtaining the codebook descriptors. However, at the same time, it is to be mentioned that the

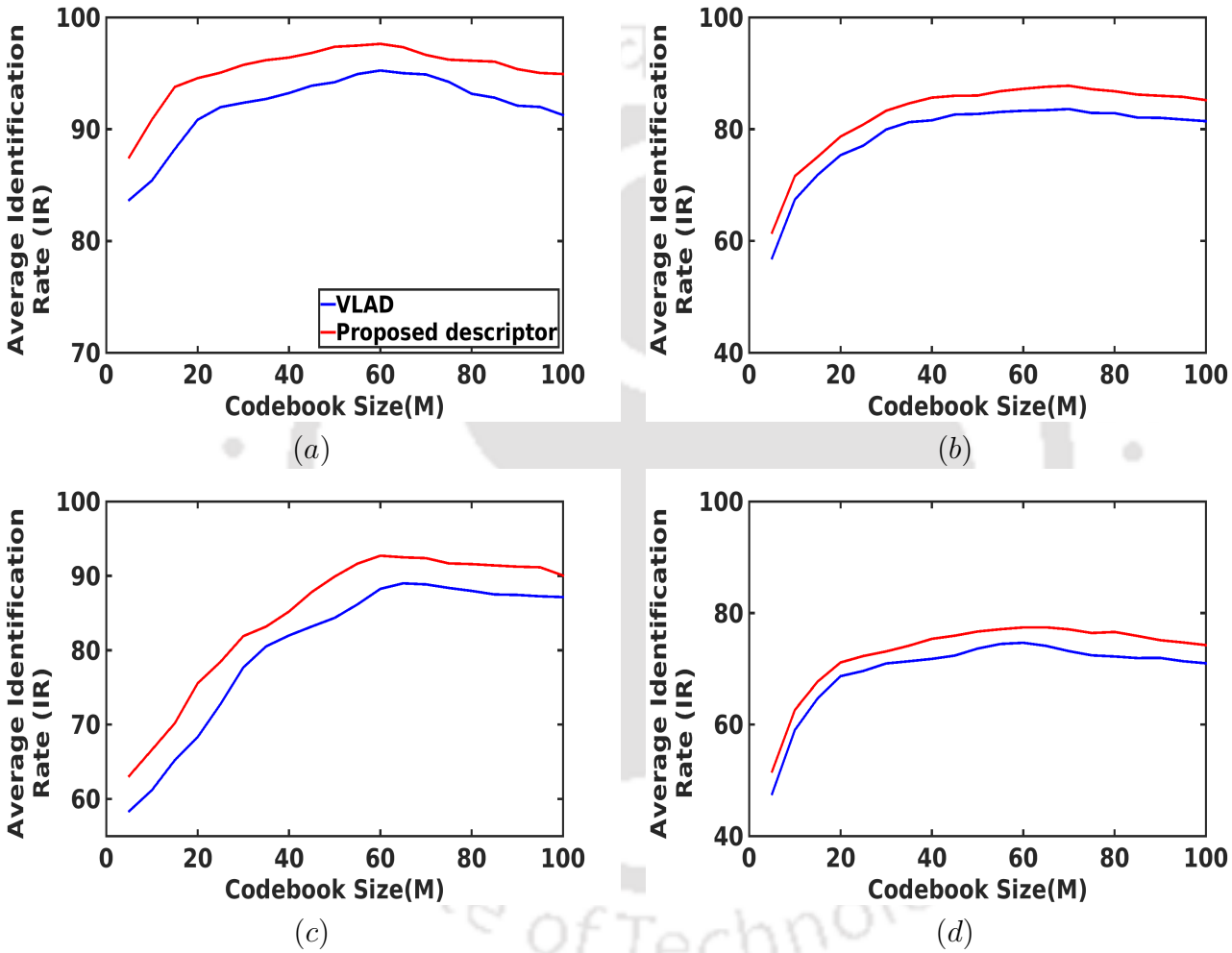


Figure 2.6: Trend of the average writer identification rate obtained for VLAD (shown in blue) and the proposed codebook descriptor (in red) for varying size of codebook M. The sub-figures (a) and (c) represent the performance at the paragraph level for the IAM and the IBM-UB1 respectively, while those of (b) and (d) denote the same at the text-line level. Note that for this experiment, we use the gap parameter $r = 1$.

gap parameter value applied on the feature set does have an influence on improving the writer identification rates, as will be demonstrated in the following subsection.

The inferences from the different plots of Figure 2.6 across the databases are as follows:

(i) For the varying codebook sizes being considered, our descriptor provides a better performance than VLAD. This is owing to its ability to assign scores based on the residual / distortion values, thereby increasing the discrimination of the feature vectors of the writers.

(ii) Taking into consideration, the trend of the performance of VLAD and the proposed descriptor, we see that the average identification rate initially increases with codebook size and becomes comparable at moderate sizes.

(iii) Owing to coarser quantization, small-sized codebooks do not adequately capture the nuances that discriminate the features of different writers and hence the low average identification rate.

2.7.2 Influence of the gap parameter

Recall from Section 2.3 that the features used for generating the codebook and the corresponding descriptor for an input handwritten document is dependent on the gap parameter value r. As a next experiment, we empirically show its influence on the variation of writer identification rates at both paragraph and text-line respectively by evaluating on the proposal discussed in Section 2.4.2. Tables 2.2 and 2.3 depicts the result of the same for values of r ranging from one to six in steps of one for select codebook sizes. The entries in the Table report the best average identification rates together with the size of the codebook employed to achieve it.

Clearly, across all the databases, a higher value of r is found to be more apt in providing a higher identification rate when compared to $r = 1$. The highest average identification rates with our proposal (marked in **bold**) is obtained with $r = 3$ for the IAM and IBM-UB1. However with increasing values of r, the performance of the writer identification begins to drop - as can

Table 2.2: Average writer identification rates (in %) obtained with our codebook descriptor proposal for different values of the gap parameter r at the paragraph and text line level for the IAM database. The best average writer identification rate obtained is highlighted in **bold**

	Codebook size M											
r	5	15	25	35	45	55	60	65	70	80	90	100
1	86.56	90.61	93.98	95.02	96.15	97.37	97.59	97.36	97.16	96.18	95.66	95.03
2	88.94	93.91	95.22	96.18	97.38	97.95	98.07	97.80	97.56	97.26	97.13	97.01
3	90.62	95.97	97.38	97.82	98.22	98.53	**98.82**	98.56	98.68	98.58	98.48	98.41
4	89.58	94.48	96.92	97.41	97.96	98.49	98.70	98.51	98.25	98.14	97.95	97.84
5	87.96	93.72	95.94	97.16	97.76	98.05	98.28	98.19	98.11	97.91	97.69	97.46
6	87.41	93.37	95.25	96.80	97.15	97.34	97.68	97.53	97.44	97.27	97.08	96.96

(a) Paragraph level

	Codebook size M											
r	5	15	25	35	45	55	60	65	70	80	90	100
1	62.31	76.45	78.20	79.61	80.43	81.88	83.23	84.85	85.12	84.47	83.77	83.20
2	65.07	80.21	83.84	85.39	86.60	86.95	87.29	87.89	87.63	87.15	86.62	86.35
3	69.03	81.44	84.32	86.86	87.82	88.97	89.15	**89.92**	89.69	88.93	88.48	87.96
4	66.56	79.02	82.85	84.60	85.18	87.52	88.22	88.74	89.19	88.25	87.72	87.16
5	60.40	77.62	81.77	83.65	84.50	86.55	87.11	87.99	57.53	86.61	86.08	85.27
6	57.36	73.39	80.74	82.23	83.60	85.50	86.10	86.97	87.03	86.36	85.66	84.92

(b) Text line level

Table 2.3: Average writer identification rates (in %) obtained with our codebook descriptor proposal for different values of the gap parameter r at the paragraph and text line level for the IBM-UB1 database. The best average writer identification rate obtained is highlighted in **bold**

	Codebook size M											
r	5	15	25	35	45	55	60	65	70	80	90	100
1	63.10	70.19	78.45	83.19	88.84	91.66	92.72	91.50	91.39	90.59	90.25	90.06
2	64.86	72.33	81.78	85.30	90.38	93.56	93.83	94.13	93.74	92.93	92.58	92.16
3	66.30	74.02	83.06	87.43	92.88	95.39	**96.10**	95.90	95.79	95.63	95.29	94.93
4	65.49	73.63	82.07	86.92	91.63	94.49	95.62	95.47	95.27	94.96	94.87	94.72
5	65.27	73.12	81.97	85.68	90.50	94.29	94.44	94.15	93.85	93.72	93.43	93.12
6	64.40	71.03	80.92	84.44	89.66	93.06	93.41	93.67	93.49	93.18	92.95	92.75

(a) Paragraph-level

	Codebook size M											
r	5	15	25	35	45	55	60	65	70	80	90	100
1	57.00	72.11	73.66	74.80	75.76	76.62	76.91	77.45	77.13	76.70	76.25	75.39
2	59.55	73.77	75.59	76.57	77.09	77.96	78.66	79.15	79.45	78.92	78.14	77.80
3	61.80	74.99	77.64	78.14	78.88	79.67	80.40	80.95	**81.59**	81.31	80.94	80.53
4	60.96	74.46	77.22	77.78	78.39	78.76	79.81	80.93	81.32	81.03	80.62	80.26
5	60.41	74.16	76.71	77.20	77.66	78.52	79.44	80.79	80.54	80.32	80.03	79.84
6	58.92	72.94	74.04	75.83	76.37	77.04	77.84	78.19	78.56	78.16	77.61	77.02

(b) Text-line level

be noted by the entries for $r = 6$.

A subset of features (speed, writing direction and curvature) in the set of features explained in Section 2.3 capture the dynamic information of each point in a stroke where as the vicinity aspect and vicinity curliness features capture the essence of the neighbourhood around the point of interest. The relatively low identification rates at $r = 1$ could be attributed to the vicinity based features not being able to capture enough neighbourhood information to be able to discriminate between writers. Though the vicinity based features are able to capture the neighbourhood information effectively at gap parameter values beyond $r = 3$, features such as writing direction and curvature tend to lose the dynamic information as they are computed between points that may be far apart and therefore may not be able to capture the writer-specific characteristics.

We also analyzed the performance trend of our proposal for different gap parameter values from the viewpoint of the feature distributions. For sake of demonstration, we considered a document, each from the IAM and IBM-UB1 database and extracted the velocity and vicinity curliness features for the gap parameter $r = 1$ and $r = 3$ respectively. The extracted features were then normalized by using the min-max normalization [7]. The histograms were constructed with the number of bins set at 20.

Figure 2.7 and 2.8 show the corresponding plots for the vicinity curliness and velocity features of the two documents being considered. It can be noticed that at $r = 1$, majority of the values are close to zero there by making the distribution skewed. For $r = 3$, however, we see that the skewness is reduced. To explain the improved performance of our writer identification system with $r = 3$, we refer to the study in [70, 71], where the authors demonstrate that the performance of the k-means algorithm are affected by features with skewed data distributions. In the context of our work, at $r = 3$, the corresponding histograms for both the velocity and vicinity curliness features across the documents present a lesser degree of skew, as compared to $r = 1$. This, in turn ensures that the k-means would do much better in clustering the features

[7] It is to be noted that the choice of the features and the min-max normalization scheme is used merely for the sake of depicting the histogram plots.

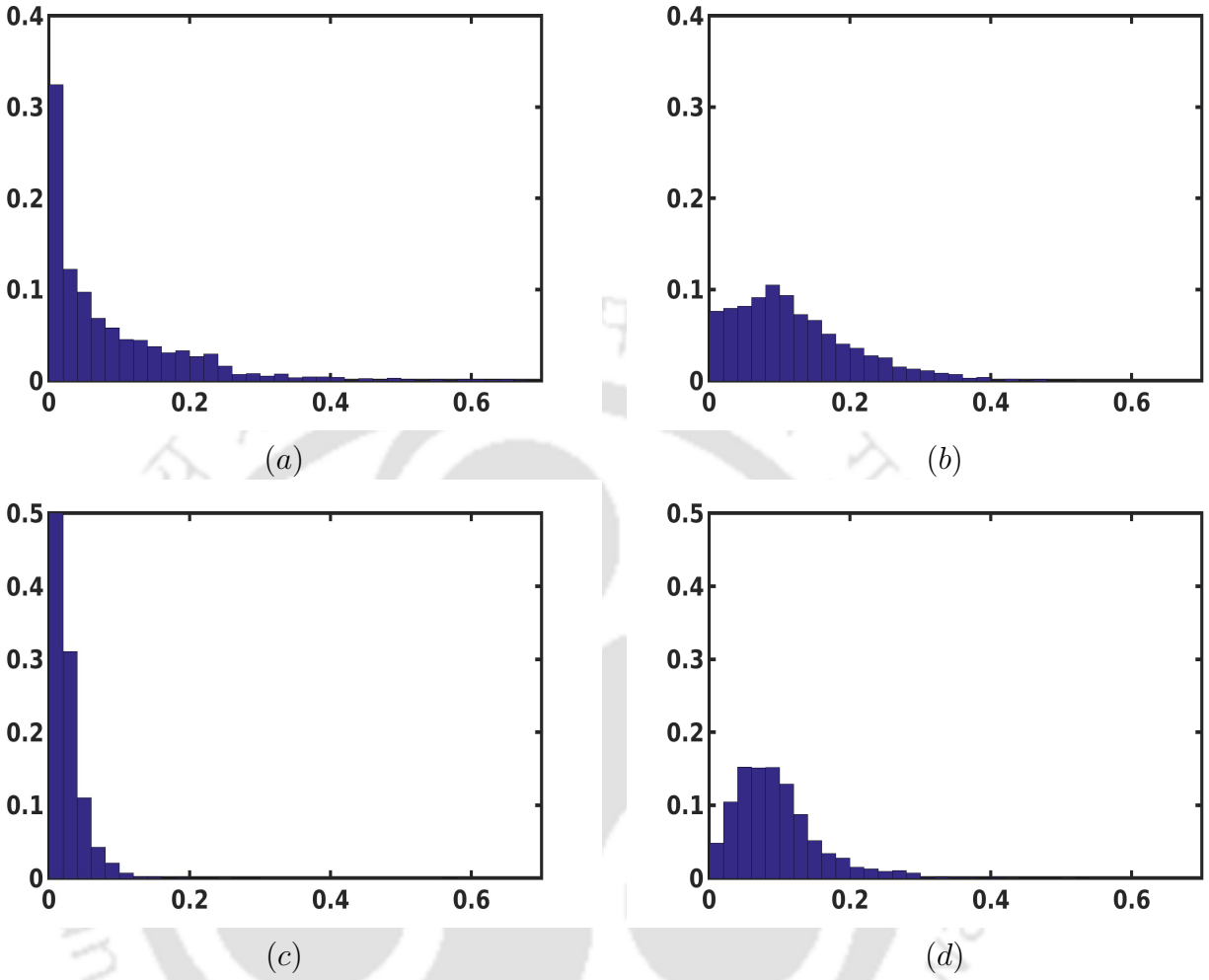


Figure 2.7: Illustration of the histograms for the velocity and vicinity curliness feature of a document belonging to the IAM database. The plots are depicted for two values of gap parameter, namely $r = 1$ and $r = 3$. The histograms in sub-figures (a) and (b) present the velocity feature, while those of (c) and (d) show the vicinity curliness feature for the two gap parameters respectively.

for the generation of the codebook, thus leading to descriptors that improve the performance of the writer identification system. For completeness, in Table 2.4 we tabulate the values of the skewness obtained with regards to the histograms of Figure 2.7 and 2.8.

Furthermore, for the implementation of the descriptors in Sections 2.7.3, 2.7.4 and 2.7.5 across the databases, we did analyze the influence of the gap parameter values on the writer identification performance. Without loss of generality, it was empirically observed that the value of the gap parameter resulting in the best average identification rate was again $r = 3$ on both the databases.

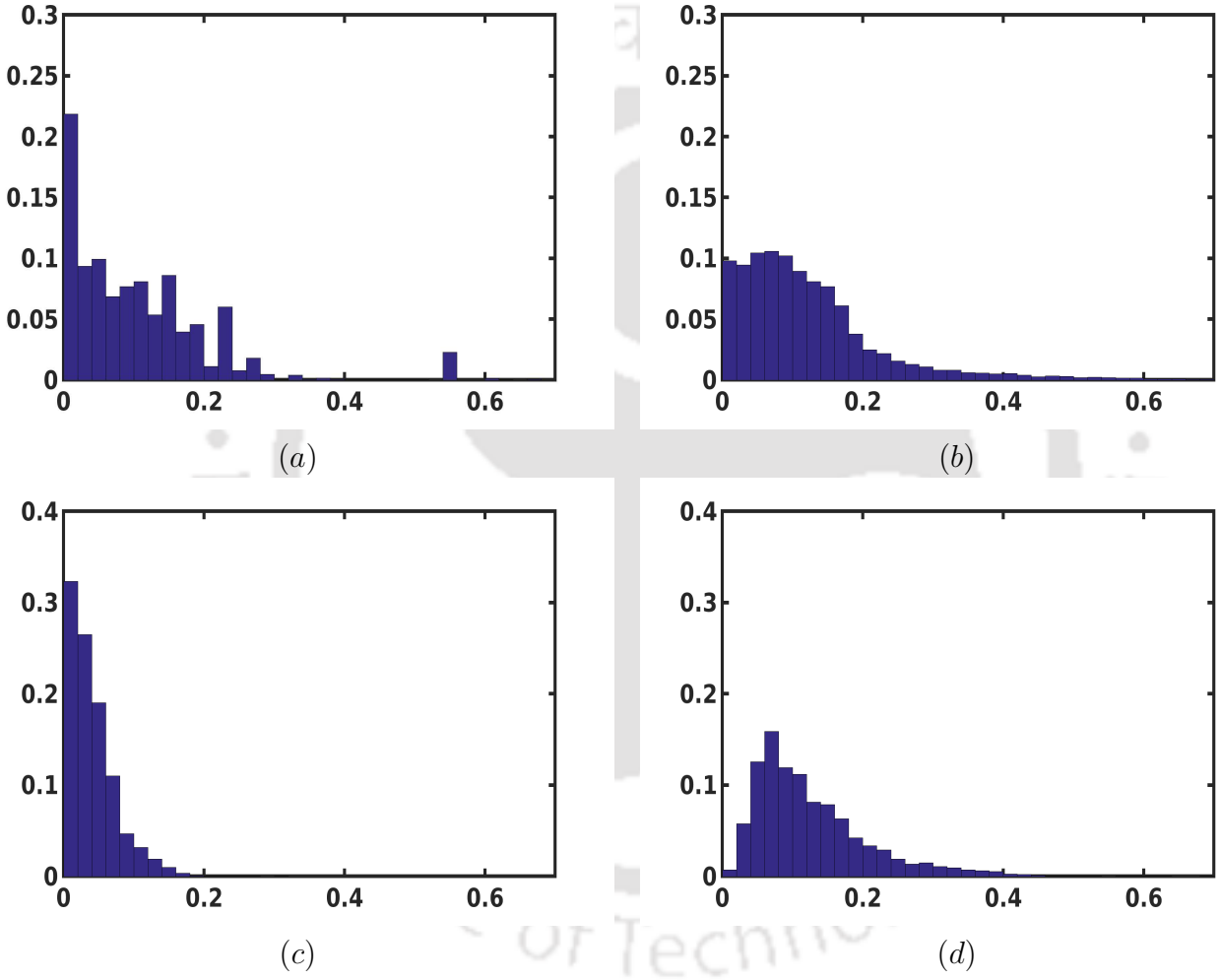


Figure 2.8: Illustration of the histograms for the velocity and vicinity curliness feature of a document belonging to the IBM-UB1 database. The plots are depicted for two values of gap parameter, namely $r = 1$ and $r = 3$. The histograms in sub-figures (a) and (b) present the velocity feature, while those of (c) and (d) show the vicinity curliness feature for the two gap parameters respectively.

Table 2.4: Values of the skewness obtained from the histograms of the features in Figures 2.7 and 2.8 at gap parameter $r = 1$ and $r = 3$.

Feature	IAM database		IBM-UB1 database	
	$r = 1$	$r = 3$	$r = 1$	$r = 3$
Velocity	4.82	2.25	3.61	2.14
Vicinity curliness	4.63	1.56	2.83	1.63

2.7.3 Comparison to variants of VLAD

The experiments outlined so far considered the original VLAD framework described in [63] for writer identification. Nevertheless, there have been variants of this and hence, for sake of completion, we compare our proposed scheme with them. In Table 2.5, we outline the results of this empirical study at paragraph and text-line level.

For sake of convenience, we have abbreviated each of the variants as follows:

- `VLAD+` L_2`-Norm` [72]: This is the version of the VLAD discussed in Section 2.4.1. It comprises accumulating the D residual vectors across the M codevectors of the codebook - thus leading to a $D \times M$ descriptor $\mathbf{S}$, which is then L_2 normalized.

- `VLAD+ Power-law` [63]: Each of the individual aggregated distortion $S_i(d)$ in a residual vector $\mathbf{S}_i$ are transformed with the function $sign(S_i(d)) \times |S_i(d)|^{\alpha}$, where $0 \leq \alpha \leq 1$. Thereafter, they are accumulated across the M codevectors, followed by a L_2 norm. Our experiments were run for different combinations of (α, k) and thereafter the best average identification results reported in Table 2.5.

- `VLAD+ innorm` [64]: This version is similar to `VLAD+` L_2`-Norm` scheme, except that L_2 norm is performed for each of the descriptors $\mathbf{S}_i$.

- `VLAD+ SSR` [64]: Each of the aggregated distortion $S_i(d)$ in $\mathbf{S}_i$ are transformed with the function $sign(S_i(d)) \times \sqrt{|S_i(d)|}$. Thereafter, they are accumulated across all M codevectors and normalized with L_2.

Table 2.5: Summary of the best average writer identification rate (in %) with different variants of VLAD and our proposal for the IAM and IBM-UB1 databases. The size of the codebook M employed in each case is indicated.

Descriptor Type	IAM database				IBM UB1 database			
	Paragraph level		Textline level		Paragraph level		Textline level	
	IR	M	IR	M	IR	M	IR	M
VLAD+L_2-Norm [72]	97.31	55	87.46	60	92.76	65	78.16	60
VLAD+Power law [63]	98.03	50	88.34	55	94.97	60	79.84	65
VLAD +innorm [64]	97.73	50	88.09	55	93.85	60	79.06	65
VLAD+SSR [64]	97.96	55	88.23	60	94.36	55	79.56	70
VLAD+RN [73]	97.72	50	88.04	60	93.71	55	78.90	65
Proposed Descriptor	**98.82**	60	**89.92**	70	**96.10**	60	**81.59**	70

- `VLAD+ RN` [73]: This version is similar to `VLAD+` L_2`-Norm`, except that L_2 normalization is performed separately for each residual vector $\mathbf{f}_{ij} - \boldsymbol{\mu_i}$, prior to accumulation.

With regards to the performances, our proposed descriptor does better across both the databases. The reason for the improvement is owing to the fact that it deviates from the VLAD variants by considering the proximity criterion for scoring rather than relying on the residual distortion.

2.7.4 Empirical study with a reduced version of our descriptor

Despite the proposed codebook descriptor outperforming the VLAD (discussed in Section 2.4.1) at both paragraph and text line levels, it comes at the cost of increased feature vector dimension. Note that for a codebook size M, the dimension of VLAD and our descriptor is $D \times M$ and $2 \times D \times (M-1)$ respectively. As a circumvention to this issue, we attempt at judging the writer identification performance by utilizing the codebook descriptor with a lower dimension as follows:

We consider only the contributions from the $\tilde{S}_i^+(d)$ scores of each of the D attributes in the $M-1$ codevectors $\{\boldsymbol{\mu_i}\}_{i=1}^{M-1}$. Note that, we could have as well considered only the scores $\tilde{S}_i^-(d)$ for making the $D \times (M-1)$ dimensional codebook descriptor. The performance of the resulting

Table 2.6: Summary of the average writer identification rates (in %) for the best performing codebook size corresponding to the different descriptors. Note, that for our work, $D = 7$, as mentioned in Section 2.3.

Descriptor	Dimension	Paragraph Level		Text line Level	
		IR	M	IR	M
VLAD	$D \times M$	97.31	55	87.46	60
Proposed	$2 \times D \times (M-1)$	98.82	60	89.92	70
Proposed	$D \times (M-1)$	98.47	65	89.10	70

(a) IAM Database

Descriptor	Dimension	Paragraph Level		Text line Level	
		IR	M	IR	M
VLAD	$D \times M$	92.76	65	78.16	60
Proposed	$2 \times D \times (M-1)$	96.10	60	81.59	70
Proposed	$D \times (M-1)$	95.12	60	80.57	65

(b) IBM-UB1 Database

$D \times (M-1)$ codebook descriptor was then evaluated for varying sizes of codebook. However, we present only the best average identification rates obtained along with the corresponding codebook size in Table 2.6 across all the databases for the paragraph and text-line level respectively.

The reduced dimension variant of our proposal works better than the VLAD. Likewise, when comparing to the full dimension $2 \times D \times (M-1)$ descriptor, we see that it trails behind in performance by up-to around 1%. Nevertheless, this trend may be acceptable, owing to the fact that the reduced dimension descriptor does not contain the complete feature description described in Section 2.4.2 for the handwritten paragraph / text under consideration.

Even though comparable identification rates are reported with large-size codebooks, the $D \times (M-1)$ reduced dimension descriptor gets outperformed by the one with $2 \times D \times (M-1)$ dimension at smaller sizes. To validate further, we present the performance trend for varying

Table 2.7: Average paragraph level writer identification rates (in %) on the IAM obtained for varying codebook sizes with the different descriptors.

Codebook size M	VLAD ($D \times M$ dimension)	Proposed descriptor ($2 \times D \times (M-1)$ dimension)	Proposed descriptor ($D \times (M-1)$ dimension)
5	88.96	90.62	84.85
10	92.03	94.12	92.46
15	93.85	95.97	94.31
20	94.93	96.66	95.53
25	95.58	97.38	96.30
30	96.14	97.92	97.04
35	96.96	97.82	97.20
40	97.02	98.11	97.44
45	97.11	98.32	97.70
50	97.19	98.45	97.97
55	97.31	98.63	98.22
60	97.24	98.82	98.38
65	97.18	98.68	98.47
70	97.10	98.56	98.28

number of codebook sizes (less than 20) at the paragraph level (Table 2.7) for the IAM Database. We see that for the codebook comprising five codevectors, the proposed $D \times (M-1)$ dimension descriptor performs worse than the VLAD. This degradation in performance at small codebook sizes is largely attributed to the omission of the $\tilde{S}_i^-(d)$ scores during the construction of the descriptor. A small codebook size implies relatively larger Voronoi cells and the process of this omission leads to a prodigious loss of information thereby lowering the identification rate. However, this problem of the reduced $D \times (M-1)$ descriptor is alleviated at higher codebook sizes (values of M greater than 20) due to the relatively finer Voronoi regions that help in a better discrimination between writers.

To summarize the preceding discussion, either of the proposed descriptors of dimension $D \times (M-1)$ or $2 \times D \times (M-1)$ would suffice in providing a comparable performance at large size codebooks. However for small codebook sizes, the higher dimensional descriptor does better in discrimination as can be inferred from the entries in Table 2.7.

We wish to state that though the above discussion was presented for the IAM, we nonetheless see a similar trend across the IBM-UB1 database - with either of the descriptors as a choice at higher sizes of codebook.

2.7.5 Performance with a variant of writer descriptor

Recall that the derivation of the writer descriptor in Section 2.4.2 is formulated in a way that, for a given codebook size M, we employ the descriptors of only $M-1$ codevectors to construct the final description for the online handwritten document. In addition, we also explored a variant of the descriptor, whereby the descriptors from all the M codevectors are considered for describing the handwritten document. This formulation is based on considering only the scores of the attributes of the feature vectors assigned to a particular codevector.

Mathematically speaking, the normalized score of the d^{th} attribute ($1 \leq d \leq D$) for the i^{th} codevector $\boldsymbol{\mu_i}$ is computed as:

$$\tilde{S}_i^+(d) = \frac{\sum_{j=1}^{n_i} S_{ij}^+(d)}{\sum_{j=1}^{n_i} S_{ij}^+(d) + \sum_{j=1}^{n_i} |S_{ij}^-(d)|}$$
$$\tilde{S}_i^-(d) = \frac{\sum_{j=1}^{n_i} |S_{ij}^-(d)|}{\sum_{j=1}^{n_i} S_{ij}^+(d) + \sum_{j=1}^{n_i} |S_{ij}^-(d)|} \qquad (2.13)$$
$$1 \leq d \leq D, \qquad 1 \leq i \leq M$$

Subsequent to obtaining the above scores, the descriptor is computed by following steps similar to those described in Equations 2.6 and 2.7. The final descriptor computed across all the M codevectors leads to a writer descriptor of dimension is $2 \times D \times M$.

The evaluation of the above descriptor at the paragraph and text line levels for the IAM and IBM-UB1 databases are presented in Table 2.8 (a) and (b). For sake of brevity, we refer to it as the '$2 \times D \times M$-dim', while the one discussed in Section 2.4.2 is abbreviated to as '$2 \times D \times (M-1)$-dim'. The best average writer identification rate along with the codebook size M is mentioned in the Table.

Table 2.8: Performance comparison with a variant of the writer descriptor on both the databases. The numbers being mentioned are the identification rates (in %)

Descriptor Type	Paragraph Level		Text line Level	
	IR	M	IR	M
$2 \times D \times M$-dim	97.11	65	87.38	70
$2 \times D \times (M-1)$-dim	98.82	60	89.92	70

(a) IAM Database

Descriptor Type	Paragraph Level		Text line Level	
	IR	M	IR	M
$2 \times D \times M$-dim	94.33	60	79.11	70
$2 \times D \times (M-1)$-dim	96.10	60	81.59	70

(b) IBM-UB1 Database

It can be noted that the '$2 \times D \times (M-1)$-dim' descriptor fares better than the '$2 \times D \times M$-dim' across both the databases. The reason for the higher writer identification rates is owing to the formulation of the former that considers the scores from the feature vectors of the entire document. Contrary to it, for the variant '$2 \times D \times M$-dim', the contribution of the scores for the generation of the descriptor of a given codevector is limited to only a subset of features assigned to it (refer Equation 2.13).

2.7.6 Statistical Significance

In this subsection, we demonstrate that the results of both the proposed descriptor ($2 \times D \times (M-1)$ dimension) and its reduced dimension version ($D \times (M-1)$) is statistically significant when compared to the VLAD. The statistical check has been performed using the Student's t-test for a significance level of 0.05. Table 2.9 outlines the p-values obtained on the databases - IAM and IBM-UB1 at paragraph and text-line level. A trend of low values for p is empirically observed across each of the entries in the Table, thereby indicating statistical significance of our proposals.

Table 2.9: Statistical significance of the performance of the proposed codebook descriptors over the VLAD obtained via the Student's t-test.

Database	Codebook descriptor dimension	Paragraph Level	Text line level
IAM	$2 \times D \times (M-1)$	1×10^{-3}	6.7×10^{-3}
	$D \times (M-1)$	4.2×10^{-2}	1.07×10^{-2}
IBM-UB1	$2 \times D \times (M-1)$	2.86×10^{-8}	4.87×10^{-10}
	$D \times (M-1)$	4.95×10^{-5}	3.68×10^{-6}

2.8 Conclusion

In this Chapter, we proposed the use of codebook descriptors for the problem of writer identification. We regard our research framework to that of a retrieval problem and adapt the so called codebook based Vector of Locally Aggregated Descriptor (VLAD) that has been promising for the object retrieval application in image processing. The codebook comprises a set of code vectors with associated Voronoi cells computed from a k-means clustering algorithm on a set of feature vectors along the online trace. However, we show that the VLAD formulation at times, cannot effectively discriminate between writers, when their respective feature vectors are not linearly separable in the Voronoi cell of the code vectors. To overcome this problem, we propose a novel descriptor that improves upon the VLAD formulation. A notable aspect is that given a codebook of size M, we consider the descriptors of only $M-1$ codevectors to construct the final descriptor by concatenation.

In addition, for constructing the codebook and subsequently the descriptor, we extracted point-based features by incorporating a so called gap parameter. Several experiments are performed on the IAM and IBM-UB1 databases to illustrate the promising nature of our approach.

3

Exploration of sparse coding based descriptors

Contents

3.1 Introduction

Recall from the previous Chapter that the codebook used for formulating the writer descriptor is generated by applying the k-means algorithm. This however comes with a limitation that each feature vector gets assigned to only one prototype. As an alleviation to this issue, we consider proposing the descriptors with respect to an over-complete dictionary obtained via a sparse representation framework. Such a strategy ensures that the feature vectors are expressed as a linear combination of atoms that serve as prototypes.

To the best of our knowledge, the use of sparse coding approaches for online writer identification have not been addressed till date in the literature. Nonetheless, it may be mentioned that there has been only one attempt till date on sparse coding with regards to identifying the authorship of offline handwritten documents [25]. The traditional pipeline followed in that work involve pooling the sparse coefficients over the dictionary atoms by an averaging or maximum operation. Though the average pooling technique is shown to work satisfactorily for identification, we do believe that there is still room for improvement. Accordingly, in the remainder of the thesis, we focus our research to exploring additional information from the sparse coefficients, that can provide useful cues to model the dynamic characteristics of the writer.

As a starting step to the above direction, we propose in this Chapter a methodology to encode the sub-strokes of the online handwritten trace with descriptors derived from a sparse coding framework. The usage of sparse representation stems from the flexibility it offers over hard clustering algorithms in terms of representing the segmented handwritten sub-stroke of a writer as a combination of more than one prototype / atom. The number of prototypes used for representation is constrained by the sparsity condition.

Our idea focusses at capturing the similarity of the attributes [1] of each individual sub-stroke to the corresponding value in the subset of dictionary atoms that contribute with a non-zero sparse coding coefficient. Keeping this in perspective, we derive descriptors from each of the dictionary atoms, which incorporate the similarity scores of the attributes in a feature vector.

[1] We refer to the feature values in a feature vector as 'attributes'. The attributes computed for each sub-stroke of the online trace are stacked to form a feature vector.

The scores, as we shall see in Section 3.6, are in a way, indicative of the reconstruction error obtained while employing a particular dictionary atom alone for reconstruction. The idea of scoring individual feature attributes separately with regards to a dictionary atom is achieved by utilizing the corresponding non-zero sparse coefficient.

It is worth noting that the work of [25] on sparse coding primarily rely on the accumulation of the coefficients over the set of dictionary atoms for generating the descriptor. In contrast, the present proposal takes a step forward in exploring additional information from the sparse coefficients for writer identification. Their utilization in separately scoring each of the individual attributes in a feature vector helps in capturing the description of the writer at a finer level. This in turn leads to improvements in performance of the writer identification system.

In addition to the above contribution, we consider in Section 3.4, several sets of histogram based attributes / features at the sub-stroke level for constructing the dictionary atoms and subsequently the descriptors. Such a representation results in a fixed feature vector dimension irrespective of the length of the sub-stroke. We provide a thorough and comprehensive analysis of the histogram based attributes by addressing the issue of the selection of the bin size to be used and their relation to the writer identification rate. More specifically, we **propose** in Section 3.4.1, an entropy based strategy for determining the appropriate bin size to be selected for the generation of these histograms. Considering that the descriptors of the sparse coding framework are constructed from the resulting histogram feature sets, the selection of the bin size using our method ensures a good discrimination between the writers. This also gets reflected in the experimental results, where we show a dependence between the bin size chosen by the entropy based analysis and the corresponding writer identification performance.

3.2 Proposed Methodology

An overview of the proposed writer identification system is provided in Figure 3.1. The spatio-temporal sequence consisting of the spatial coordinates and the time duration from a document is initially passed through the pre-processing block. The output of this module is then used to segment the strokes into smaller fragments called sub-strokes, following which

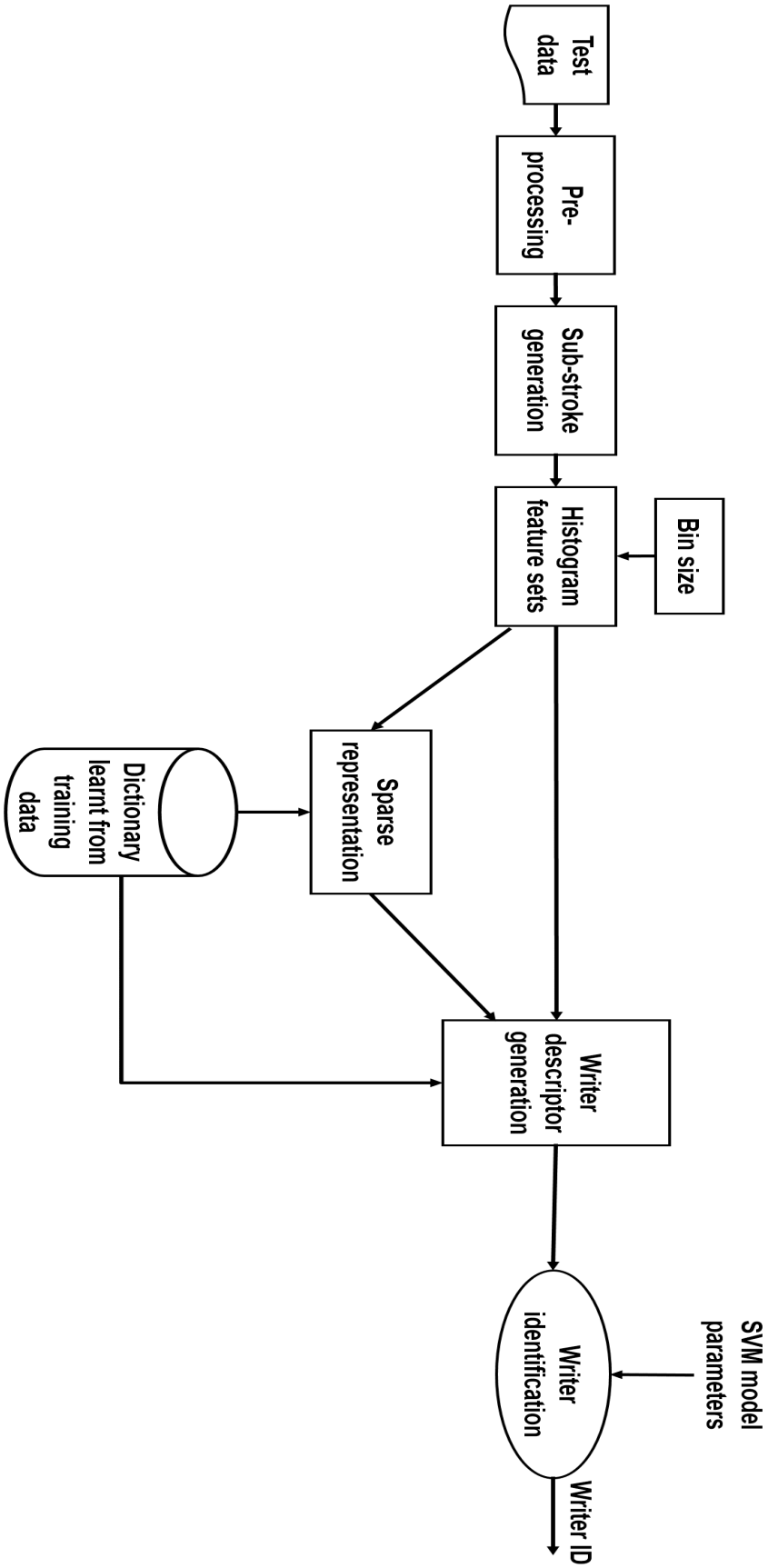


Figure 3.1: Block diagram of the proposed online writer identification system that employs descriptors from a sparse coding framework. The test data refer to paragraphs or text lines written by the enrolled writers.

histogram based feature sets parametrized by a bin size are extracted for each of them. The atoms of a pre-learnt dictionary are then used in conjunction with the attributes of sub-stroke feature vectors to generate a descriptor, that encapsulates the writer specific characteristics of the document. Finally, a Support Vector Machine (SVM) classifier is utilized to establish the authorship of the handwritten content.

The organization of the remainder of this Chapter is as follows: We begin by outlining the sub-stroke generation process in Section 3.3. Following this, the histogram based feature sets along with an entropy based analysis for the appropriate bin size is presented in Section 3.4. An overview of the sparse coding framework for online writer identification is then provided in Section 3.5. The sub-stroke feature vectors from a document together with their corresponding sparse-coded coefficients are utilized in deriving the writer descriptor in Section 3.6. The efficacy of our proposal is demonstrated by several experiments in Section 3.7.

3.3 Sub-stroke generation

We begin by applying the sequence of preprocessing steps presented in Subsection 2.2. Thereafter, each stroke is divided into fragments called sub-strokes, that are in a way, indicative of the basic structural units present in a document. The segmentation procedure utilized in this work considers the generation of sub-strokes obtained via two strategies as outlined below: In the first method, we rely on locating the local maxima in a stroke. The strokes are then split at these points and individual sub-strokes obtained thereof. In the second strategy, we consider sub-strokes of a fixed length by splitting the stroke into smaller segments containing a specific number of points. The successive segments are allowed to have an overlap that is governed by a stride factor. For our work, we have considered a sub-stroke to contain 30 points with a stride of five points. The generation of sub-strokes using the segmentation strategies may be regarded as a data augmentation process, that facilitates the learning of the sparse representation framework on their corresponding feature vectors discussed in the next section.

As an illustration, we depict in Figure 3.2 (a), an online trace of a word and the sub-strokes obtained using the two strategies in sub-figures (b) and (c) respectively. For better clarity,

(a) (b) (c)

Figure 3.2: Illustration of the segmentation approach - (a) Input online handwritten trace. (b) Substrokes obtained by locating the local maxima in a stroke (c) Substrokes generated with considering fixed length sample points. Note that we use different colours to distinguish between subsequent sub-strokes

different colours are employed to distinguish between subsequent sub-strokes.

3.4 Feature extraction

Subsequent to the generation of the sub-strokes, we extract features that encode their shape information, thereby providing discriminative cues for a writer. Inspired from the success of Histogram of Oriented Gradients [74] in the area of image processing, we propose sets of histogram based features / attributes for each sub-stroke as follows:

Consider a sub-stroke S of N_S points denoted by $\{\mathbf{p}_i\}_{i=1}^{N_S}$ where $\mathbf{p}_i = (x_i, y_i)$ consists of the spatial x and y coordinates of the i^{th} point. Let $\mathbf{p}_c = (x_c, y_c)$ be the coordinates of its centroid. We calculate $\theta_{\mathbf{p}_i}$, the angle that the line segment $\mathbf{p}_i - \mathbf{p}_c$ makes with the horizontal and the Euclidean distance $d_{\mathbf{p}_i}$ between $\mathbf{p}_i$ and $\mathbf{p}_c$ as follows:

$$\begin{aligned} \theta_{\mathbf{p}_i} &= \tan^{-1}\left\{\frac{(y_i - y_c)}{(x_i - x_c)}\right\} \\ d_{\mathbf{p}_i} &= \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} \end{aligned} \tag{3.1}$$

Once the $(\theta_{\mathbf{p}_i}, d_{\mathbf{p}_i})$ pairs are evaluated for each point of the sub-stroke, we generate a histogram comprising B bins. The value of B decides the extent to which the angular values are quantized into. To start with, the votes corresponding to the bins are initialised to zero. For each $\mathbf{p}_i$, we increment the vote of the bin in which $\theta_{\mathbf{p}_i}$ falls into with a magnitude $d_{\mathbf{p}_i}$. In this manner, the accumulation of the votes across the bins is done for all the N_S points of S under consideration. The resulting operation gives us an B dimensional vector, which is then

normalized by its Euclidean norm to give $\mathbf{h_p}$.

Apart from the above, we generate two additional sets of histograms that take into the account the information of the vicinity around the sample points of a sub-stroke. Accordingly, for each point $\mathbf{p}_i$ in S, we employ a set of $2r+1$ points $\{\mathbf{p}_{i-r}, ..., \mathbf{p}_i, ..., \mathbf{p}_{i+r}\}$ in its neighbourhood $\mathbb{N}_{\mathbf{p}_i}$ and compute the following:

$$\begin{aligned} \mathbf{\Delta_i} &= \frac{1}{2r+1}\sum_{j=-r}^{r} j\mathbf{p}_{i+j} \quad \forall i \in [r+1, N_S - r] \\ \mathbf{a_i} &= \frac{1}{2r+1}\sum_{j=-r}^{r} j\mathbf{\Delta_{i+j}} \quad \forall i \in [2r+1, N_S - 2r] \end{aligned} \tag{3.2}$$

The value of r determines the number of sample points in $\mathbb{N}_{\mathbf{p}_i}$ used for obtaining the above values at $\mathbf{p}_i$. In our work, the value of r was set to two. This choice, as such is reasonable, considering that we do not have to lose more than four points at the start and end of each stroke when compared to a higher value of r.

Both $\mathbf{\Delta_i}$ and $\mathbf{a_i}$, corresponding to point $\mathbf{p}_i$, encode the change in the spatial coordinates and the rate of change respectively. It may be mentioned that the form of Equation 3.2 bears semblance to 'delta' and 'acceleration' coefficients popularly used in the research of speech processing [75]. With regards to our work, these consist of two components represented by $\mathbf{\Delta_i} = (\Delta_{ix}, \Delta_{iy})$ and $\mathbf{a_i} = (a_{ix}, a_{iy})$ corresponding to the spatial x and y coordinates respectively. Accordingly, we refer to them as delta and acceleration vectors respectively. By noting that the vectors encode the relative spatial information, we utilize them to compute their orientation and magnitude as follows.

$$\begin{aligned} \theta_{\mathbf{\Delta_i}} &= \tan^{-1}\left\{\frac{\Delta_{iy}}{\Delta_{ix}}\right\} \\ d_{\mathbf{\Delta_i}} &= \sqrt{(\Delta_{ix})^2 + (\Delta_{iy})^2} \end{aligned} \tag{3.3}$$

$$\begin{aligned} \theta_{\mathbf{a_i}} &= \tan^{-1}\left\{\frac{a_{iy}}{a_{ix}}\right\} \\ d_{\mathbf{a_i}} &= \sqrt{(a_{ix})^2 + (a_{iy})^2} \end{aligned} \tag{3.4}$$

Analogous to the generation of $\mathbf{h_p}$, a voting procedure for the B bins of the histogram is applied for $(\theta_{\mathbf{\Delta_i}}, d_{\mathbf{\Delta_i}})$ and $(\theta_{\mathbf{a_i}}, d_{\mathbf{a_i}})$ pairs to obtain $\mathbf{h_\Delta}$ and $\mathbf{h_a}$ respectively. Thereafter, the vectors $\mathbf{h_p}$, $\mathbf{h_\Delta}$ and $\mathbf{h_a}$ are concatenated to provide the feature vector description $\mathbf{f}_S$ for S.

$$\mathbf{f}_S = [\mathbf{h_p}, \mathbf{h_\Delta}, \mathbf{h_a}] \tag{3.5}$$

It can be noticed that the dimension D of $\mathbf{f}_S$ is $3 \times B$. Accordingly, for a document consisting of N_T sub-strokes, we denote the sequence of feature vectors as $\{\mathbf{f}_j\}_{j=1}^{N_T}$, where each $\mathbf{f}_j$ is of dimension $3 \times B$.

As a pictorial representation, we present in Figure 3.3, a segmented sub-stroke and its corresponding histograms, namely $\mathbf{h_p}$, $\mathbf{h_\Delta}$ and $\mathbf{h_a}$. The plots are constructed using ten bins, thereby resulting in a 30 dimensional feature representation.

3.4.1 Entropy based selection of bin size B

From the preceding discussion, it is evident that the dimension of the feature representation derived from the segmented sub-strokes is reliant on the number of bins B used for generating the set of histograms. Further to this, from the work-flow of Figure 3.1, it can be seen that these are utilized subsequently to derive the writer descriptors. Hence, it becomes imperative to select an appropriate value of B, so that increased discrimination of descriptors between writers can be achieved. As a step towards making this choice, we propose a scheme based on analysing the values of entropy, obtained subsequent to a two level clustering scheme as detailed below:

In the first level, feature vectors corresponding to the sub-strokes are pooled among the set of paragraphs of the W writers and partitioned into k_1 clusters. The resulting prototypes relate on an average, to the shape characteristics of the frequently occurring sub-strokes. In the second level, we partition the feature vectors assigned to each of the k_1 clusters separately into W clusters - corresponding to the number of writers, whose data has been enrolled for training. For the sake of clarity, we refer to each of the W clusters at the second level as a 'sub-cluster'. Thus, in total, we obtain $k_1 \times W$ sub-clusters at the second level.

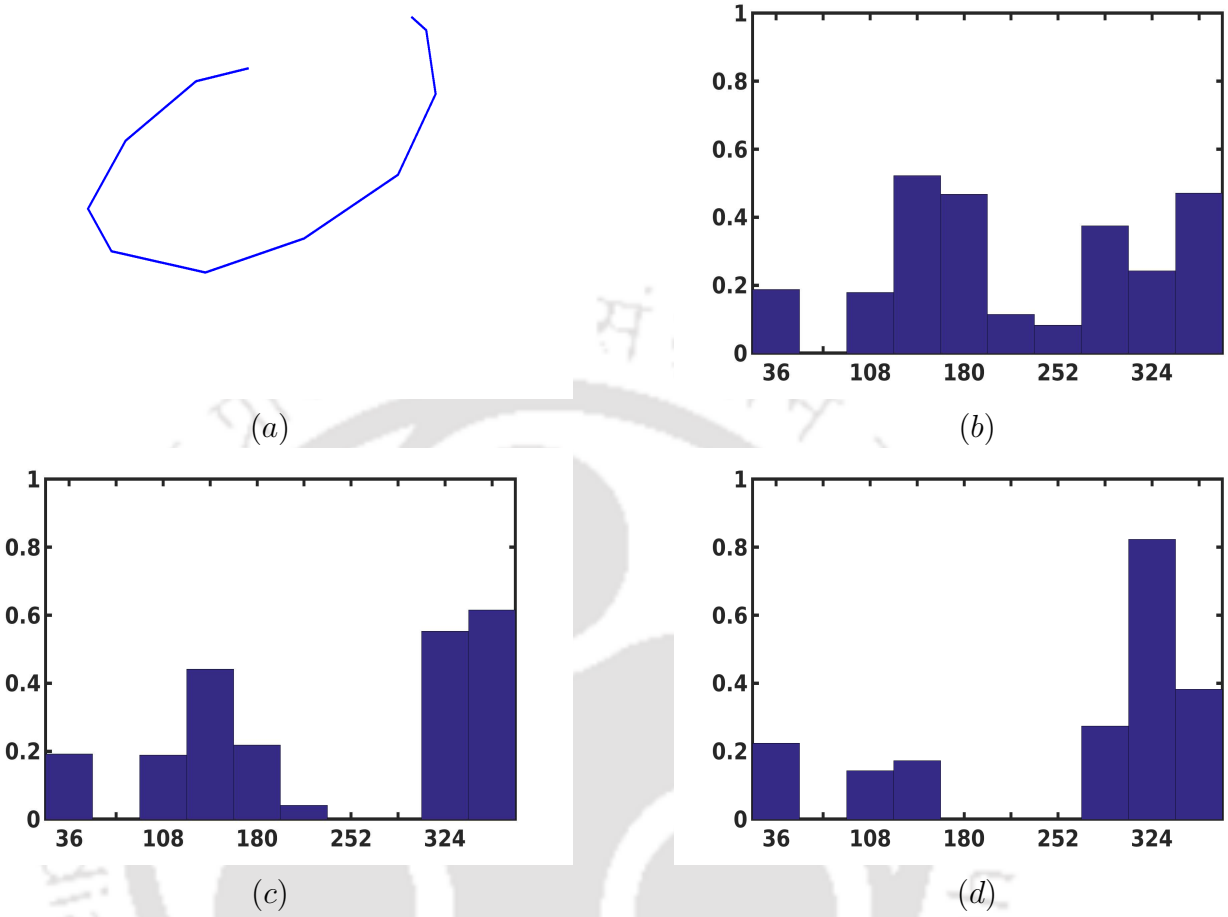


Figure 3.3: The sub-figure (a) shows a sub-stroke of a particular writer. The sub-figures (b), (c) and (d) are the histograms $\mathbf{h_p}$, $\mathbf{h_\Delta}$ and $\mathbf{h_a}$ respectively corresponding to this sub-stroke. The number of bins B used for generating these plots is set to ten.

The fraction of feature vectors corresponding to each of the W writers in a given sub-cluster can now be used to compute the entropy measure as:

$$H_{ik} = \sum_{j=1}^{W} -p_{ik}^{j} \log_2(p_{ik}^{j}) \qquad 1 \leq i \leq k_1, \quad 1 \leq k \leq W \tag{3.6}$$

From the preceding definition, the entropy values corresponding to the W sub-clusters generated from the i^{th} cluster can be written as $\{H_{ik}\}_{k=1}^{W}$. Clearly, the calculation of a specific H_{ik} in Equation 3.6 entails obtaining the probability values for the W writers, namely $\{p_{ik}^{j}\}_{j=1}^{W}$.

To motivate the use of entropy measure for the selection of the bin size B, consider the scenario wherein the feature vectors assigned to the k^{th} sub-cluster encapsulate the complete

information from only the n^{th} writer, $1 \leq n \leq W$. Assuming that the sub-clusters have been generated from the i^{th} cluster, we can infer that the value of p_{ik}^{j} will be one for $j = n$ and zero otherwise. Accordingly from Equation 3.6, we see that the entropy value H_{ik} will achieve a value of zero.

As an extension of the above, when each of the $k_1 \times W$ sub-clusters generated in the second level comprise feature vectors corresponding to only one writer, the average entropy measure defined by

$$H = \frac{1}{k_1 \times W} \sum_{i=1}^{k_1} \sum_{k=1}^{W} H_{ik} \tag{3.7}$$

will again be zero. That being said, we can thus infer that lower entropy may be regarded as a measure to represent better separability of the feature vectors across the writers. Lastly, by recalling that the feature set of the segmented sub-strokes themselves are parametrized with a bin size B, we can thus explicitly refer to the average entropy value obtained in Equation 3.7 by H_B. The choice value for B (denoted as B^*) that can be considered is as follows:

$$B^* = \arg\min_{B} H_B \tag{3.8}$$

For a better understanding of the preceding discussion, we provide a toy example illustration in Figure 3.4 with three writers enrolled to the identification system. The feature vectors of the writers are depicted with different symbols. In the sub-figure (a), we depict the k_1 clusters obtained at the first level with their corresponding Voronoi cells. In this example, the value of k_1 used is 20. Thereafter, the feature vectors in a given Voronoi cell are used to generate $W = 3$ sub-clusters at the second level. The value for W corresponds to the number of writers enrolled in the system. The sub-clusters of one such Voronoi cell, highlighted with a yellow color in Figure 3.4 (a) is shown in sub-figure (b). The entropy corresponding to the distribution of the writers in each of the three sub-clusters are computed separately and then accumulated. Likewise, this process of accumulation is carried out across all the sub-clusters in the feature space, thus leading to the value of the average entropy H_B (refer Equation 3.7). For sake of completion, in the sub-figure 3.4 (b), we also present the probability values corresponding to the writers of a sub-cluster (indicated with an arrow).

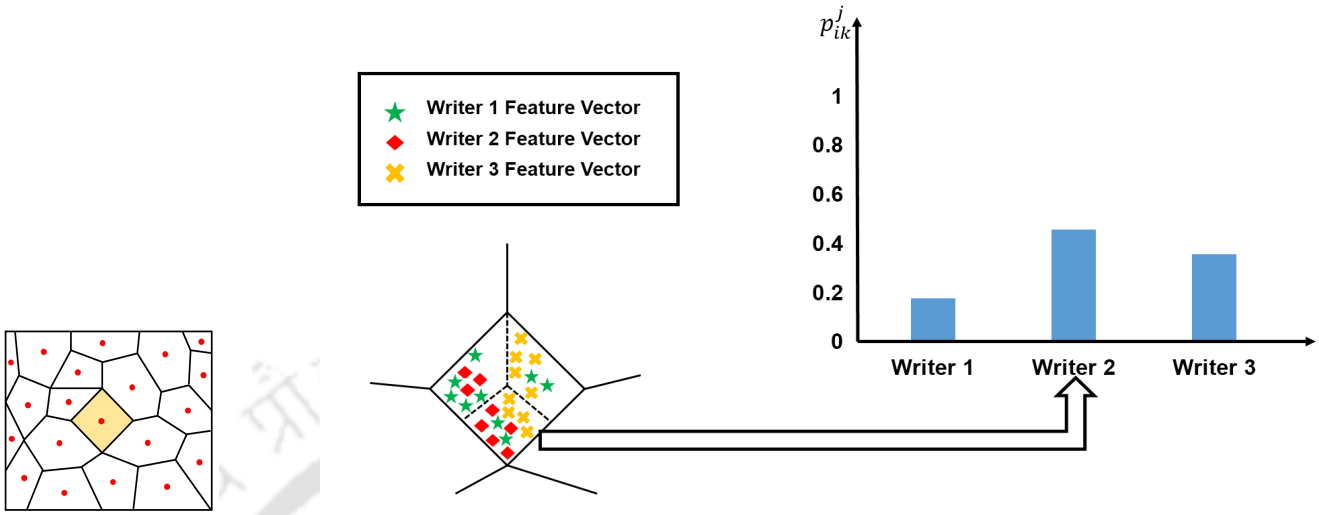


Figure 3.4: A toy-example for the computation of H_B - (a) Illustration of the first level of $k_1 = 20$ clusters obtained from the feature vectors of the three writers with their prototypes and Voronoi cells highlighted. (b) Depiction of the three sub-clusters for a particular Voronoi cell shaded in yellow color in sub-figure (a). In addition, we also present the computation of $\{p_{ik}^{j}\}_{j=1}^{3}$ (probability values corresponding to the three writers) from the sub-cluster marked with an arrow.

3.5 Sparse coding: an overview

The sparse framework aims at efficiently determining an over-complete set of basis vectors, in a way so that each feature vector can be approximated closely as a linear combination of them. The term 'sparse' comes from the fact that most of the coefficients obtained while representing a feature vector are close to zero.

In our work, we consider that a document with N_T feature vectors $\{\mathbf{f}_j\}_{j=1}^{N_T}$ and a pre-learnt over-complete dictionary $\mathbf{D}$ of M atoms, $[\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, ..., \boldsymbol{\phi}_M]$ are available to us. Any feature vector $\mathbf{f}_j$ of a segmented sub-stroke can then be closely represented / approximated as a linear combination of the atoms $\boldsymbol{\phi}_i$, $1 \leq i \leq M$ with a set of coefficients $\boldsymbol{\alpha}_j = [\alpha_{1j}\ \alpha_{2j}\ \cdots\ \alpha_{Mj}]^T$ as follows:

$$\mathbf{f}_j \approx \sum_{i=1}^{M} \alpha_{ij} \boldsymbol{\phi}_i \tag{3.9}$$

Since M is higher than D (the dimension of $\mathbf{f}_j$), there are infinitely large solutions for Equation 3.9. In order to obtain the unique set of coefficients, we consider a sparsity constraint

that ensures that only a few of the M coefficients in $\boldsymbol{\alpha}_j$ are non zero. For a pre-learnt dictionary $\mathbf{D}$ of atoms, the values of these coefficients can be obtained by solving the following optimization formulation:

$$\min_{\boldsymbol{\alpha}_j} \parallel \mathbf{f}_j - \sum_{i=1}^{M} \alpha_{ij}\boldsymbol{\phi}_i \parallel_2 + \lambda \parallel \boldsymbol{\alpha}_j \parallel_1 \tag{3.10}$$

The factor λ is a Lagrange multiplier which controls the influence that the two terms have on each other.

With regards to the estimation / learning of the dictionary $\mathbf{D}$ in the training phase, we use the algorithm of [76] implemented in the SPArse Modeling Software (SPAMS) optimization toolbox.

Traditionally, the descriptor used for writer identification involve a max or average pooling of sparse coefficients [25]. For an over-complete dictionary of M atoms, and a document comprising the feature vectors of N_T sub-strokes, we construct a matrix of size $N_T \times M$. The $(i,j)^{th}$ element of this matrix contains α_{ij}, the sparse coefficient from the j^{th} sub-stroke corresponding to the i^{th} dictionary atom $\boldsymbol{\phi}_i$. The average pooling strategy of generating the writer descriptors from the sparse coefficients can be defined by a vector $\mathbf{z}$ of size M, whose i^{th} element is given as :

$$z_i = \frac{1}{\sum_{j=1}^{N_T} \mathcal{I}(\alpha_{ij})} \sum_{j=1}^{N_T} |\alpha_{ij}| \tag{3.11}$$

where $\mathcal{I}(\alpha_{ij})$ is an indicator variable denoted by:

$$\mathcal{I}(\alpha_{ij}) = \begin{cases} 1 & |\alpha_{ij}| > 0 \\ 0 & Otherwise \end{cases} \tag{3.12}$$

Likewise, for the traditional max pooling strategy, we have:

$$z_i = \max_{1 \le j \le N_T} |\alpha_{ij}| \tag{3.13}$$

In the following section, we propose descriptors that aim to capture additional information from the sparse coefficients, that in a way, surpass the performance of the traditional max /average pooled strategies

3.6 Proposed sparse coding based writer description

Prior to deriving the descriptor, the feature vectors $\{\mathbf{f}_j\}_{j=1}^{N_T}$ extracted from the N_T sub-strokes, together with their respective sparse coefficient vectors $\{\boldsymbol{\alpha}_j\}_{j=1}^{N_T}$ are assumed to be known. Let the subset of feature vectors in $\{\mathbf{f}_j\}_{j=1}^{N_T}$ that have a non-zero sparse coefficient α_{ij} for the i^{th} dictionary atom be represented as $\{\mathbf{f}_{ij}\}_{j=1}^{n_i}$. Without loss of generality, we have $n_i \leq N_T$.

We define two scores $S_{ij}^{+}(d)$ and $S_{ij}^{-}(d)$ for the d^{th} attribute of the j^{th} feature vector $\mathbf{f}_{ij}$ having a non-zero sparse coding coefficient α_{ij} for the i^{th} dictionary atom $\boldsymbol{\phi}_i$.

$$S_{ij}^{+}(d) = \begin{cases} \frac{1}{1+|f_{ij}(d)-\alpha_{ij}\times\phi_i(d)|} & f_{ij}(d) \geq \alpha_{ij} \times \phi_i(d) \\ 0 & otherwise \end{cases}$$
$$S_{ij}^{-}(d) = \begin{cases} \frac{-1}{1+|f_{ij}(d)-\alpha_{ij}\times\phi_i(d)|} & f_{ij}(d) < \alpha_{ij} \times \phi_i(d) \\ 0 & otherwise \end{cases} \tag{3.14}$$
$$1 \leq i \leq M, \quad 1 \leq j \leq n_i, \quad 1 \leq d \leq D$$

Here $\phi_i(d)$ is the value of the d^{th} attribute ($1 \leq d \leq D$) in the atom $\boldsymbol{\phi}_i$. In a sense, the Equation 3.14 attempts to capture the error obtained along the d^{th} attribute while considering a single dictionary atom for reconstruction. If we define this error as $e_{ij}(d) = f_{ij}(d) - \alpha_{ij} \times \phi_i(d)$, then with regards to the attribute $f_{ij}(d)$ for which $e_{ij}(d) \geq 0$, we assign it with the score $S_{ij}^{+}(d)$. Likewise, when $e_{ij}(d) < 0$, this attribute is given a score of $S_{ij}^{-}(d)$. In particular, we observe that the more proximal $f_{ij}(d)$ is to $\alpha_{ij} \times \phi_i(d)$, the higher is the value of the score $S_{ij}^{-}(d)$ or $S_{ij}^{+}(d)$ (as applicable).

The obtained scores across the d^{th} feature attribute of $\{\mathbf{f}_{ij}\}_{j=1}^{n_i}$ are subsequently accumulated and normalized with regards to the values obtained from the entire document to obtain $\tilde{S}_i^{+}(d)$

and $\tilde{S}_i^-(d)$.

$$\tilde{S}_i^+(d) = \frac{\sum_{j=1}^{n_i} S_{ij}^+(d)}{\sum_{k=1}^{M}\sum_{j=1}^{n_k} S_{kj}^+(d)}$$
$$\tilde{S}_i^-(d) = \frac{\sum_{j=1}^{n_i} S_{ij}^-}{\sum_{k=1}^{M}\sum_{j=1}^{n_k} S_{kj}^-(d)} \qquad (3.15)$$
$$1 \leq i \leq M, \quad 1 \leq d \leq D$$

The scores for each of the D feature attributes are stacked to form the descriptors $\mathbf{S}_i^+$ and $\mathbf{S}_i^-$ as shown:

$$\mathbf{S}_i^+ = \begin{bmatrix} \tilde{S}_i^+(1) & \cdots & \tilde{S}_i^+(d) & \cdots & \tilde{S}_i^+(D) \end{bmatrix}^T$$
$$\mathbf{S}_i^- = \begin{bmatrix} \tilde{S}_i^-(1) & \cdots & \tilde{S}_i^-(d) & \cdots & \tilde{S}_i^-(D) \end{bmatrix}^T \qquad (3.16)$$

To ensure compactness of the descriptor for the i^{th} dictionary atom, we consider the L_2 norm representation of the $\mathbf{S}_i^+$ and $\mathbf{S}_i^-$ vectors.

$$\mathbf{S}_i = [\| \mathbf{S}_i^+ \|_2, \| \mathbf{S}_i^- \|_2]^T \qquad (3.17)$$

The final descriptor is obtained by concatenating the descriptors across all the M dictionary atoms.

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_1 & \mathbf{S}_2 & \cdots & \mathbf{S}_M \end{bmatrix}^T \qquad (3.18)$$

We see that the dimension of $\mathbf{S}$ is $2 \times M$. In the event of a dictionary atom $\boldsymbol{\phi}_i$ not having any of the points in the document with a non-zero sparse coding coefficient, its descriptor $\mathbf{S}_i$ is set to the zero vector. To summarize our strategy, the pseudo-code of our proposed descriptor is shown in Algorithm 2.

3.6.1 Discussion

From Equation 3.17, it can be observed that we considered stacking the L_2 norm of the $\mathbf{S}_i^+$ and $\mathbf{S}_i^-$ vectors for constructing $\mathbf{S}_i$ rather than the entire vectors themselves. The rationale behind this step is to construct a compact descriptor for a document in terms of feature dimen-

Algorithm 2 Proposed writer descriptor generation

- **Input:**
  - A set of histogram based feature vectors $\{\mathbf{f}_j\}_{j=1}^{N_T}$ extracted from a document with N_T sub-strokes.
  - Dictionary with M atoms $\{\boldsymbol{\phi}_i\}_{i=1}^{M}$.
- **Output:**
  - Writer descriptor $\mathbf{S}$.
- **Algorithm:**

Compute:
- $\boldsymbol{\alpha}_j$ - the sparse coding coefficient vector for each feature vector $\mathbf{f}_j$.
- n_i - the number of feature vectors having non-zero sparse coding coefficient for $\boldsymbol{\phi}_i$.

% `Descriptor Computation`:

For every dictionary atom in $\{\boldsymbol{\phi}_i\}_{i=1}^{M}$
For every feature vector having non-zero sparse-coding coefficient for $\boldsymbol{\phi}_i$
Compute $S_{ij}^{+}(d)$ and $S_{ij}^{-}(d)$ scores across the D attributes
End For
End For

% `Normalization Step`:

For every dictionary atom in $\{\boldsymbol{\phi}_i\}_{i=1}^{M}$
Calculate $\tilde{S}_i^{+}(d)$ and $\tilde{S}_i^{-}(d)$ scores across the D attributes.
End For

Stack the $\tilde{S}_i^{+}(d)$ and $\tilde{S}_i^{-}(d)$ scores across the D attributes to obtain $\mathbf{S}_i^{+}$ and $\mathbf{S}_i^{-}$ vectors respectively.

Evaluate L_2 norm of the $\mathbf{S}_i^{+}$ and $\mathbf{S}_i^{-}$ vectors to get $\mathbf{S}_i$.

Concatenate $\{\mathbf{S}_i\}_{i=1}^{M}$ to represent the writer descriptor $\mathbf{S}$.

sion. If we consider the entire $\mathbf{S}_i^+$ and $\mathbf{S}_i^-$ vectors for generating $\mathbf{S}_i$, then our final descriptor would have a dimension increased by a factor of D. To ensure that the L_2 norm operation does maintain the discrimination between the writers, we conduct an experiment that determines the ratio of the average inter writer to intra writer descriptor distance, as follows:

Let $\{\mathbf{S}_{w_1 i}\}_{i=1}^{N_{w_1}}$ correspond to the final descriptors for the N_{w_1} documents written by writer w_1. The average pair-wise intra-writer City Block distance between these descriptors is denoted as $d_1(w_1)$.

$$d_1(w_1) = \frac{1}{\binom{N_{w_1}}{2}} \sum_{i=1}^{N_{w_1}} \sum_{\substack{j=1 \\ j \neq i}}^{N_{w_1}} \| \mathbf{S}_{w_1 i} - \mathbf{S}_{w_1 j} \|_1 \tag{3.19}$$

As a first step to computing the inter writer descriptor distance, we consider the average of the City-block distance obtained between the descriptors of two different writers w_1, w_2 as follows.

$$d_2(w_1, w_2) = \frac{1}{N_{w_1} \times N_{w_2}} \sum_{i=1}^{N_{w_1}} \sum_{j=1}^{N_{w_2}} \| \mathbf{S}_{w_1 i} - \mathbf{S}_{w_2 j} \|_1 \tag{3.20}$$

Subsequent to the above calculations between each pair of writers, the average inter-writer City-block distance for writer w_1 can be written as:

$$d_2(w_1) = \frac{1}{W-1} \sum_{\substack{j=1 \\ m \neq w_1}}^{W} d_2(w_1, j) \tag{3.21}$$

The ratio R of the average inter-writer to the intra-writer distance is then defined by:

$$R = \frac{\sum_{i=1}^{W} d_2(i)}{\sum_{i=1}^{W} d_1(i)} \tag{3.22}$$

The value of R is related to, in a way, the degree of separation of the writer descriptors in the feature space. In Table 3.1, we present the results on the data of the writers from the databases, namely the IAM and IBM-UB1. For this set-up, the value of bins for the histogram based features and the dictionary size was set to 10 and 400 respectively The choice behind these values are based on the discussion in subsection 3.7.2.

We considered two possible representation of the descriptors as outlined below:

Table 3.1: Illustration of the R ratio at paragraph and text line level on the IAM and IBM-UB1 databases. The full descriptor corresponds to using the entire vectors of $\mathbf{S}_i^+$ and $\mathbf{S}_i^-$ across all the dictionary atoms for representation. The compact descriptor, on the other hand, considers the L_2 norm.

Descriptor	IAM database		IBM-UB1 database	
Type	Paragraph	Text line	Paragraph	Text line
Full	1.60	1.15	2.30	1.36
Compact	1.73	1.21	2.55	1.48

- We stack the entire vector of $\mathbf{S}_i^+$ and $\mathbf{S}_i^-$, obtained across all the M dictionary atoms - thus resulting in a $2 \times D \times M$ dimensional descriptor. For sake of brevity, we refer to this as the 'full descriptor'.
- We stack the L_2 norm of the $\mathbf{S}_i^+$ and $\mathbf{S}_i^-$ vectors, obtained across all the M dictionary atoms. We refer to this as the 'compact descriptor'.

With the above two representations, we obtain a comparable value of the ratio R across both the databases at paragraph and text-line levels respectively. This in turn suggests that either of the descriptors may be used for writer identification. Nonetheless, employing L_2 norm leads to a reduction in the dimension by a factor D over the full descriptor.

Following the generation of the descriptor by employing Equations 3.14 to 3.18, a one versus all SVM with the Radial Basis Function (RBF) kernel is used to establish the identity of the writer for the test document.

3.7 Results and Discussion

In this Section, we demonstrate several experiments, that reflect the various aspects of the proposal, being explored for establishing the authorship of online handwritten documents. With regards to the training and evaluation protocols, we employ the same as discussed in Section 2.6.2 for the IAM and IBM-UB1 databases. For each database, the atoms of the dictionary are learnt from the enrolled data corresponding to 25% of the number of users, that are randomly chosen.

3.7.1 Analysis of average entropy values H_B with bin size B

We begin by investigating on the size of the bin B that can be utilized for the histogram based feature sets, while ensuring discrimination between the writers. As part of the experimental set up, the number of clusters k_1 used at the first level is varied from 25 to 100 in steps of 25. As discussed in Section 3.4.1, each of these clusters are further split into W sub-clusters at the second level for the computation of the average entropy value H_B. For our case, the values of W used are 217 and 43, as these correspond with the number of writers in the IAM and IBM-UB1 database respectively.

The average entropy H_B is recorded for bin sizes B, ranging from 2 to 16 in steps of 2 in Table 3.2 (a) and (b) on the two databases. As for the trend, its value initially decreases with increasing values of B. Subsequent to attaining a minimum, it begin to increase again for higher number of bins. The lowest value of H_B is obtained with the bin size of 10 in both the databases.

As the number of bins increase, it is likely that, for large values, a perturbation to one or more of the orientations $\theta_{\mathbf{p}_i}, \theta_{\mathbf{\Delta_i}}, \theta_{\mathbf{a_i}}$ can lead to a change in the bin to which the votes are assigned. This results in relatively higher variation between the attributes of the histogram feature vectors corresponding to the sub-strokes of the same writer - thereby leading to the increase in the average entropy value. On the other hand, the increased H_B values at small number of bins can be explained due to the generated feature vectors not being able to adequately capture the nuances that discriminate the feature vectors belonging to different writers.

3.7.2 Performance with varying bins B and dictionary size M

In this subsection, we evaluate the performance of the proposed writer identification system with the variation of the number of bins B for the histogram based features and the number of the atoms M in the dictionary. We vary the value of B from two to 16 in steps of two and number of atoms from 50 to 500 in steps of 50. Tables 3.3 (a) and (b) report the results of our strategy for the IAM database. We observe that the proposed writer descriptor results

Table 3.2: Variation of the average entropy values H_B for varying number of bins B on the IAM and IBM-UB1 databases. The value of k_1 denote the number of clusters at the first level. For more details, please refer the text.

Number of bins B	Entropy H_B			
	k_1=25	k_1=50	k_1=75	k_1=100
2	6.44	5.86	5.70	5.54
4	6.35	5.61	5.43	5.29
6	6.11	5.47	5.20	5.04
8	5.76	5.20	4.82	4.77
10	**5.56**	**4.80**	**4.67**	**4.62**
12	5.63	4.89	4.86	4.73
14	6.04	5.15	5.08	4.91
16	6.21	5.40	5.25	5.08

(a) IAM Database

Number of bins B	Entropy H_B			
	k_1=25	k_1=50	k_1=75	k_1=100
2	4.93	4.84	4.79	4.72
4	4.76	4.71	4.62	4.57
6	4.59	4.53	4.46	4.41
8	4.42	4.37	4.31	4.26
10	**4.24**	**4.20**	**4.15**	**4.13**
12	4.45	4.36	4.28	4.24
14	4.57	4.52	4.43	4.35
16	4.68	4.65	4.54	4.48

(b) IBM-UB1 Database

in the best average identification rate of 99.45 % and 90.28% at paragraph and text line level respectively for a dictionary size $M = 400$ and bin size $B = 10$. With regards to the IBM-UB 1 database (shown in Table 3.4 (a) and (b)), we achieve a best average writer identification rate of 97.21% and 83.49% at paragraph and text line level respectively for the same value of (B, M).

It is interesting to note that the value of the bin size B leading to the best average writer identification rate coincides to that providing the lowest average entropy measure H_B in Tables 3.3 and 3.4 for the IAM and IBM-UB1 databases. This validates the need to select an appropriate choice of bin size for the extraction of the histogram feature sets, thereby leading to better discrimination of the sparse descriptors between the writers.

With regards to the trend in identification rates across all the databases, for a fixed dictionary size as we vary the number of bins, the average identification rate increases. After reaching a maximum value, it begins to decrease for larger values of B. Moreover, it may be observed that the performance on the IAM and IBM-UB1 database is inversely related to values of H_B reported in Tables 3.3 and 3.4 respectively. Said in another way, lower identification rates for a dictionary size M correspond to an increase in H_B and vice-versa.

Furthermore, it can also be noticed that for a particular value of bin size B, the average writer identification rate increases with dictionary size and then becomes comparable for values of $M \geq 400$. A small dictionary size leads to a low average identification rate due to the limited prototypes available for differentiating the sparse coding descriptors of the different writers.

Considering that the best performance on both the data-sets were obtained with writer descriptors constructed from histogram features parametrized with $B = 10$, we use the same bin size for the experiments described from subsections 3.7.4 to 3.7.7.

3.7.3 Influence of histogram feature sets on writer description

In the preceding experiments, we evaluated the efficacy of the writer descriptor constructed from the histogram based feature sets ($\mathbf{h_p}$, $\mathbf{h_\Delta}$ and $\mathbf{h_a}$) extracted from the sub-strokes (as discussed in Section 3.4). We now validate on the influence of the different choice of feature

Table 3.3: Comparison of the average writer identification rates (in %) at the paragraph and text line level for the proposed descriptor with varying number of bins B for the histogram feature sets and dictionary size M on the IAM database. The best average identification rate is marked in **bold**.

Number of	Dictionary size M							
bins B	50	100	150	200	250	300	400	500
2	60.94	66.46	69.24	73.49	75.94	77.19	80.81	80.55
4	65.79	68.76	71.22	75.38	77.99	79.64	84.45	84.69
6	72.03	76.34	79.29	83.15	86.77	88.73	91.31	91.04
8	76.41	81.61	84.54	87.35	89.54	92.07	94.77	94.54
10	79.10	84.19	87.99	89.76	93.65	97.80	**99.45**	98.78
12	78.15	82.84	86.19	87.07	91.80	94.88	98.73	98.38
14	76.99	79.68	83.50	86.92	88.84	92.19	97.65	97.40
16	75.50	78.28	81.77	84.61	86.61	89.73	97.03	96.87

(a) Paragraph level

Number of	Dictionary size M							
bins B	50	100	150	200	250	300	400	500
2	35.76	40.66	46.27	48.12	51.86	54.65	60.02	59.80
4	59.51	64.12	67.48	68.04	69.82	70.72	72.38	72.14
6	64.10	67.78	69.33	70.53	72.00	73.30	77.60	77.29
8	69.56	74.02	77.94	79.13	81.98	82.90	86.89	86.69
10	72.11	79.61	81.09	84.83	85.87	87.19	**90.28**	89.47
12	71.54	79.24	80.84	83.95	85.32	86.78	88.73	88.52
14	70.76	78.58	80.55	83.07	84.69	85.15	87.62	87.37
16	70.58	77.67	79.73	82.04	83.13	84.24	86.72	86.60

(b) Text-line level

Table 3.4: Comparison of the average writer identification rates (in %) at the paragraph and text line level for the proposed descriptor with varying number of bins B for the histogram feature sets and dictionary size M on the IBM-UB1 database. The best average identification rate is marked in **bold**.

Number of bins B	Dictionary size M							
	50	100	150	200	250	300	400	500
2	58.35	64.10	67.83	71.72	73.46	75.58	79.02	78.89
4	61.51	66.07	69.90	74.62	77.04	79.86	83.47	83.34
6	66.85	72.23	75.47	78.99	81.00	84.32	87.19	87.08
8	70.91	75.48	78.04	81.29	84.63	88.11	91.55	91.38
10	75.28	79.46	82.64	85.93	89.75	93.61	**97.21**	96.98
12	74.63	78.82	81.76	84.34	87.99	91.81	95.17	95.04
14	72.76	76.30	79.97	82.65	85.58	89.74	93.49	93.33
16	71.67	75.78	78.31	80.09	84.96	88.35	92.84	92.72

(a) Paragraph level

Number of bins B	Dictionary size M							
	50	100	150	200	250	300	400	500
2	28.70	37.10	45.15	49.30	50.09	51.24	53.77	53.42
4	44.64	51.68	57.47	62.17	64.57	66.42	67.99	67.80
6	53.19	59.05	65.44	69.30	72.74	75.95	77.78	77.65
8	57.49	64.68	68.99	71.72	74.14	77.71	78.74	78.67
10	59.68	66.69	70.84	74.76	78.69	82.12	**83.49**	83.39
12	58.06	65.88	69.63	73.88	77.57	81.73	82.45	82.31
14	57.34	64.67	68.97	72.16	76.32	79.56	80.43	80.35
16	56.81	63.95	67.89	71.52	75.43	78.29	79.83	79.76

(b) Text-line level

Table 3.5: Performance comparison of proposed writer identification system (in %) with combinations of histogram feature sets.

Histogram set used	IAM database				IBM-UB1 database			
	Paragraph Level		Text line Level		Paragraph Level		Text line Level	
	IR	(B, K)	IR	(B, K)	IR	(B, K)	IR	(B, K)
$\mathbf{h_p}$	95.76	(8,450)	67.45	(8,450)	87.18	(8,400)	74.94	(8,350)
$\mathbf{h_\Delta}$	96.53	(10,450)	71.12	(10,350)	89.82	(8,450)	76.24	(8,350)
$\mathbf{h_a}$	95.64	(8,400)	66.95	(8,450)	89.10	(10,350)	75.42	(10,400)
($\mathbf{h_p}$,$\mathbf{h_\Delta}$)	98.07	(10,350)	84.16	(10,400)	94.34	(10,400)	79.13	(10,450)
($\mathbf{h_p}$,$\mathbf{h_a}$)	98.44	(10,350)	85.45	(10,400)	94.64	(10,350)	79.76	(10,400)
($\mathbf{h_a}$,$\mathbf{h_\Delta}$)	98.73	(10,400)	87.42	(10,350)	95.42	(10,400)	82.48	(10,450)
($\mathbf{h_p}$,$\mathbf{h_\Delta}$,$\mathbf{h_a}$)	99.45	(10,400)	90.28	(10,400)	97.21	(10,400)	83.49	(10,400)

sets for writer description in Table 3.5. The best writer identification accuracies are reported with the number of dictionary atoms M and bin size B being specified. Hereinafter, for brevity sake, we abbreviate the average identification rate as IR.

The bin size for each combination of histogram sets reported in Table 3.5 correspond to the one minimizing the average entropy value H_B (defined in sub-section 3.4.1). From the entries, we see that the augmentation of feature sets $\mathbf{h_\Delta}$ and $\mathbf{h_a}$, that capture the vicinity information improve the performance of the identification system beyond that provided by $\mathbf{h_p}$.

3.7.4 Influence of the segmentation strategy

As mentioned in Section 3.3, there were two strategies described for the generation of sub-strokes. In this experiment, we consider the performance of our proposal on each of them separately by using the histogram feature set $\mathbf{h_p}$, $\mathbf{h_\Delta}$ and $\mathbf{h_a}$ obtained from the sub-strokes. For sake of simplicity, we denote the method of splitting the online trace at the local maxima in a stroke as 'SEG-1'. The second strategy 'SEG-2' corresponds to generating sub-strokes with a fixed number of points (chosen as 30 in this work).

Table 3.6 presents the best average writer identification results across all the databases at paragraph and text-line levels, with the bin and dictionary size being specified. From the

Table 3.6: Performance comparison of our proposal for the different segmentation strategies. The numbers being mentioned are the identification rates (in %)

Segmentation strategy	IAM Database				IBM-UB1 Database			
	Paragraph level		Text line level		Paragraph level		Text line level	
	IR	M	IR	M	IR	M	IR	M
SEG-1	99.17	350	82.15	450	94.56	350	76.60	400
SEG-2	99.05	300	83.66	400	93.43	450	78.02	400
Combined	99.45	400	90.28	400	**89.54**	400	83.49	400

entries of the third row, it may be inferred that the substrokes from both the strategies SEG-1 and SEG-2 leads to improvement of the writer identification system.

3.7.5 Comparison with max and average pooling based descriptors

In this section, we compare the performance of our proposed sparse coded descriptor with the traditional max and average pooling based writer descriptors (discussed in Equations 3.11 to 3.13). For this experiment, the histogram based feature sets were extracted with the bin size as 10 and the number of atoms in the dictionary was varied from 50 to 500 in steps of 50. The histogram sets ($\mathbf{h_p}$, $\mathbf{h_\Delta}$ and $\mathbf{h_a}$) corresponding to the segmented sub-strokes of the handwritten data are employed to generate all the three descriptors.

Table 3.7 provides the best average identification rate IR obtained with the corresponding dictionary size M. It can be noticed that our proposed sparse coding based descriptor provides improved performance when compared to conventional max and average pooling based writer descriptor. This enhancement could be attributed to our proposal capturing finer details of the writer when compared to traditional sparse representation based classification strategies.

3.7.6 Impact of the sparse framework for writer description

As mentioned in Section 3.5, sparse coding framework offers the flexibility in the representation of the segmented sub-strokes over hard clustering algorithms such as k-means. As a

Table 3.7: Comparison of our proposal with max and average pooling based descriptors. The best average identification rate together with the dictionary size are mentioned for both the databases at paragraph and text-line level.

Descriptor Type	IAM database				IBM UB1 database			
	Paragraph level		Text line level		Paragraph level		Text line level	
	IR	M	IR	M	IR	M	IR	M
Max pooling	98.03	400	85.89	400	94.28	400	78.57	400
Average Pooling	98.41	400	87.69	400	95.16	400	79.47	400
Proposed	**99.45**	400	**90.28**	400	**97.21**	400	**83.49**	400

demonstration to this, we investigate its influence on writer description over the k-means algorithm for varying dictionary sizes / number of prototypes. The average writer identification rates obtained at the paragraph level for the IAM and IBM-UB1 databases is shown in Table 3.8. The descriptors, as such are derived from the histogram feature set combination ($\mathbf{h_p}$,$\mathbf{h_\Delta}$ and $\mathbf{h_a}$) extracted from the segmented sub-strokes of the handwritten data.

With regards to the IAM database, the descriptors from the codebook generated with the k-means [2] attain a best performance of 96.81% for $M = 450$. Contrast to this, the sparse coding descriptors achieve an average identification rate of 99.45% for $M = 400$. This corresponds to a performance improvement of 2.73%. For the IBM-UB1 database, a best average writer identification rate of 97.21 % is achieved with the descriptors derived via sparse coding for a dictionary size $M = 400$, which is a performance improvement of 3.59% over those obtained from the k-means.

Moving further, an experimental comparison was also performed at the text line level across both the databases. The average writer identification rates for varying dictionary sizes are presented in Figure 3.5. From the best average writer identification results presented in the sub-figures, the descriptors from a sparse coding framework provide an improvement of 19.02% and 6.98% for the IAM and IBM-UB1 databases respectively.

[2] In our implementation, we set the value of k to the size of the codebook M.

Table 3.8: Comparison of the average writer identification rates (in %) at the paragraph level for the descriptor obtained via k-means and through sparse coding with varying dictionary sizes M. The best average identification rate is marked in **bold**.

Dictionary	IAM database		IBM-UB1 database	
size M	k-means	Sparse coding	k-means	Sparse coding
50	72.34	79.10	69.72	75.28
100	78.22	84.19	75.73	79.46
150	83.84	87.99	78.88	82.64
200	86.48	89.76	81.45	85.93
250	90.07	93.65	86.38	89.75
300	94.29	97.80	89.76	93.61
350	95.95	98.88	92.29	95.13
400	96.68	**99.45**	**93.84**	**97.21**
450	**96.81**	98.93	93.75	97.10
500	96.63	98.78	93.66	96.98

3.7.7 Performance with a variant of writer descriptor

Recall that in the derivation of the writer descriptor in Section 3.6, we separately score each attribute of the segmented sub-stroke by taking into consideration on whether the reconstruction error is positive or negative. For sake of completion, we also explored the possibility of a variant of the descriptor, wherein for a dictionary atom, attributes are scored irrespective to the sign of the reconstruction error. Mathematically speaking, the score of the d^{th} attribute $(1 \leq d \leq D)$ of the j^{th} feature vector $\mathbf{f}_{ij}$ having a non-zero sparse coding coefficient α_{ij} for the i^{th} dictionary atom $\boldsymbol{\phi}_i$ is given as:

$$S_{ij}(d) = \frac{1}{1 + |f_{ij}(d) - \alpha_{ij} \times \phi_i(d)|} \tag{3.23}$$

$$1 \leq i \leq M \ , \ 1 \leq j \leq n_i$$

Subsequent to obtaining the scores, the descriptor is computed by following steps similar to those described from Equation 3.15 to Equation 3.18. However, in place of obtaining two separate vectors $\mathbf{S}_i^+$ and $\mathbf{S}_i^-$ for the i^{th} dictionary atom $\boldsymbol{\phi}_i$, we get a single vector of dimension D in this variant. The final descriptor, as such, comprises the L_2 norm of these vectors,

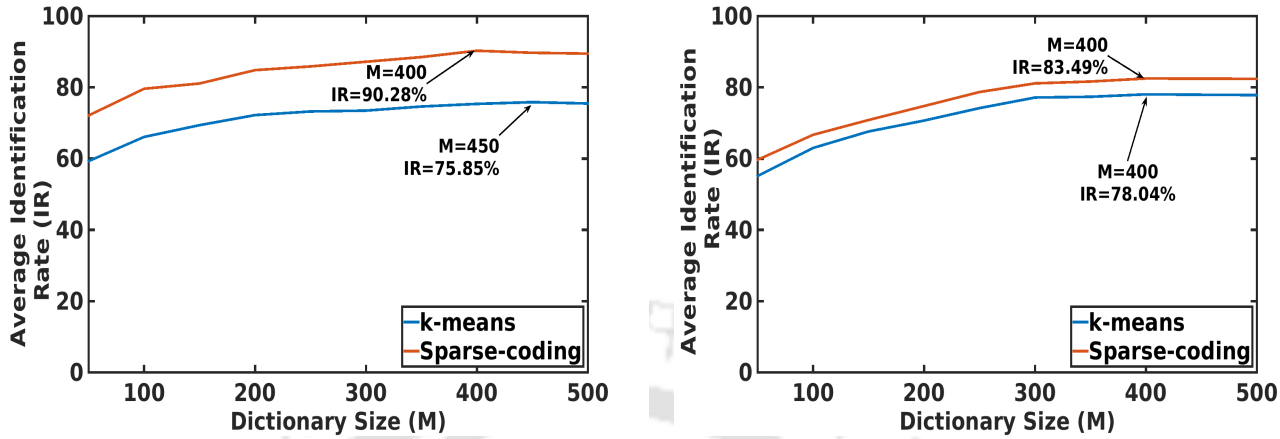


Figure 3.5: Variation of average writer identification rates (in %) at the text-line level for the proposed descriptor obtained via k-means (shown in blue) and through sparse coding (in red) on the IAM (sub-figure (a)) and IBM-UB1 databases (sub-figure (b)).

computed across all the M dictionary atoms - thus leading to a writer descriptor of dimension M. The histogram feature set combination ($\mathbf{h_p}$,$\mathbf{h_\Delta}$ and $\mathbf{h_a}$) extracted from the segmented substrokes are used for the experiment.

The evaluation of the above descriptor at the paragraph and text line levels for the IAM and IBM-UB1 databases is presented in Table 3.9. For sake of brevity, we refer it as the 'M-dim', while the one discussed in Section 3.6 is abbreviated to as '$2 \times M$-dim'. The best average writer identification rate along with the dictionary size M is mentioned in the Table. From the entries, we can infer that across both the data-sets, the '$2 \times M$-dim' descriptor provides improved results at paragraph and text-line levels, when compared to its variant - 'M -dim'. This, we believe, may be due to the increased discrimination achieved with the scoring of the attributes of the segmented sub-stroke feature vectors based on both the value and sign of the reconstruction error. Needless to say, the higher performance is achieved with the trade-off of increased dimension by a factor of nearly two (450 versus 800).

3.7.8 Time complexity comparison with our proposal in Chapter 2

In this sub-section, we present a time complexity analysis of the present proposal to the one described in Chapter 2. The experimental setting used for this purpose is the paragraph

Table 3.9: Performance comparison with a variant of the writer descriptor on both the databases. The numbers being mentioned are the identification rates (in %)

Writer descriptor	IAM Database				IBM-UB1 Database			
	Paragraph level		Text line level		Paragraph level		Text line level	
	IR	M	IR	M	IR	M	IR	M
M - dim	98.84	450	87.12	450	95.89	450	80.36	450
$2 \times M$-dim	99.45	400	90.28	400	97.21	400	83.49	400

Table 3.10: Comparison of the average time complexity (in seconds) of the proposals in Chapters 2 and 3. The experimental setting used for this purpose is the paragraph level training for the IAM database.

Description	Chapter 2	Chapter 3
Pre-processing(per document)	2.29	4.28
Feature extraction(per document)	0.22	0.07
Codebook/Dictionary generation	2439.60	15.85
Writer descriptor computation (per document)	1.09	0.29
SVM training (per writer)	0.02	0.02
SVM testing (per document and writer)	8×10^{-3}	6×10^{-3}

level training for the IAM database. The values of codebook/dictionary size which provided the best identification rate were considered for this study. The analysis was done on a Dell Precision Tower 3620 workstation with 32 GB RAM and an Intel Xeon processor.

From Table 3.10, we make the following observations:

(a) The pre-processing module for the sparse-coded descriptor approach is relatively greater than that for the codebook descriptor system. This can be attributed to the addition of the substroke generation module for the proposal in Chapter 3.

(b) The increased computation time however gets compensated with the feature extraction and the dictionary generation module respectively.

(c) The writer descriptor computation for Chapter 2 takes 0.8 seconds more per document when compared with the present approach.

(d) The time for training and testing the SVMs are comparable for both the descriptors.

Thus, on the whole, we observe that the proposal of the present Chapter takes lesser time for computation when compared to the one in Chapter 2.

3.8 Conclusion

We conclude this Chapter with an enumeration of the contributions:

(i) Proposal of descriptors derived from a set of dictionary atoms. The descriptors for each dictionary atom encode the error while using it alone for reconstruction.

(ii) Utilization of histogram based feature vectors extracted at a sub-stroke level, together with their corresponding sparse coefficients, for deriving the sparse coded descriptors.

(iii) Proposal of an entropy based analysis for the appropriate bin size to be selected for obtaining the features so as to ensure the discrimination between the writer descriptors.

4

Exploration of saliency information in the sparse framework

Contents

4.1 Introduction

In this Chapter, we attempt to explore additional information from the sparse coding coefficients that can be useful for writer identification. Recall from Section 3.5 that traditional sparse representation based classification strategies consider the accumulation of sparse coefficients over a set of dictionary atoms followed by a averaging process. The averaged value from each dictionary atom as such is used as a feature vector for classification. This strategy is termed as mean / average pooling. Likewise, another technique is that of max pooling where in the maximum of a set of sparse coefficients is computed for each dictionary atom in lieu of the average value. In this work, we propose to reformulate the above strategies by taking into regard prior knowledge that is obtained from the set of dictionary atoms.

The additional information, as we shall see, is related to quantifying in an average sense, the degree of importance of each of the dictionary atoms with regards to the dynamic characteristics of the enrolled writers. In this context, we define in this work, the term "saliency" [1] for an dictionary atom - the value of which is learnt from the sparse coefficients corresponding to the sub-strokes of the handwritten training data. In order to calculate its value, we propose two separate strategies as outlined below.

- In the first method, the saliency values for the over-complete set of dictionary atoms are obtained from computing entropy measures over histograms generated from the sparse coefficients. The number of histograms considered for this strategy correspond to the size of the dictionary.

- Our second technique for determining the saliency utilizes a single histogram, that is generated from sum pooling the sparse coefficients over all the feature descriptors of sub-strokes segmented from the training data. The calculation of its value, as we shall see, relies on an equation, that bears semblance to that of the inverse document frequency (idf) used in the area of document image retrieval.

[1] In this thesis on online writer identification, this term is not to be interpreted to that used by researchers of the image processing and computer vision community.

During the identification of a test document, the saliency values (computed by one of the above two proposals) serve as a-priori information, that can be incorporated in the traditional framework of max or average pooling. This, in turn leads to a modified descriptor, that can be used for establishing the author of the document under question. To the best of our knowledge, the proposed saliency computation outlined in this Chapter is a first of its kind to be used in the sparse coding domain. Experiments performed with the proposed sparse classification strategy shows improved writer identification rates over the traditional schemes.

Further to the above, we also consider incorporating writer-specific adaptation of saliency values, that quantifies how important a dictionary atom is for a given writer. Our approach, as we shall see in this Chapter, employs the reconstruction error on the sub-stroke based feature vectors to derive a similarity score for each dictionary atom with regards to a writer using only his / her handwriting. The obtained scores across all the dictionary atoms are subsequently fused with their respective saliency values [2] to generate the adapted values for the purpose of identification. In particular, we formulate an ensemble of SVMs, wherein the descriptor to the SVM trained for a writer is based on the saliency values adapted for that writer. The final decision on the authorship is proposed as the maximum of the prediction score obtained from the SVMs.

In the subsequent Sections, we present details of each of the contributions made in this Chapter. To begin with, we provide a block schematic of our proposed framework, that employs the saliency information of the dictionary atoms in Section 4.2. In order to compute these values, we consider two new strategies - the details of which are described in Sections 4.3 and 4.4 respectively. Subsequent to obtaining the saliency of dictionary atoms, we discuss their incorporation into the traditional sparse framework in Section 4.5, that leads in a modified writer descriptor. As an extended idea in Section 4.6, we present our approach of determining writer specific adapted saliency values and their utilization in writer identification. Last but not the least, several experiments have been outlined in Section 4.7 to establish the various

[2]These values correspond to those obtained without adaptation and quantify the importance of the atoms with regards to the average dynamic characteristics of the enrolled writers.

aspects of our contributions to writer identification.

4.2 Schematic of the proposed framework

Figure 4.1 provides a pictorial description of our approach. To begin with, the dynamic information of the handwritten trace of a test document is passed through a preprocessing module. Subsequent to this, we segment the data into substrokes and extract histogram based features from each of them. The resulting feature vectors are then sparsely coded using an over-complete dictionary. The learning of the dictionary atoms is accomplished using documents enrolled for training in the writer identification system. The differing aspect of our work, is however, in the association of saliency values for each of the dictionary atoms. As mentioned earlier, these values are derived from the sparse codes of the training data via two approaches outlined in the previous Section.

Likewise, for constructing the writer description for a test data, we rely on the sparse coefficients corresponding to the dictionary atoms together with the knowledge from the saliency values determined a-priori during training. The resulting descriptor is then sent as an input to an Support Vector Machine for establishing the authorship of the document.

In the following two sections, we provide the details of the methods for determining the degree of saliency of the dictionary atoms with regards to the dynamic characteristics of a writer. We begin with an elaboration of the first method, that is proposed based on entropy in Section 4.3. This is followed by the description of our second proposed strategy that makes use of the sparse coefficients (obtained from sum pooling) to compute the saliency values in Section 4.4.

Before moving ahead, we wish to reiterate on the following in the context of the present work.

- The preprocessing steps being used are those of isolated point removal, same time stamp removal and stroke merge elaborated in Section 2.2.

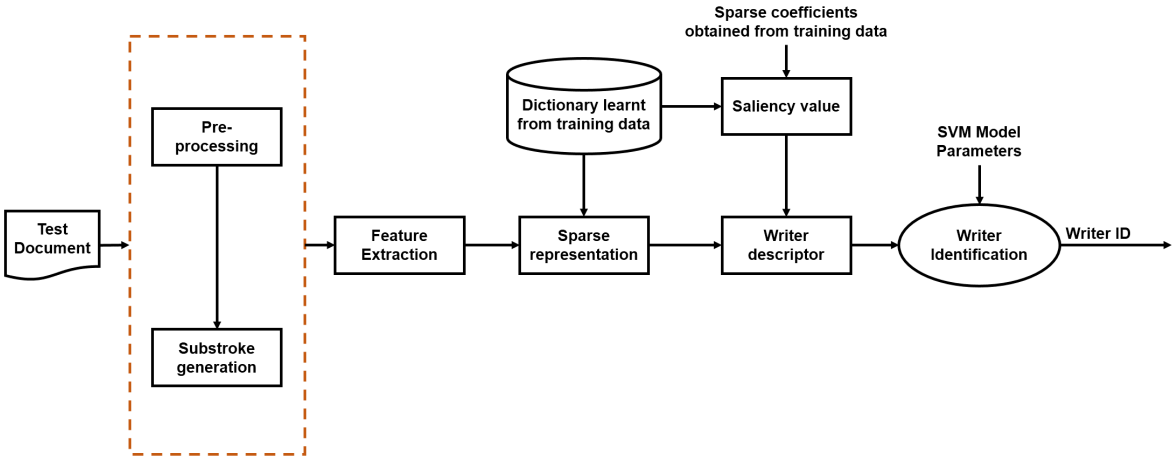


Figure 4.1: Block diagram of the proposed sparse framework based online writer identification system, that employs the saliency values of the dictionary atoms for writer description. The test data refer to paragraphs or text lines written by the enrolled writers.

- The generation of the sub-strokes is based on the strategies discussed in Section 3.3.
- The feature vectors used for dictionary learning and subsequent sparse coding correspond to the histogram set described in Section 3.4 with the bin size B set to ten.
- The atoms of the dictionary $\mathbf{D}$ are obtained by employing the algorithm of [76] implemented in the SPArse Modeling Software (SPAMS) optimization toolbox.

4.3 Entropy based saliency computation

We assume that a learnt dictionary $\mathbf{D}$ comprising an over-complete set of M atoms is available for us. Let N_{Tr} denote the total number of sub-strokes from all the writers whose data is enrolled to the system for training. Assuming, that we have W writers, we partition the N_{Tr} sub-strokes into W subsets $\{\mathbb{S}_k\}_{k=1}^{W}$ in a way that each subset comprises the feature vectors of all sub-strokes segmented from the training handwritten samples of a single writer. Accordingly, the cardinality of $\mathbb{S}_k$, denoted by $|\mathbb{S}_k|$ corresponds to the number of sub-strokes for the k^{th} writer.

The pipeline of steps that we follow in our approach can be enumerated as follows:

(i) Generation of histograms for each of the dictionary atoms.

(ii) Computation of the entropy for the set of histograms generated in step (i).

(iii) Computation of saliency value as a function of the entropy.

The details corresponding to each of them is provided in the following subsections.

4.3.1 Histogram generation

As stated, we construct separate histograms for the M atoms $\{\boldsymbol{\phi}_i\}_{i=1}^{M}$ of the over-complete dictionary $\mathbf{D}$ by proposing a generic vote accumulation framework - the details of which are outlined below for a specific dictionary atom.

Let $\mathcal{H}_\mathbf{i}$ denote the histogram for the i^{th} dictionary atom $\boldsymbol{\phi}_i$, $1 \leq i \leq M$. The process of generation is based on utilizing the sparse coefficients corresponding to the dictionary atom $\boldsymbol{\phi}_i$ across all the writers. The size of the bins are set to W - corresponding to the number of writers enrolled into the system. Accordingly, the values of the votes in each of the bins of $\mathcal{H}_\mathbf{i}$ can now be defined by $\{h_{i1}, h_{i2}, \cdots, h_{iW}\}$.

To begin with, the votes are set to zero across all the W bins. The process of accumulation of the votes in the k^{th} bin, $(1 \leq k \leq W)$ is proposed as follows:

For every feature vector of the sub-stroke from the k^{th} writer (in subset $\mathbb{S}_k$), we consider the sparse coefficient associated to the i^{th} dictionary atom. The present vote value for the k^{th} bin of $\mathcal{H}_\mathbf{i}$ is then accumulated by the magnitude of this sparse coefficient.

The resulting accumulated votes from the sub-strokes of each of the W writers leads to the generation of the histogram $\mathcal{H}_\mathbf{i}$. Said in another way, each bin represents the agglomeration of the magnitude of the sparse coefficients corresponding to a given writer. Nonetheless, without loss of generality, it may be noted that not all sparse coefficients being accumulated are necessarily non-zero. Keeping this in perspective, we obtain an average value for the votes / sparse coefficients across each of the bins of $\mathcal{H}_\mathbf{i}$ as follows:

Let $\{\alpha_{ij}^k\}$ denote the sparse coefficient corresponding to the i^{th} dictionary atom, as obtained from the j^{th} sub-stroke of the k^{th} writer (in $\mathbb{S}_k$). We introduce an indicator variable $\mathcal{I}(\alpha_{ij}^k)$ as follows:

$$\mathcal{I}(\alpha_{ij}^k) = \begin{cases} 1 & |\alpha_{ij}^k| > 0 \\ 0 & Otherwise \end{cases} \tag{4.1}$$

$$1 \leq i \leq M,\ 1 \leq j \leq |\mathbb{S}_k|,\ 1 \leq k \leq W$$

From the above, it may be inferred that $\sum_{j=1}^{|\mathbb{S}_k|} \mathcal{I}(\alpha_{ij}^k)$ corresponds to the number of non-zero sparse coefficients for the i^{th} dictionary atom from the k^{th} writer. Taking this into consideration, the average value of the votes / sparse coefficients for the k^{th} bin can now be written as:

$$\tilde{h}_{ik} = \frac{h_{ik}}{\sum_{j=1}^{|\mathbb{S}_k|} \mathcal{I}(\alpha_{ij}^k)} \tag{4.2}$$

As a further step, by applying the following transformation, we ensure that the above votes of the histogram are normalized to values in the range between 0 and 1.

$$p_{ik} = \frac{\tilde{h}_{ik}}{\sum_{k=1}^{W} \tilde{h}_{ik}} \tag{4.3}$$

4.3.2 Computation of saliency with entropy based values

Subsequent to the generation of the histograms $\{\mathcal{H}_\mathbf{i}\}_{i=1}^M$ for the M atoms $\{\boldsymbol{\phi}_i\}_{i=1}^M$, we compute the entropy value for each of them as:

$$H_i = \sum_{k=1}^{W} -p_{ik} \log_2(p_{ik}) \quad 1 \leq i \leq M \tag{4.4}$$

The set of obtained entropy values are then used to determine the saliency value for each atom $\boldsymbol{\phi}_i$ through an inverse relation defined below:

$$w_i = \frac{1}{1 + H_i} \tag{4.5}$$

It may be noted that the value of w_i is between 0 and 1.

As a motivation to the use of the entropy measure, consider the case, wherein all the sparse

coefficients for a dictionary atom (say $\boldsymbol{\phi}_i$) encapsulate the information of only the n^{th} writer. Clearly, with regards to the histogram $\mathcal{H}_\mathbf{i}$, we can now write:

$$p_{ik} = \begin{cases} 1 & k = n \\ 0 & Otherwise \end{cases} \tag{4.6}$$

For this scenario, the obtained value of the entropy H_i is zero. It is thus natural to suggest that with regards to the sparse representation framework, the importance or saliency value for $\boldsymbol{\phi}_i$ should be maximum. This indeed is satisfied from utilizing the Equation 4.5, wherein we obtain a value of one for w_i.

Further to the above, one can also infer that in practice, dictionary atoms with histograms presenting a lower entropy value indicate a higher degree of importance or saliency and vice-versa. In short, the proposed importance value determination of all the dictionary atoms can be regarded as a-priori information, that can be incorporated into the sparse classification based writer description for performance improvement. The details of the same is discussed in Section 4.5.

As a further demonstration to the preceding discussion on saliency computation, we present histograms corresponding to four dictionary atoms with their corresponding normalized votes in Figure 4.2. For sake of visual clarity, we construct them by accumulating the votes of the sparse coefficients from the sub-strokes corresponding to four paragraphs of 50 writers of the IAM-Online database [3]. Accordingly, the bins of the histograms are set to 50 - the number of writers.

The atoms being utilized for generating the two histograms in sub-figures 4.2 (a) and (b) are those that provide the lowest and highest saliency values respectively [4]. For the present discussion, we denote them as $\boldsymbol{\phi}_{min}$ and $\boldsymbol{\phi}_{max}$. It can be inferred that the trend in the normalized votes are quite different for these histograms. The histogram in sub-figure 4.2 (a) corresponding to the atom $\boldsymbol{\phi}_{min}$ (with the minimum saliency value) has bins with normalized votes more or

[3] We follow the protocol mentioned in Section 2.6.2, where four paragraphs of a writer are reserved used for training.

[4] The over-complete dictionary $\mathbf{D}$, for this illustration, comprises 400 atoms.

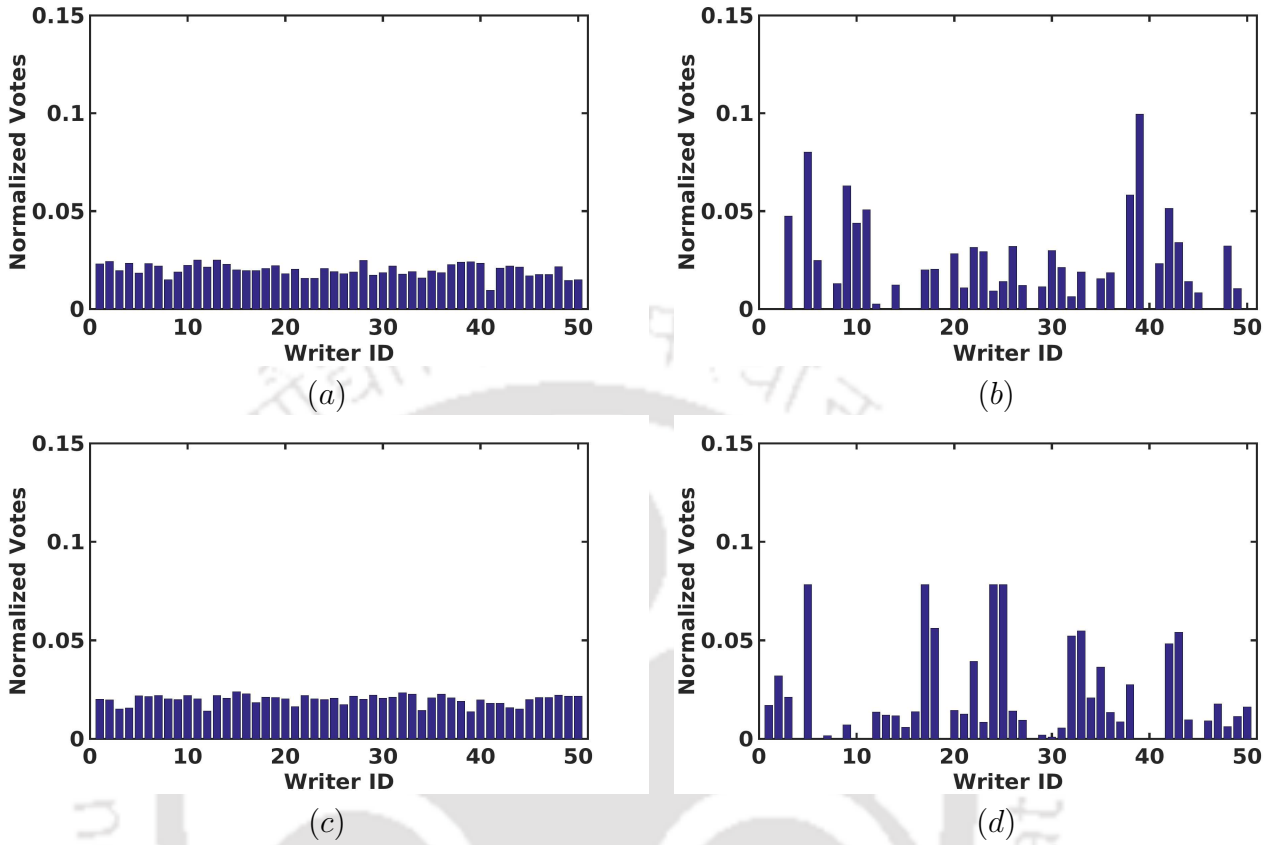


Figure 4.2: Sub-figures (a) and (b) illustrate the histograms corresponding to two dictionary atoms $\boldsymbol{\phi}_{min}$ and $\boldsymbol{\phi}_{max}$ presenting the minimum and maximum saliency value. Likewise, the sub-figures (c) and (d) are the histograms of the atoms presenting the second minimum and maximum saliency value. The normalized votes for each of the four histograms are obtained by accumulating the sparse coefficients from the sub-strokes corresponding to four paragraphs of 50 writers of the IAM-Online database. For more details refer the text.

less same across all the 50 writers - implying a near uniform distribution. This in turn results in higher entropy and subsequently a low saliency / importance value for the atom (via Equation 4.5).

Compared to Figure 4.2(a), we see that the votes for the histogram of the atom $\boldsymbol{\phi}_{max}$ in sub-figure 4.2 (b) exhibit more variation with relatively higher values being assigned to a subset of the writers. In fact, a few of the bins corresponding to the writers have votes to as low as zero. Thus, from the perspective of entropy, this histogram presents a lower value, thereby indicating that the atom $\boldsymbol{\phi}_{max}$ is more salient with regards to the sparse classification framework.

Furthermore, we present the histograms of the dictionary atoms providing the second mini-

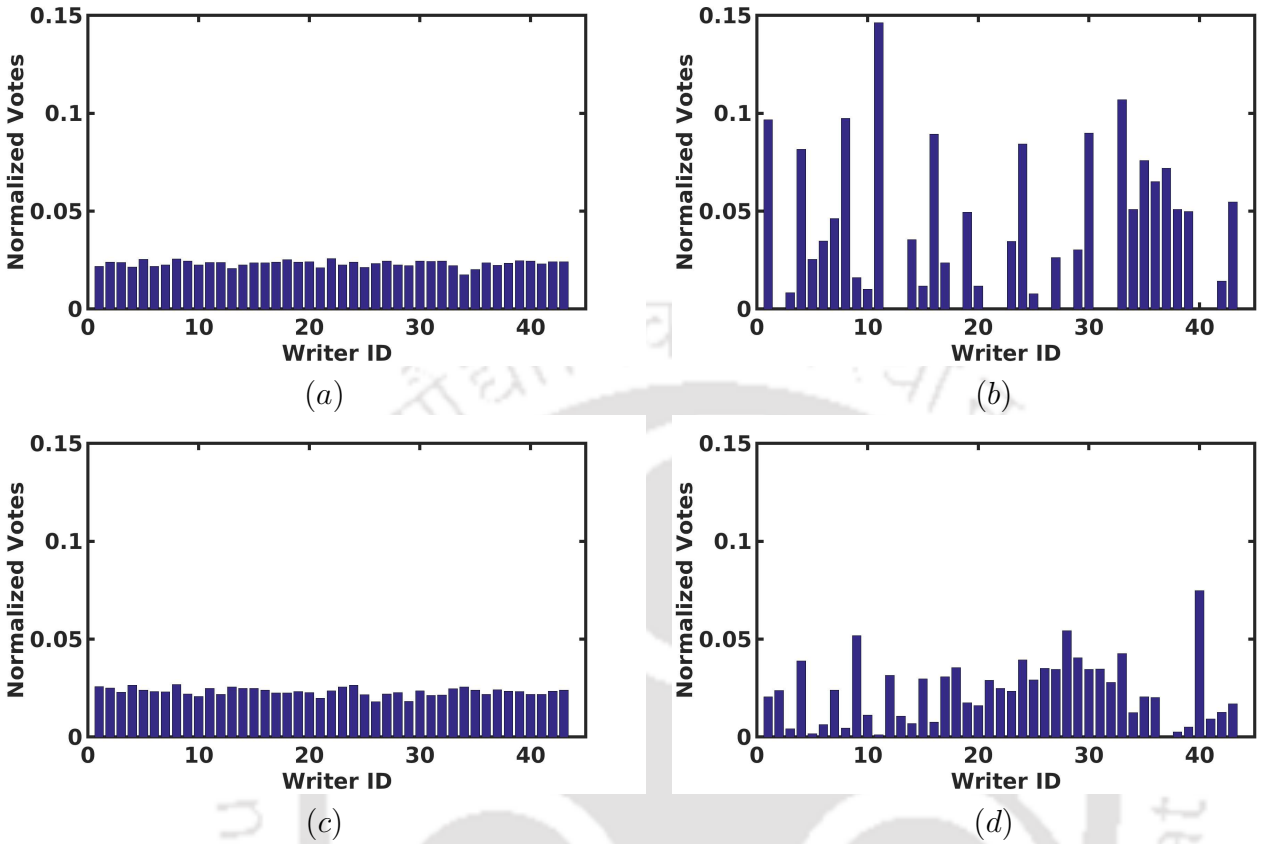


Figure 4.3: The different sub-figures illustrate the histograms corresponding to the dictionary atoms presenting the first and second minimum and maximum saliency value for the IBM-UB1 database. The normalized votes for each of the four histograms are obtained by accumulating the sparse coefficients from the sub-strokes corresponding to the data of the 43 writers of this database.

mum and maximum saliency values in the sub-figures 4.2(c) and (d). We observe that the trend of the normalized votes being presented in these histograms are quite in line to those of sub-figures 4.2(a) and (b) respectively. For sake of completeness, we also show the histograms corresponding to the dictionary atoms with the first and second minimum and maximum saliency value for the IBM-UB1 database in Figure 4.3.

Lastly, for better clarity, we summarize the above idea in Algorithm 3 with an enumeration of the pipeline of steps proposed for saliency value computation of the dictionary atoms. It may be noted that the computation complexity for each of the steps, namely histogram generation, entropy calculation and saliency value computation are $\mathcal{O}(M \times N_{Tr})$, $\mathcal{O}(M \times W)$ and $\mathcal{O}(M)$ respectively.

Algorithm 3 Proposed entropy based saliency value computation

- **Input:**
  - A set of histogram based feature vectors $\{\mathbf{f}_j\}_{j=1}^{N_{Tr}}$ extracted from N_{Tr} sub-strokes corresponding to enrolled training documents of W writers. (Refer Section 3.4).
  - Dictionary $\mathbf{D}$ with M atoms $\{\boldsymbol{\phi}_i\}_{i=1}^{M}$.
- **Output:** Saliency values $\{w_i\}_{i=1}^{M}$
- **Algorithm:**

  Compute: $\boldsymbol{\alpha}_j$ - the sparse coding coefficient vector for each feature vector $\mathbf{f}_j$ as described in Section 3.5.

  For every dictionary atom $\boldsymbol{\phi}_i$ in $\mathbf{D}$
  - Utilize the sparse coefficients to compute the histogram $\mathcal{H}_\mathbf{i}$ comprising W bins.
  - Compute the entropy of $\mathcal{H}_\mathbf{i}$.
  - Compute the saliency value w_i using Equation 4.5.

  End For

4.4 Sum-pooling based saliency computation

For the discussion of this strategy, we assume that a dictionary $\mathbf{D}$ comprising M atoms has been pre-learnt using feature vectors corresponding to N_{Tr} sub-strokes. We partition the sub-strokes to W subsets $\{\mathbb{S}_k\}_{k=1}^{W}$ in a way that $\mathbb{S}_k$ comprises the feature vectors of all sub-strokes segmented from the training handwritten samples of the k^{th} writer.

Compared to the entropy based approach detailed in Section 4.3, the present strategy for saliency computation, as we shall see shortly, relies on the information obtained from a `single` histogram. The process of generation of this histogram is based on utilizing the sparse coefficients corresponding to the dictionary atom $\boldsymbol{\phi}_i$ across all the writers.

Let $\mathcal{H}$ denote the histogram with the number of bins set at M- the dictionary size. Accordingly, we can specify $h_1, h_2, ...h_M$ to represent the values of the votes in the bins. To begin with, these votes are initialized to zero. The process of accumulation of the i^{th} bin, $(1 \leq i \leq M)$

is proposed as follows:

For every feature vector of the sub-stroke from the k^{th} writer (in subset $\mathbb{S}_k$), we consider the normalized sparse coefficient associated to the i^{th} dictionary atom. The present vote value for the i^{th} bin of $\mathcal{H}$ is then accumulated by the magnitude of this sparse coefficient.

The resulting accumulated votes from the sub-strokes of each of the W writers leads to the generation of the histogram $\mathcal{H}$. Said in another way, each bin represents the agglomeration of the magnitude of the normalized sparse coefficients from all writers.

Let α_{ij}^k denote the sparse coefficient associated with the i^{th} dictionary atom, as obtained from the j^{th} sub-stroke of the k^{th} writer. Subsequent to the generation of the histogram $\mathcal{H}$, the vote for the i^{th} bin h_i is given by

$$h_i = \sum_{k=1}^{W} \sum_{j=1}^{|\mathbb{S}_k|} \bar{\alpha}_{ij}^k \tag{4.7}$$

where each of the summation terms

$$\bar{\alpha}_{ij}^k = \frac{\alpha_{ij}^k}{\sum_{i=1}^{M} \alpha_{ij}^k} \tag{4.8}$$

corresponds to the normalized value of the sparse coefficient α_{ij}^k that lies between $[0, 1]$.

We now define the saliency for the dictionary atom $\boldsymbol{\phi}_i$ as:

$$w_i = \log_e \frac{N_{Tr}}{h_i} \tag{4.9}$$

Without loss of generality, the number of sub-strokes N_{Tr} pooled across all the W writers are greater than the votes received in each bin $\{h_i\}_{i=1}^{M}$. Owing to this, the M values of the saliency computed from Equation 4.9 are greater than 1. Typically, for a dictionary size $M = 400$, they lie in the range $[5, 7]$ for most of the atoms on the IAM and IBM-UB1 databases. The plots demonstrating the trend of saliency values, as obtained in these databases, are shown in Figure 4.4 (a) and (b) respectively.

In order to reduce the dynamic range of the values of the saliency values, we employ the

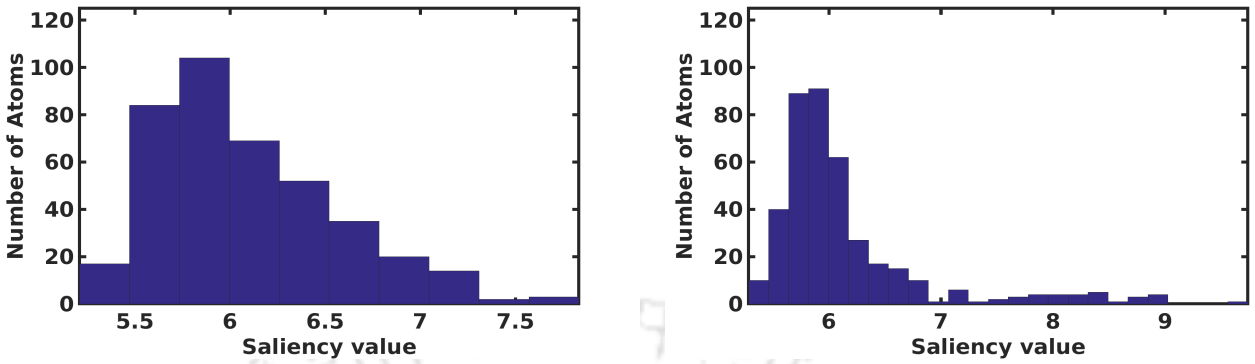


Figure 4.4: The two sub-figures represent the plots depicting the trend in the saliency values, obtained with respect to the dictionary atoms for the IAM and IBM-UB1 database respectively. For this illustration, the size of the dictionary $M = 400$.

following factor NF to map them in the range $[0, 1]$.

$$NF = \max_{1 \leq i \leq M} \log_e \frac{N_{Tr}}{\sum_{k=1}^{W} \sum_{j=1}^{|\mathbb{S}_k|} \alpha_{ij}^k} \tag{4.10}$$

It may be worth citing that the relationship proposed above bears semblance to the equation of inverse document frequency (idf) from the field of document retrieval. In idf based scoring, we emphasize on providing degree of importance to a given word based on its occurrence in a set of documents [35]. In the context of our work, we analogize upon this idea by encoding the degree of importance (or saliency) of a dictionary atom based on the sparse coefficients, that are derived from the feature vectors of the sub-strokes of the enrolled writers.

For brevity, we summarize the proposed sum pooling based saliency computation in Algorithm 4. In the next Section, we discuss on how the saliency values are incorporated into the traditional sparse framework, leading to a modified writer descriptor.

4.5 Modified Writer Descriptor

Recall from Section 3.5 that the conventional strategy is to generate a writer descriptor that is based on max or average pooling of the sparse codes.

Algorithm 4 Proposed sum pooling based saliency value computation

- **Input:**
  - A set of histogram based feature vectors $\{\mathbf{f}_j\}_{j=1}^{N_{Tr}}$ extracted from N_{Tr} sub-strokes corresponding to enrolled training documents of W writers.
  - Dictionary $\mathbf{D}$ with M atoms $\{\boldsymbol{\phi}_i\}_{i=1}^{M}$.
- **Output:** Saliency values $\{w_i\}_{i=1}^{M}$
- **Algorithm:**

  Compute: $\boldsymbol{\alpha}_j$ - the sparse coding coefficient vector for each feature vector $\mathbf{f}_j$.

  Initialize a histogram $\mathcal{H}$ of M bins with zero votes.

  For every feature vector in $\{\mathbf{f}_j\}_{j=1}^{N_{Tr}}$
  Utilize the normalized sparse coefficients to accumulate the votes in the M bins of $\mathcal{H}$.
  End For

  Compute the saliency of the M dictionary atoms using Equation 4.9.

  Normalize the values of the saliency in the range $[0, 1]$ with the factor in Equation 4.10.

- Max pooling

$$z_i = \max_{1 \le j \le N_T} |\alpha_{ij}| \tag{4.11}$$

- Average pooling

$$z_i = \frac{1}{\sum_{j=1}^{N_T} \mathcal{I}(\alpha_{ij})} \sum_{j=1}^{N_T} |\alpha_{ij}| \tag{4.12}$$

$$1 \le i \le M$$

Here, N_T corresponds to the number of sub-strokes in the test document.

In our proposal, we attempt at modifying each of the above elements (obtained by either of the strategies) by incorporating the prior knowledge available in the form of saliency values for the dictionary atoms. In particular, we obtain a modified M dimensional descriptor $\hat{\mathbf{z}}$ using

Algorithm 5 Writer descriptor generation using saliency values

- **Input:**
  - A set of histogram based feature vectors $\{\mathbf{f}_j\}_{j=1}^{N_T}$ extracted from a document with N_T sub-strokes.
  - Dictionary $\mathbf{D}$ with M atoms $\{\boldsymbol{\phi}_i\}_{i=1}^{M}$.
  - Saliency values $\{w_i\}_{i=1}^{M}$.
- **Output:** Modified writer descriptor
- **Algorithm:**

  Calculate $\boldsymbol{\alpha}_j$ - the sparse coding coefficient vector for each feature vector $\mathbf{f}_j$.

  For every dictionary atom $\boldsymbol{\phi}_i$ in $\mathbf{D}$
  - Compute z_i by using the average pooling or max pooling strategy.
  - Modify z_i with the saliency value of $\boldsymbol{\phi}_i$ to give $\hat{z}_i$.

  End For

  Stack $\{\hat{z}_i\}_{i=1}^{M}$ to obtain the modified writer descriptor $\hat{\mathbf{z}}$.

the product rule, with the i^{th} element being expressed as

$$\hat{z}_i = w_i \times z_i \tag{4.13}$$

Our formulation of the modified descriptor ensures that higher preference is given to sparse coefficients corresponding to dictionary atoms of higher saliency / importance and vice-versa. With regards to the computational complexity, the modified descriptor involves an additional time complexity of $\mathcal{O}(M)$, as compared to the traditional average or max pooling method, *viz.* $\mathcal{O}(N_T \times M)$. Following the generation of the modified descriptor, the one-versus-all Support Vector Machine (SVM) with the RBF kernel is used to establish the identity of the writer for the test document.

Last but not the least, we provide a high level summary of the proposed writer descriptor in the form of pseudo-code (refer Algorithm 5).

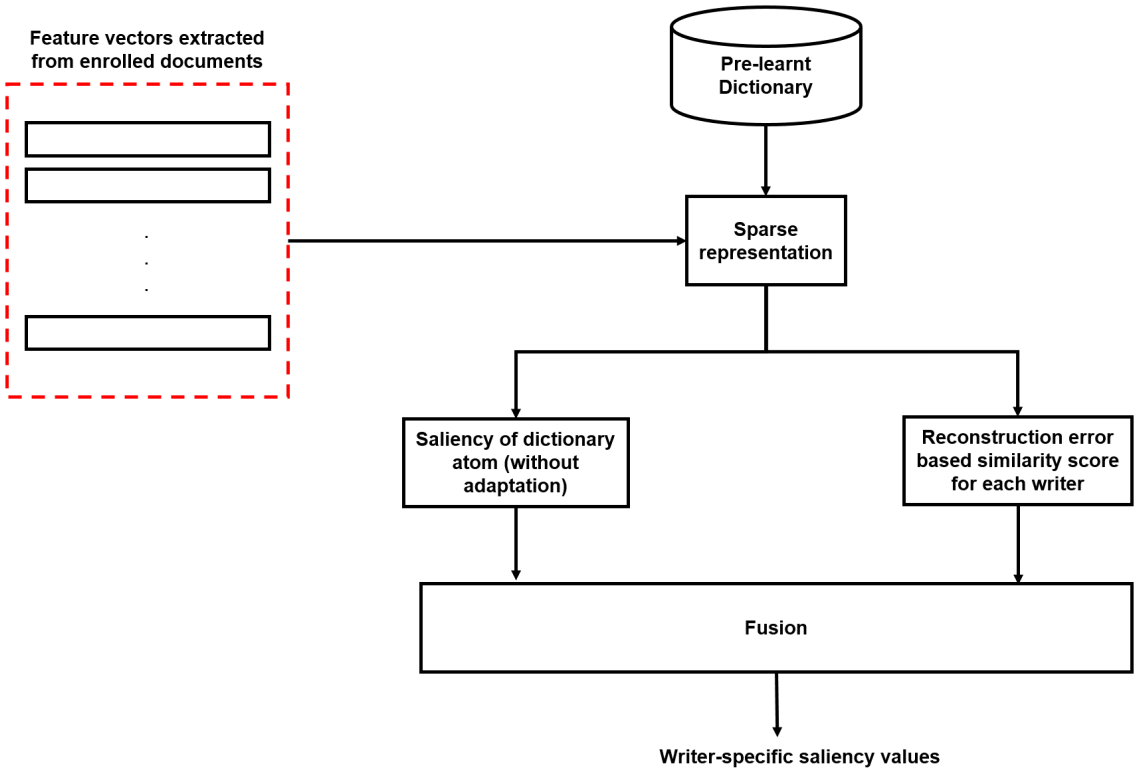


Figure 4.5: Pictorial overview of the approach depicting the adaptation of saliency values to a given writer. For more details, refer the text.

4.6 Writer specific saliency value adaptation

In this section, we extend the idea of using saliency values in a way, so that they signify the importance of each of the dictionary atoms for a **given** writer. In particular, we focus on adapting them to the writer in question by considering the dynamic characteristics of his / her handwriting only.

Figure 4.5 presents the high level pictorial view of our approach, that comprises two steps. In the first step, we compute the saliency values by utilizing one of the two methods discussed earlier in Sections 4.3 and 4.4 of this Chapter. At this point, it may be worth recalling that each such computed value represents the importance of a dictionary atom with regards to the average dynamic characteristics of the writers. Without loss of generality, the saliency values

at this step are obtained without adapting them to any specific writer.

The added distinction / novel aspect comes via the second step, wherein we attempt to formulate a similarity score for each atom, based on the reconstruction errors obtained from the training feature vectors of the writer under consideration. Accordingly, for a dictionary size of M atoms, we get a set of M similarity scores that capture the intra-writer information. These are then fused using a simple linear combination with the M saliency values (obtained in the first step) thereby resulting in the adaptation (in the form of weights) to a specific writer.

We now provide the mathematical framework for adapting the saliency values of the dictionary atoms to each of the W writers. To begin with, for every feature vector $\mathbf{f}_j^k$ of the j^{th} substroke from the k^{th} writer in $\mathbb{S}_k$, we compute similarity scores s_{ij}^k that are related to the reconstruction errors obtained separately from each of the M atoms of the dictionary.

$$s_{ij}^k = \frac{1}{1+ \parallel \mathbf{f}_j^k - \alpha_{ij}^k \times \boldsymbol{\phi}_i \parallel_2} \qquad 1 \leq i \leq M, \quad 1 \leq j \leq |\mathbb{S}_k|, \quad 1 \leq k \leq W \tag{4.14}$$

The obtained similarity scores for a dictionary atom $\boldsymbol{\phi}_i$ are then averaged with regards to each writer separately and normalized as follows

$$\bar{s}_{ik}^{norm} = \frac{\bar{s}_{ik}}{\bar{s}_{i1} + \bar{s}_{i2} + ... + \bar{s}_{iW}} \tag{4.15}$$

where

$$\bar{s}_{ik} = \frac{1}{|\mathbb{S}_k|} \sum_{j=1}^{|\mathbb{S}_k|} s_{ij}^k \tag{4.16}$$

is the average similarity score of the k^{th} writer corresponding to the atom $\boldsymbol{\phi}_i$. It may be noted that the sum of the normalized components in Equation 4.15 is one.

In our proposal, for each atom $\boldsymbol{\phi}_i$, the adapted saliency value w_{ik} with regards to the k^{th} writer is determined by a linear combination of the $\bar{s}_{ik}^{norm}$ with w_i.

$$w_{ik} = \beta \times \bar{s}_{ik}^{norm} + (1-\beta) \times w_i \qquad 1 \leq i \leq M, \quad 1 \leq k \leq W \tag{4.17}$$

Here $\{w_i\}_{i=1}^M$ denotes the set of M saliency values obtained without adaptation to a specific writer by employing either one of the strategies discussed in Section 4.3 and 4.4.

4.6.1 Proposed Identification Framework

In order to assign the authorship of an input test document, we employ an ensemble of SVMs as shown in Figure 4.6. The number of SVMs correspond to the writers enrolled in the system. Each SVM is trained to predict the probable score of a specific writer. Thereafter, the maximum of the scores obtained is used to retrieve the identity of the writer.

During the training process of a specific SVM in the ensemble, the descriptors obtained from the enrolled documents of the writer in question are treated as positive samples. The negative samples comprise the descriptors from the training documents of all the remaining $W-1$ writers. Though this framework bears semblance to the one-versus-all SVM, there is an important distinction with regards to the data being employed to train the individual classifiers. In our approach, the parameters of the k^{th} SVM of the ensemble are learned from descriptors, obtained by incorporating the M saliency adapted values for the k^{th} writer.

Mathematically, for an input test document, we first generate the M dimensional vector $\mathbf{z}$ by either of pooling strategies described in Section 3.5. Prior to utilizing the k^{th} SVM of the ensemble, we modify this vector by incorporating the M saliency adapted values pre-computed for the k^{th} writer. In other words, we have,

$$\hat{z}_{ik} = w_{ik} \times z_i \qquad 1 \leq i \leq M, \qquad 1 \leq k \leq W \tag{4.18}$$

If we denote the vector $\mathbf{w}_k$ to represent the saliency values for the k^{th} writer, we can rewrite the above equation using an element by element multiplication operation as

$$\hat{\mathbf{z}}_k = \mathbf{w}_k \odot \mathbf{z} \tag{4.19}$$

where $\mathbf{w}_k = [w_{1k} \;\; w_{2k} \;\; ..... \;\; w_{Mk}]^T$. Note that $\hat{\mathbf{z}}_k$ corresponds to the modified descriptor that is fed to the k^{th} SVM of the ensemble.

For completion and ease of clarity, we provide a high level summary of the proposed writer identification framework in pseudo-code (Algorithm 6), with the adapted saliency values of dictionary atoms.

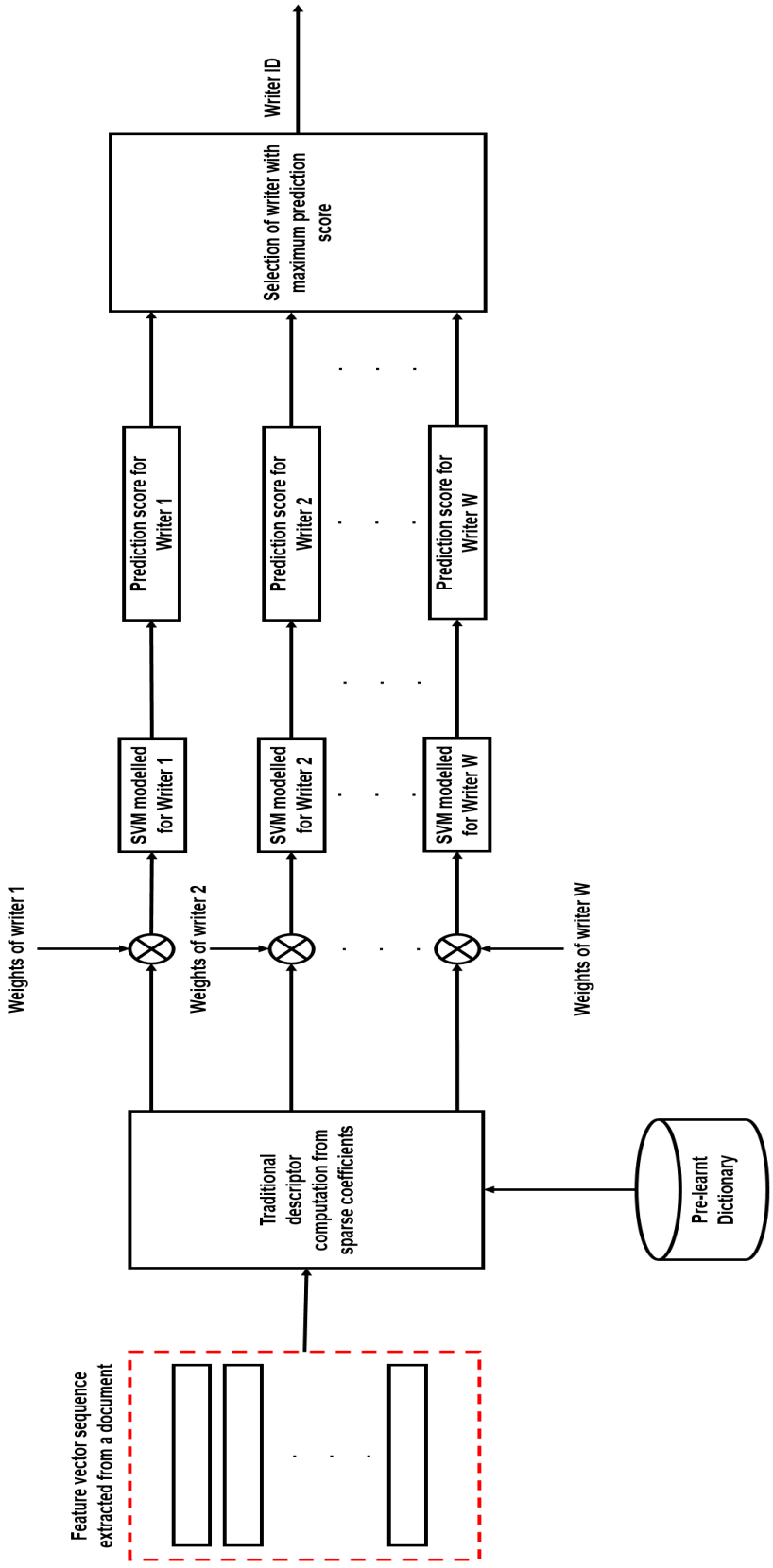


Figure 4.6: Block diagram of the proposed identification framework that makes use of an ensemble of SVMs to predict the author of the test document. The input to each of the SVM trained for a specific writer is a descriptor obtained by incorporating the appropriate saliency adapted values. For more details, refer the text.

Algorithm 6 Proposed identification system based on writer adopted saliency values

- **Input:**
  - A set of histogram based feature vectors $\{\mathbf{f}_j\}_{j=1}^{N_{Tr}}$ extracted from N_{Tr} sub-strokes corresponding to enrolled training documents of W writers.
  - Test document containing N_T feature vectors.
  - Dictionary $\mathbf{D}$ with M atoms $\{\boldsymbol{\phi}_i\}_{i=1}^{M}$.
- **Output:** Writer identity
- **Algorithm:**

% `Training`

Compute saliency values of a dictionary w_i by employing either Algorithm 3 or 4.

`% Calculation of writer-specific saliency values`

For every writer enrolled in the system
For every dictionary atom $\boldsymbol{\phi}_i$ in $\mathbf{D}$
- Calculate the similarity scores s_{ij}^k across all the enrolled feature vectors belonging to the writer in question.
- Derive the normalized average similarity scores $\bar{s}_{ik}^{norm}$ using Equation 4.15.
- Utilize Equation 4.17 to compute the adapted saliency value.

End For
End For

%`Testing`

Obtain traditional descriptor $\mathbf{z}$ by using the average pooling strategy.

For k= 1: Number of Writers W
Utilize the modified descriptor $\mathbf{z}_k$ as an input to the k^{th} SVM of the ensemble.
Obtain the predictive score of the k^{th} writer from the k^{th} SVM of the ensemble.
End For

Assign the authorship to the writer with the maximum score.

4.7 Experiments and discussion

We start this Section by outlining the various writer identification systems implemented, together with their abbreviations.

- *SPF*: This system corresponds to the traditional sparse representation framework based on average / max pooling (without the incorporation of saliency).
- *EN-SL*: This system refers to the modified sparse representation framework based on average / max pooling with the incorporation of saliency obtained from the entropy approach.
- *SP-SL*: This system corresponds to the modified sparse representation framework based on average / max pooling with the incorporation of saliency obtained from the sum-pooling approach.
- *SAL-ADP*: refers to the system where the modified writer descriptor is computed by incorporating writer-specific saliency values to the writer descriptor obtained via traditional pooling strategies.

With regards to the training and evaluation protocols, we employ the same as discussed in Section 2.6.2 for the IAM and IBM-UB1 databases. For each database, the atoms of the dictionary are learnt from the enrolled data corresponding to 25% of the number of users, that are randomly chosen.

4.7.1 Influence of the saliency based incorporation

In this experiment, we compare the performance of the *EN-SL* and *SP-SL* for writer identification over the traditional *SPF* system (with average based pooling approach). The results obtained at the paragraph and text-line levels are shown in Table 4.1 (a) and (b) for varying dictionary sizes on the IAM and IBM-UB1 databases.

From Table 4.1 (a), we observe that the performance of the two modified systems attains a best performance of 99.28% and 99.07% for $M = 400$ at the paragraph level on the IAM

database. In contrast, the traditional descriptors of the *SPF* system trails behind, achieving a best average identification rate of 98.41% again at $M = 400$. Likewise, at text-line level, we see an improvement in performance accuracy from 87.69% (*SPF* system) to 89.54% (*EN-SL* system) and 88.61% (*SP-SL* system). Not to mention, across the different dictionary sizes considered, at both paragraph and text-line levels, the modified writer descriptors with saliency provides a better identification rate than that obtained without their incorporation. It may be noted here that a similar trend of improvement is also observed with respect to the IBM-UB1 database in Table 4.1 (b).

Further, we also consider experimenting our proposed frameworks of saliency value incorporation on the traditional max pooling strategy. The results enumerated in Table 4.2 (a) and (b) on both the databases do show a promising trend with the modified writer descriptor at both paragraph and text-line level across all the dictionary sizes.

We now demonstrate that the results of the proposed writer descriptors with the incorporation of the saliency values for the dictionary atoms is statistically significant when compared to the traditional averaged and max pooling classification frameworks. The statistical check has been performed using the Student's t-test for a significance level of 0.05. Table 4.3 (a) and (b) outlines the p-values obtained with regards to the systems - *EN-SL* and *SP-SL* at both paragraph and text-line levels on the IAM and IBM-UB1 databases. A trend of low values for p is empirically observed across each of the entries in the Table - thus indicating the statistical significance of our proposal.

4.7.2 Influence of writer-specific adaptation of saliency values

In this section, we present the efficacy of our proposed $SAL - ADP$ system, wherein the saliency values are adapted for each writer separately by considering the linear combination in Equation 4.17. To begin with, we investigate the influence of the weight β on the identification performance by varying its value from zero to one in steps of 0.1. We employ the entropy approach discussed in Section 4.3 for obtaining the saliency of dictionary atoms. Moreover, the descriptor to each SVM in the ensemble of Figure 4.6 is obtained by incorporating the adapted

Table 4.1: Comparison of average writer identification rates (in %) for the descriptor obtained via the traditional average pooling approach and our proposals *EN-SL* and *SP-SL* with varying dictionary sizes M. The best average identification rate is marked in **bold**.

Dictionary size M	Paragraph-level			Text-level		
	SPF system	*EN-SL* system	*SP-SL* system	*SPF* system	*EN-SL* system	*SP-SL* system
50	92.45	93.60	92.92	64.29	68.92	66.40
100	93.33	94.80	94.15	71.19	73.25	72.07
150	94.50	96.18	95.83	74.42	76.44	75.73
200	95.84	97.07	96.62	78.70	79.62	79.11
250	96.78	97.82	97.48	80.99	82.59	82.03
300	97.65	98.52	98.11	84.99	86.84	85.97
350	98.09	98.99	98.57	86.51	88.65	87.49
400	**98.41**	**99.28**	**99.07**	**87.69**	**89.54**	**88.61**
450	98.27	99.21	98.96	87.44	89.43	88.35
500	98.16	99.13	98.89	87.25	89.27	88.16

(a) IAM Database

Dictionary size M	Paragraph-level			Text-level		
	SPF system	*EN-SL* system	*SP-SL* system	*SPF* system	*EN-SL* system	*SP-SL* system
50	72.65	74.33	73.34	52.53	54.64	53.05
100	76.16	78.19	77.10	60.25	62.72	61.17
150	80.29	81.77	81.16	64.95	67.67	66.75
200	83.14	84.46	83.89	69.35	71.59	70.13
250	87.28	88.56	88.07	72.14	74.46	73.07
300	91.10	92.45	91.92	76.30	78.82	77.35
350	93.64	94.88	94.03	78.25	80.27	79.11
400	**95.16**	**96.37**	**95.78**	**79.47**	**81.61**	**80.35**
450	95.03	96.24	95.64	79.29	81.53	80.25
500	94.91	96.16	95.53	79.18	81.46	80.18

(b) IBM-UB1 Database

Table 4.2: Comparison of the average writer identification rates (in %) for the descriptor obtained via the traditional max pooling approach and our proposals *EN-SL* and *SP-SL* with varying dictionary sizes M. The best average identification rate is marked in **bold**.

Dictionary size M	Paragraph-level			Text-level		
	SPF system	*EN-SL* system	*SP-SL* system	*SPF* system	*EN-SL* system	*SP-SL* system
50	43.06	54.59	51.83	26.84	28.52	27.63
100	79.38	81.13	80.56	44.42	46.53	45.49
150	89.81	91.78	90.74	59.22	62.11	61.20
200	92.94	94.10	93.67	65.64	67.87	66.84
250	95.95	96.69	96.19	75.13	77.44	76.57
300	96.61	97.26	96.84	78.55	80.42	79.75
350	97.47	97.84	97.66	82.16	84.21	83.81
400	**98.03**	**98.56**	**98.25**	**85.89**	**87.91**	**86.95**
450	97.92	98.41	98.17	85.69	87.58	86.79
500	97.83	98.32	98.09	85.54	87.42	86.64

(a) IAM Database

Dictionary size M	Paragraph-level			Text-level		
	SPF system	*EN-SL* system	*SP-SL* system	*SPF* system	*EN-SL* system	*SP-SL* system
50	71.34	73.75	72.53	50.49	53.44	52.09
100	75.41	77.91	76.80	58.16	60.32	59.45
150	79.76	81.37	80.64	62.78	66.18	64.57
200	82.37	83.80	83.25	67.76	69.95	68.78
250	86.50	88.07	87.57	70.07	73.83	72.36
300	90.44	91.99	91.29	74.83	77.31	76.29
350	92.69	94.15	93.58	77.39	79.14	78.21
400	**94.28**	**95.79**	**94.98**	**78.57**	**80.66**	**79.72**
450	94.16	95.66	94.83	78.34	80.49	79.60
500	94.02	95.53	94.71	78.19	80.37	79.47

(b) IBM-UB1 Database

Table 4.3: Statistical significance of the *EN-SL* and *SP-SL* over the average and max pooling strategies for both the databases.

Database	Classification framework	Paragraph Level	Text line level
IAM	Average pooling	1.30×10^{-3}	1.98×10^{-4}
	Max pooling	2.20×10^{-3}	1.79×10^{-6}
IBM-UB1	Average pooling	1.02×10^{-4}	2.66×10^{-7}
	Max pooling	2.29×10^{-5}	9.33×10^{-7}

(a) EN-SL system

Database	Classification framework	Paragraph Level	Text line level
IAM	Average pooling	1.93×10^{-2}	2.31×10^{-2}
	Max pooling	4.08×10^{-2}	6.39×10^{-4}
IBM-UB1	Average pooling	1.41×10^{-2}	3.50×10^{-3}
	Max pooling	4.20×10^{-3}	5.79×10^{-4}

(b) SP-SL system

saliency values to the SPF system, that is based on the average pooling strategy [5]

The best average identification rates are presented in Table 4.4 for both the databases. As for the trend, the identification rate increases with β, attains a maximum and then decreases. The best identification rates at paragraph and text line levels are obtained at a β value of 0.5 and 0.6 for the IAM and the IBM-UB1 database respectively. It may be noted here that the value $\beta = 0$ relates to the performance of the *EN-SL* system.

Further to the above, we analyse the performance of the *SAL-ADP* system for varying number of dictionary atoms M. We consider the dictionary size from 50 to 100 in increments of 50 and values of β from 0 to 1 in steps of 0.1. From Table 4.5, we see that the highest writer identification rates at paragraph and text line level for IAM database are at 99.54% and 91.26% respectively for M= 400. With respect to the IBM-UB1 database, an identification rate of 97.24% and 83.54 % is achieved - once again for the same dictionary size.

[5]The SPF with average pooling strategy outperforms over the max pooling approach, thus justifying our choice.

Table 4.4: Influence of the weighting factor β used for adapting the saliency values on writer identification.

Database	Training framework	β value						
		0	0.2	0.4	0.5	0.6	0.8	1
IAM	Paragraph level	99.28	99.37	99.46	**99.54**	99.41	99.10	98.77
	Text line level	89.54	90.23	90.89	**91.26**	90.37	88.96	88.04
IBM-UB1	Paragraph level	96.37	96.62	96.86	97.10	**97.24**	97.03	96.77
	Text line level	81.61	82.17	82.73	83.09	**83.54**	82.96	82.49

Table 4.5: Influence of dictionary size on the modified writer descriptor obtained from the writer-specific saliency adapted values. For this experiment, the saliency is computed using the entropy-based approach discussed in Section 4.3.

Dictionary size M	IAM database				IBM-UB1 database			
	Paragraph level	β	Text line level	β	Paragraph level	β	Text line level	β
100	98.73	0.5	81.36	0.4	79.98	0.6	66.84	0.5
200	98.94	0.6	86.49	0.4	86.22	0.5	75.01	0.6
250	99.19	0.4	88.29	0.5	90.19	0.6	78.97	0.5
300	99.31	0.5	89.47	0.6	93.84	0.6	82.33	0.5
350	99.42	0.4	90.66	0.4	95.36	0.5	82.95	0.6
400	**99.54**	0.5	**91.26**	0.5	**97.24**	0.6	**83.54**	0.6
450	99.41	0.6	90.71	0.6	97.11	0.5	83.49	0.5
500	99.27	0.5	89.99	0.5	97.03	0.6	83.42	0.5

Table 4.6 provides the performance of the $SAL - ADP$ system, for the case where the saliency values are obtained by employing the sum-pooling strategy of Section 4.4. The best results are obtained at a dictionary size of $M = 400$ for both the paragraph and text-line levels.

For completeness, we also tabulate the p values, as obtained by Student's t-test in Table 4.7. The low values (p being less than 0.05) do suggest that the results of *SAL-ADP* are statistically significant with the $EN - SL$ and $SP - SL$ based systems that do not perform adaptation to each writer.

Table 4.6: Influence of dictionary size on the modified writer descriptor obtained from the writer-specific saliency values. For this experiment, the saliency is computed using the sum pooling based approach discussed in Section 4.4.

Dictionary size M	IAM database				IBM-UB1 database			
	Paragraph level	β	Text line level	β	Paragraph level	β	Text line level	β
100	97.34	0.4	74.39	0.5	78.70	0.6	62.15	0.5
200	98.43	0.6	81.52	0.4	84.56	0.5	71.04	0.6
250	98.70	0.6	84.38	0.6	88.68	0.6	74.57	0.5
300	99.04	0.5	86.70	0.5	92.49	0.5	78.47	0.4
350	99.21	0.4	88.59	0.4	94.42	0.4	80.10	0.5
400	**99.37**	0.5	**90.43**	**0.5**	**96.38**	**0.5**	**81.53**	**0.6**
450	99.29	0.4	90.27	0.4	96.21	0.6	81.34	0.4
500	99.17	0.5	90.01	0.6	96.13	0.5	81.07	0.4

Table 4.7: Statistical significance of the performance of the saliency adapted based writer descriptors via the Student's t-test over those without adaptation.

Database		Paragraph Level	Text line level
IAM	*EN-SL*	4.32×10^{-2}	9.92×10^{-4}
	SP-SL	4.91×10^{-2}	7.63×10^{-6}
IBM-UB1	*EN-SL*	5.10×10^{-3}	5.97×10^{-7}
	SP-SL	1.86×10^{-2}	9.80×10^{-4}

4.8 Conclusion

In this Chapter, we presented a sparse classification based writer identification system for online handwritten documents by introducing a-priori (saliency) information for each dictionary atom. The saliency values quantify in an average sense, the degree of importance of each of the dictionary atoms with regards to the dynamic characteristics of the enrolled writers. We proposed two strategies for computing the saliency values. The first methodology relies on entropies that computed over several histograms - the number of which corresponds to the size of the dictionary used. Contrary to this, in the second methodology, only a single histogram is used, wherein the sparse coefficients resulting from the sum pooling operation are employed to

obtain the saliency values.

In addition to the above contributions, we also proposed a strategy for adapting the saliency values to each of the writers enrolled to the system. In order to identify the writer, we formulate an ensemble of SVMs, wherein the input to the SVM trained for a writer is based on the adapted saliency values for that writer. Lastly, we empirically showed through several experiments that the incorporation of saliency values (with / without adaptation) outperforms the traditional max and average pooling of sparse codes on the handwritten samples of the IAM and IBM-UB1 respectively.

5

Summary

Contents

5.1 List of contributions

In this section, we summarize on the different contributions made in this thesis.

- Investigation on the merit of the Vector of Locally Aggregated Descriptor (VLAD) approach, a codebook based descriptor popularly used in the field of image retrieval for online writer identification.
- Demonstration of a potential drawback with the VLAD framework for writer identification.
- Proposal of a novel codebook based descriptor that alleviates the limitation of the VLAD and its variants.
- Empirical study of a reduced dimension version of our proposed codebook descriptor and its comparison to the VLAD.
- For constructing the codebook, and subsequently the descriptor, point based features are proposed by specifying a gap parameter. The value of the parameter ensures that the features extracted capture the neighbourhood information of the point under consideration.
- Exploration of sparse dictionary framework to the domain of online writer identification. The main crux is in utilizing novel information from the sparse coefficients to generate writer descriptors.
- Proposal of novel histogram based features for shape description of the sub-strokes segmented from the online trace. These have been inspired by the idea of Histogram of Oriented Gradients, that is popular in the area of image processing.
- Utilization of the histogram based feature vectors and their sparse coefficients, for deriving sparse coded descriptors. The descriptors provide the description of the writer at a finer level, when compared to the traditional max / average pooling strategies.

- Proposal of an entropy based analysis for the appropriate bin size to be selected for obtaining the features so as to ensure discrimination between the writer descriptors.

- Modification of the traditional average / max pooling writer descriptor by incorporating the derived saliency values of the dictionary atoms, that are pre-learnt during the training phase.

- Exploration of two novel schemes to quantify the saliency values of the dictionary atoms with regards to the average dynamic characteristics of a writer. The first proposal make use of entropy measure computed over histograms generated from sparse coefficients while the second utilizes a histogram obtained from sum pooling the sparse coefficients across all the writers.

- Proposal to adapt the saliency values of the dictionary atoms for each writer separately by taking into consideration the characteristics of his / her handwriting.

- Formulation of an ensemble of SVMs, wherein the descriptor to the SVM trained for a writer is based on the saliency values adapted for that writer. The final decision on the authorship is proposed as the maximum of the prediction score obtained from the SVMs.

5.2 Discussion of prior works

The goal of this Section is to see on how the proposed strategies of this thesis fares with works from literature that use the IAM and IBM-UB1 databases. We summarize on the performances obtained from each of the contributing Chapters in Table 5.1. It can be noticed that the sparse representation based approaches provide an improved performance when compared to the codebook based descriptor. This is owing to the flexibility that the sparse representation methods provide in terms of representing a substroke as a linear combination of the atoms in the dictionary. Contrast to that, the codebook based approach allows each feature vector to get associated with one codevector only.

Table 5.1: Summary of results of proposals of thesis on the IAM and IBM-UB1 database.

Database	Proposal	Paragraph Level	Text line level
IAM	Chapter 2	98.82	89.62
	Chapter 3	99.45	90.28
	Chapter 4	**99.54**	**91.26**
IBM-UB1	Chapter 2	96.10	81.59
	Chapter 3	97.21	83.49
	Chapter 4	**97.24**	**83.54**

An enumeration of the previous explorations are thereafter presented in Table 5.2. The entries are the writer identification rates (in %) as reported from the respective references. It may be noted that the systems being outlined can differ with regards to the features as well as the classifier and enrolment strategies. Therefore, it is to be borne in mind that a direct one to one comparison to these approaches may not be fair.

To the best of our knowledge, the writer identification systems proposed in [13,29,44] utilize the IAM database for experimentation. The authors of [29] use a GMM based framework and present an identification rate of 98.56% and 88.96% at paragraph and text line level. The concept of Latent Dirichlet Allocation utilized in [13] and the subtractive clustering based approach of [44] lead to a paragraph-level writer identification rate of 93.39% and 96.30 % respectively. In comparison, our proposals of the thesis provide a competent performance at paragraph and text line level respectively.

For the IBM-UB1 database, there has been only one exploration till date [13] wherein, a writer identification rate of 89.47 % is reported at paragraph level. Once again we do better than this work.

5.3 Possible pointers for the future

We now present possible research avenues that can be taken up in future.

- The generation of the codebook based descriptor of Chapter 2 is derived by accumulating

Table 5.2: Survey of online writer identification systems on the IAM and IBM-UB1 database. The numbers are the writer identification rates (in %) as reported from the respective references. Moreover, entries in the table marked with (-) symbol indicate that the respective authors have not reported the results for them.

Database	Method	Paragraph Level	Text line level
IAM	GMM based likelihood system [29]	98.56	88.96
	Latent Dirichlet Allocation (LDA) [13]	93.39	-
	Subtractive Clustering [44]	96.30	-
IBM-UB1	Latent Dirichlet Allocation (LDA) [13]	89.47	-

scores that are a function of the residual / distortion between a feature vector and the nearest codevector. In this regard, only the first order statistic of the handwritten data of a writer has been considered in the description. A possible investigation could thus be in the direction of incorporating second order statistics [77, 78] into our formulation study.

- In Chapter 2, the point based features used for obtaining the codebook based descriptors are parametrized by a gap parameter. In the present thesis, each of the feature attributes is computed by considering the same value of this parameter. A possible avenue for the future is to adapt these values for each feature attribute individually and accordingly judge the performance.

- In Chapter 3 and 4, we focus mainly on generating descriptors based on the pre-learnt over complete dictionary of atoms. As such, the choice of dictionary learning algorithms has not been considered in the thesis. It would be interesting to see how the performance of our proposed writer identification systems would relate to the choice of the dictionary learning algorithm being used.

- The ideas proposed in the three contributing Chapters of this thesis may be extended for the task of offline writer identification. The road-map of steps would be to first segment out the allographs or letter parts and then describe each patch thereof with features such as SIFT or RootSIFT [21]. The extracted features be used to generate the codebook

/ dictionary and subsequently the writer descriptor. At this juncture, it may be worth mentioning that, in place of hand crafted SIFT based features, one can also attempt to describe the allographs with Convolutional Neural Network activation features as done in [79, 80].

- The strategies proposed in this thesis consider the case of single script writer identification where in a writer has written in one language only. A possible investigation would be to analyse the trend of these systems in a multi-script writer identification scenario.

- In the traditional SVM formulation, given a new handwriting sample from an unseen writer, all models need to be retrained. This issue is however alleviated by the use of the Exemplar-SVM that has been proposed in the area of object detection [81]. Very recently, such a framework has found much success for offline writer identification [21]. The adaptation of the same to online writer identification can be a possible avenue for investigation.

Notwithstanding the above extensions, the present thesis is the first of its kind to propose interesting novel writer descriptors based on the code vectors of codebook and atoms of a dictionary

References

[1] M. Anderson, "The Demographics of Device Ownership." [Online]. Available: https://www.pewinternet.org/2015/10/29/the-demographics-of-device-ownership/

[2] A. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 14, no. 1, pp. 4–20, Jan 2004.

[3] B. Zhang, W. Li, P. Qing, and D. Zhang, "Palm-print classification by global features," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 43, no. 2, pp. 370–378, 2013.

[4] A. Sharma and S. Sundaram, "A Novel Online Signature Verification System Based on GMM Features in a DTW Framework," *IEEE Transactions on Information Forensics and Security*, vol. 12, pp. 705–718, 2017.

[5] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of Fingerprint Recognition*, 2nd ed. Springer Publishing Company, 2009.

[6] J. Lu, V. E. Liong, X. Zhou, and J. Zhou, "Learning compact binary face descriptor for face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, pp. 2041–2056, 2015.

[7] Z. Sun, H. Zhang, T. Tan, and J. Wang, "Iris image classification based on hierarchical visual codebook," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, pp. 1120–1133, 2014.

[8] L. Fei, B. Zhang, Y. Xu, and L. Yan, "Palmprint recognition using neighboring direction indicator," *IEEE Transactions on Human-Machine Systems*, vol. 46, pp. 787–798, 2016.

[9] A. Sharma and S. Sundaram, "On the exploration of information from the DTW cost matrix for online signature verification," *IEEE Transactions on Cybernetics*, vol. 48, pp. 611–624, 2018.

[10] A. Morales, J. Fiérrez, R. Tolosana, J. Ortega-Garcia, J. Galbally, M. Gomez-Barrero, A. Anjos, and S. Marcel, "Keystroke biometrics ongoing competition," *IEEE Access*, vol. 4, pp. 7736–7746, 2016.

[11] Y. Zhang, G. Pan, K. Jia, M. Lu, Y. Wang, and Z. Wu, "Accelerometer-based gait recognition by sparse representation of signature points with clusters," *IEEE Transactions on Cybernetics*, vol. 45, pp. 1864–1875, 2015.

[12] M. L. Bulacu, "Statistical pattern recognition for automatic writer identification and verification," 2007.

[13] A. Shivram, C. Ramaiah, and V. Govindaraju, "A hierarchical Bayesian approach to online writer identification," *IET Biometrics*, vol. 2, no. 4, pp. 191–198, 2013.

[14] A. Schlapbach, *Writer Identification and Verification*. Akademische Verlagsgesellschaft, 2008.

[15] M. Liwicki and H. Bunke, *Recognition of Whiteboard Notes: Online, Offline and Combination*. World Scientific, 2008.

[16] M. Liwicki, A. Schlapbach, H. Bunke, S. Bengio, J. Mariéthoz, and J. Richiardi, "Writer identification for smart meeting room systems," in *International Workshop on Document Analysis Systems*. Springer, 2006, pp. 186–195.

[17] A. Bensefia, T. Paquet, and L. Heutte, "A writer identification and verification system," *Pattern Recogn. Lett.*, vol. 26, no. 13, pp. 2080–2092, Oct. 2005.

[18] I. Siddiqi and N. Vincent, "Text independent writer recognition using redundant writing patterns with contour-based orientation and curvature features," *Pattern Recognition*, vol. 43, no. 11, pp. 3853 – 3865, 2010.

[19] D. Bertolini, L. S. Oliveira, E. Justino, and R. Sabourin, "Texture-based descriptors for writer identification and verification," *Expert Systems with Applications*, vol. 40, no. 6, pp. 2069–2080, 2013.

[20] A. Nicolaou, A. D. Bagdanov, M. Liwicki, and D. Karatzas, "Sparse radial sampling LBP for writer identification," in *13th International Conference on Document Analysis and Recognition (ICDAR 2015)*, 2015, pp. 716–720.

[21] V. Christlein, D. Bernecker, F. Hnig, A. Maier, and E. Angelopoulou, "Writer identification using GMM supervectors and exemplar-SVMs," *Pattern Recognition*, vol. 63, pp. 258 – 267, 2016.

[22] A. Brink, J. Smit, M. Bulacu, and L. Schomaker, "Writer identification using directional ink-trace width measurements," *Pattern Recognition*, vol. 45, no. 1, pp. 162 – 171, 2012.

[23] S. He and L. Schomaker, "Co-occurrence features for writer identification," in *15th International Conference on Frontiers in Handwriting Recognition, ICFHR 2016, Shenzhen, China, October 23-26, 2016*, 2016, pp. 78–83.

[24] J. Wen, B. Fang, J. Chen, Y. Tang, and H. Chen, "Fragmented edge structure coding for Chinese writer identification," *Neurocomputing*, vol. 86, no. Supplement C, pp. 45 – 51, 2012.

[25] R. Kumar, B. Chanda, and J. Sharma, "A novel sparse model based forensic writer identification," *Pattern Recognition Letters*, vol. 35, pp. 105 – 112, 2014, frontiers in Handwriting Processing.

[26] X. Wu, Y. Tang, and W. Bu, "Offline text-independent writer identification based on scale invariant feature transform," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 3, pp. 526–536, 2014.

[27] R. Plamondon, G. Pirlo, É. Anquetil, C. Rémi, H. Teulings, and M. Nakagawa, "Personal digital bodyguards for e-security, e-learning and e-health: A prospective survey," *Pattern Recognition*, vol. 81, pp. 633–659, 2018.

[28] A. M. Namboodiri and S. Gupta, "Text independent writer identification from online handwriting," in *Proc. International Workshop on Frontiers in Handwriting Recognition*, vol. 10, October 2006, pp. 287–292.

[29] A. Schlapbach, M. Liwicki, and H. Bunke, "A writer identification system for online whiteboard data," *Pattern recognition*, vol. 41, no. 7, pp. 2381–2397, 2008.

[30] A. Schlapbach and H. Bunke, "Fusing asynchronous feature streams for online writer identification," in *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, vol. 1. IEEE, 2007, pp. 103–107.

[31] M. Liwicki, A. Schlapbach, and H. Bunke, "Writer-dependent recognition of handwritten whiteboard notes in smart meeting room environments," in *Document Analysis Systems, 2008. DAS'08. The Eighth IAPR International Workshop on.* IEEE, 2008, pp. 151–157.

[32] D. A. Reynolds, "Speaker identification and verification using Gaussian mixture speaker models," *Speech communication*, vol. 17, no. 1, pp. 91–108, 1995.

[33] 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.

[34] G. X. Tan, C. Viard-Gaudin, and A. C. Kot", "Automatic writer identification framework for online handwritten documents using character prototypes," *Pattern Recognition*, vol. 42, no. 12, pp. 3313 – 3323, 2009.

[35] G. Salton, A. Wong, and C.-S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[36] G. Tan, C. Viard-Gaudin, and A. Kot, "Online writer identification using fuzzy c-means clustering of character prototypes," in *ICFHR*, 2008, pp. 475–480.

[37] S. K. Chan, C. Viard-Gaudin, and Y. H. Tay, "Online writer identification using character prototypes distributions," in *Electronic Imaging*, 2008, pp. 68 150H–1–68 150H–9.

[38] G. X. Tan, C. Viard-Gaudin, and A. C. Kot, "Individuality of alphabet knowledge in online writer identification," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 13, no. 2, pp. 147–157, 2010.

[39] K. Yu, Y. Wang, and T. Tan, "Writer identification using dynamic features." in *First International Conference on Biometric Authentication*, 2004, pp. 512–518.

[40] B. Li, Z. Sun, and T. Tan, "Online text-independent writer identification based on stroke's Probability Distribution Function," in *International Conference on Biometrics.* Springer, 2007, pp. 201–210.

[41] 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.

[42] 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.

[43] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[44] 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*, Aug 2015.

[45] A. Chaabouni, H. Boubaker, M. Kherallah, A. M. Alimi, and H. El Abed, "Multi-fractal modeling for online text-independent writer identification," in *2011 International Conference on Document Analysis and Recognition*, 2011, pp. 623–627.

[46] M. Gargouri, S. Kanoun, and J.-M. Ogier, "Text-independent writer identification on online Arabic handwriting," August 2013, pp. 428–432.

[47] J. Chapran, "Biometric writer identification: feature analysis and classification,," *International Journal on Pattern Recognition and Artificial Intelligence*, vol. 20, pp. 483–503, 2006.

[48] J. Chapran and M. Fairhurst, "Biometric writer identification based on the interdependency between static and dynamic features of handwriting," in *Proceedings of the 10th International Workshop on Frontiers in Handwriting Recognition*, 2006, pp. 505–510.

[49] 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*, vol. 2. IEEE, 2005, pp. 1264–1268.

[50] T. Dhieb, W. Ouarda, H. Boubaker, and A. M. Alimi, "Beta-elliptic model for online writer identification from online Arabic handwriting," *Journal of Information Assurance & Security*, vol. 11, no. 5, pp. 263–272, 2016.

[51] T. Dhieb, W. Ouarda, H. Boubaker, M. B. Halima, and A. M. Alimi, "Online Arabic writer identification based on beta-elliptic model," in *Proceedings of the International Conference on Intelligent Systems Design and Application (ISDA)*, 2015, pp. 74–79.

[52] T. Dhieb, W. Ouarda, H. Boubaker, and A. M. Alimi, "Deep neural network for online writer identification using beta-elliptic model," in *International Joint Conference on Neural Networks*, 2016, pp. 1863–1870.

[53] A. Chaabouni, H. Boubaker, M. Kherallah, A. M. Alimi, and H. El Abed, "Combining of offline and online feature extraction approaches for writer identification," in *Document Analysis and Recognition (ICDAR), 2011 International Conference on.* IEEE, 2011, pp. 1299–1303.

[54] A. Chaabouni, H. Boubaker, M. Kherallah, H. El-Abed, and A. Alimi, "Static and dynamic features for writer identification based on multi-fractals," *International Arab Journal of Information Technology*, vol. 11, no. 4, pp. 416–424, 2014.

[55] A. Shivram, C. Ramaiah, U. Porwal, and V. Govindaraju, "Modeling writing styles for online writer identification: a hierarchical bayesian approach," in *Frontiers in Handwriting Recognition (ICFHR), 2012 International Conference on*, 2012, pp. 387–392.

[56] C. Ramaiah and V. Govindaraju, "A hierarchical framework for accent based writer identification," in *Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on.* IEEE, 2014, pp. 21–25.

[57] C. Ramaiah, A. Shivram, and V. Govindaraju, "A Bayesian framework for modeling accents in handwriting," in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on.* IEEE, 2013, pp. 917–921.

[58] Ramaiah, Chetan and Shivram, Arti and Govindaraju, Venu, "Data sufficiency for online writer identification: A comparative study of writer-style space vs feature space models," in *Pattern Recognition (ICPR), 2014 International Conference on*, 2014, pp. 3121–3125.

[59] W. Yang, L. Jin, and M. Liu, "Chinese character-level writer identification using path signature feature, dropstroke and deep cnn," in *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, 2015, pp. 546–550.

[60] Yang, Weixin and Jin, Lianwen and Liu, Manfei, "DeepWriterID: An End-to-End Online Text-Independent Writer Identification System," *IEEE Intelligent Systems*, vol. 31, no. 2, pp. 45–53, 2016.

[61] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[62] X. Y. Zhang, G. S. Se, C. L. Liu, and Y. Bengio, " End-to-End Online Writer Identification With Recurrent Neural Network," *IEEE Trans. Human Machine Systems*, 2017.

[63] H. Jegou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1704–1716, 2012.

[64] R. Arandjelović and A. Zisserman, "All about VLAD." in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[65] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. Journal on Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[66] K. Barkoula, G. Economou, and S. Fotopoulos, "Online signature verification based on signatures turning angle representation using longest common subsequence matching," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 16, no. 3, pp. 261–272, 2013.

[67] C. J. Burges, "A tutorial on Support Vector Machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998.

[68] M. Liwicki and H. Bunke, "IAM-OnDB - An On-Line English Sentence Database Acquired from Handwritten Text on a Whiteboard." in *ICDAR*, 2005, pp. 956–961.

[69] A. Shivram, C. Ramaiah, S. Setlur, and V. Govindaraju, "IBM_UB_1: A Dual Mode Unconstrained English Handwriting Dataset," in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, Aug 2013, pp. 13–17.

[70] H. Xiong, J. Wu, and J. Chen, "k-means clustering versus validation measures: a data-distribution perspective," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 318–331, 2009.

[71] J. Liang, L. Bai, C. Dang, and F. Cao, "The k-means-type algorithms versus imbalanced data distributions," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 4, pp. 728–745, 2012.

[72] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2010, pp. 3304–3311.

[73] J. Delhumeau, P.-H. Gosselin, H. Jégou, and P. Pérez, "Revisiting the VLAD image representation," in *Proceedings of the 21st ACM International Conference on Multimedia*, 2013, pp. 653–656.

[74] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, 2005, pp. 886–893.

[75] S. Furui, "Speaker-independent isolated word recognition based on emphasized spectral dynamics," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'86.*, vol. 11, 1986, pp. 1991–1994.

[76] F. Bach, J. Mairal, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *The Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[77] J. Sanchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image Classification with the Fisher Vector: Theory and Practice," *International Journal of Computer Vision*, vol. 105, no. 3, pp. 222–245, 2013.

[78] V. Garg, S. Chandra, and C. V. Jawahar, "Sparse Discriminative Fisher Vectors in Visual Classification," in *Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing*, ser. ICVGIP '12. New York, NY, USA: ACM, 2012, pp. 55:1–55:8.

[79] V. Christlein, D. Bernecker, A. Maier, and E. Angelopoulou, "Offline writer identification using Convolutional Neural Network activation features," in *German Conference on Pattern Recognition*. Springer, 2015, pp. 540–552.

[80] Y. Tang and X. Wu, "Text-independent writer identification via CNN features and joint Bayesian," in *15th International Conference on Frontiers in Handwriting Recognition, ICFHR 2016, Shenzhen, China, October 23-26, 2016*, 2016, pp. 566–571.

[81] T. Malisiewicz, A. Gupta, and A. A. Efros, "Ensemble of Exemplar-SVMs for Object Detection and Beyond," in *ICCV*, 2011.

LIST OF PUBLICATIONS

Journal Publications:

1. Vivek Venugopal and Suresh Sundaram, "Online writer identification with sparse coding based descriptors," IEEE Trans. Information Forensics and Security, vol. 33, no. 10, pp. 2538 - 2552, Oct 2018.

2. Vivek Venugopal and Suresh Sundaram, "An improved online writer identification framework using codebook descriptors," Pattern Recognition, vol. 78, pp. 318-330, June 2018.

3. Vivek Venugopal and Suresh Sundaram, "A Modified Sparse Representation Classification Framework for Online Writer Identification," IEEE Trans. Systems, Man, and Cybernetics: Systems, In press.

4. Vivek Venugopal and Suresh Sundaram, "An online writer identification system using regression-based feature normalization and codebook descriptors," Expert System with Application, vol. 72, pp. 196-206, April 2017.

Under Review:

1. Vivek Venugopal and Suresh Sundaram, " An adaptive sparse representation framework for online writer identification", submitted to Pattern Recognition Letters.

Conference Publications:

1. Isht Dwivedi, Swapnil Gupta, Vivek Venugopal and Suresh Sundaram, "Online Writer Identification Using Sparse Coding and Histogram Based Descriptors ", International Conference Frontiers in Handwriting Recognition 2016, pp. 572-577.