A Text Independent Writer Identification System from Handwritten Document Images

A Project Report

Submitted in partial fulfillment of the

requirements for the Degree of

Bachelor of Technology

in the Computer Science and Engineering

Submitted By

Tathagata Bandyopadhyay(1305382)

School of Computer Engineering

KIIT University

BHUBANESWAR – 751 024

April 9, 2017

A Text Independent Writer Identification System from Handwritten Document Images

A Project Report

Submitted in partial fulfillment of the

requirements for the Degree of

Bachelor of Technology

in the Computer Science and Engineering

Submitted By

Tathagata Bandyopadhyay(1305382)

School of Computer Engineering

KIIT University

BHUBANESWAR – 751 024

April 9, 2017

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

School of Computer Science and Engineering

Bhubaneswar, ODISHA 751024

CERTIFICATE

This is certify that the project entitled

"A Text Independent Writer Identification System from Handwritten Document Images"

submitted by

Tathagata Bandyopadhyay(1305382)

is a record of bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science and Engineering) at KIIT UNIVERSITY, Bhubaneswar. This work is done during year 2016-2017, under **my/our** guidance.

Date: / /

(Prof. Mamata Motwani)

Project Guide

TO

My father Mr. Pranab Kanti Bandyopadhyay

and

My mother Mrs. Nupur Bandyopadhyay

Acknowledgements

This project presented before you is a blessing of the Almighty, who through His utmost will has helped me during this project. I cordially thank professor *Dr. Nibaran Das* from Department of Computer Science and Engineering, Jadavpur University, Kolkata, who gave me the whole idea and also helped me in many of the image processing concepts. I am also greatful to one of my siniors, *Mr. Sandipan Choudhury*, whom I consulted many times for clearing many of my conceptual doubts during the project work. Here, I can not deny the contribution of my guide, *Ms. Mamata Motwani* for her time-to-time suggestions, reviews and ideas. I would like to express my gratitude to her for her constant motivation, kind co-operation and strong support during this whole project. I am thankful to my very own *KIIT University* for providing me such an exciting learning platform. And lastly, I do believe, any expression of gratitude on my part is not enough for the moral support and motivation I got from my parents regarding this project.

Tathagata Bandyopadhyay

1305382

Abstract

This project presents a GUI application for verifying the authenticity of a writer of a hand written text document. It takes a binarized image of hand written text and predicts the author id from a set of pre-enrolled authors. Approximate word regions are segmented using gaussian and motion blur and connected component analysis. Then, SURF features (SDS and SOH) are extracted from each of the word regions of the text images and are fed to an ELM classifier, which is trained with a feature database of pre-enrolled authors with known ids, for predicting the class label i.e. the author id for the new sample. The system also provides the functionality of enrolling a new author to the existing database and also training the system with the newly updated feature database. Author verification functionality can be accessed by any user, where as only an Admin has access to the new author enroll and classifier training functionality.

Contents

List of Tables

List of Figures

Keywords

text-Independent writer identification, SURF, word region segmentation, SURF Descriptor Signature, Scale and Orientation Histogram, Extreme Learning Machine.

Notation and Abbreviations

In the whole document, the terms *Author Database*, *Feature Database*, *Database* refers to the same data structure containing the feature values and corresponding Author ids. The Abbreviations used in the proposed mechanism are given in Table 1.

Table 1: Abbreviations

SIFT	Scale Invarient Feature Transform
SURF	Speed Up Robust Feature
SDS	SURF Descriptor Signature
SOH	Scale and Orientation Histogram
ELM	Extreme Learning Machine

Chapter 1

Introduction

Hand writing of an individual is a unique charecteristic of the writer just like a fingerprint [1], [2]. This enables us to identify the author of an handwritten text based on a pre-rgistered hand writing sample. Author identification from hand writing has found a wide area of applications starting from forensic analysis, document authoraization, examini verification to analysis of historical documents. With the advent of modern computer based image processing techniques, this field has become a very popular and active area of research in the domain of pattern recognication.

1.1 Writer Identification vs Verification

Although, both writer identification and verification system share a common subset of processes, there is a fundamental conceptual difference between the two systems. Bensefia *et al.* in [3], explains this difference. Writer verification is the task of saying whether two input hand writing samples are written by same person or not. Generally this do not require a database concept. Writer identification, on the other hand, deals with the retrival of the hand written documents from a database with a sample image as an graphical querry. A slight variation of it could be retrival of the author id from the

databse based on feature match with the input hand writing sample. This variation is used in this application.

1.2 Types of writer identification

Based on the hand writing sample recording techniques writer identification can be broadly of two types: (i) On-line and (ii) Off-line. These concepts are nicely explained in [4].

1.2.1 On-line writer identification

This type of system identifies the writer while he or she is writing. Here the hand writing is recorded with a stylus and pen enabled input devices like Pocket PC, Tablet PC, Graphic Tablet etc. Other than the shape and geometry of the hand written cherectors, online handwriting documents contain some temporal information about the author, like pen-up pen-down events, velocity and pressure of writing etc. This type of writer identification finds its application mostly in automated identity verification systems such as secure data access devices where user authentication can be done with a text written using a stylus [4].

1.2.2 Off-line writer identification

In this process, hand written documents are scanned to produce digital image of those samples. This type of images only captures the shape or geometrical features of the hand writing of a writer. This type of writer identification is mostly used in forensic analysis[5] and examination answer script verification. In this project this approach is used.

1.3 Approaches of text-independent off-line writer indentification

Based on the features used, text-independent off-line writer indentification system can be broadly classified in to two catagories: (i) Texture-based Approach and (ii) Structure-based Approach[6].

1.3.1 Texture-based Approach

In this approach, the whole hand written text is considered as a texture image and textural features are extracted to identify a writer. Local Binary Patterns (LBP) and Local Phase Quantization (LPQ) are used as texture descriptors by Betrolini *et al.* in [7]. He *et al.* in [8] and Du *et al.* in [9] used wavelet based textural features. Grey Level Co-occurrence Matrix based textural features have been used by Said *et al.*, Zhu *et al.* and Hanusiak *et al.* in [10], [11] and [12] respectively.

1.3.2 Structure-based Approach

This approach tries to capture the shape geometry and structural features of the charecters in the hand written document. This structural features are more useful for writer identification because authors' charecteristic features are embeded mostly into the shape geometry of the allographs. Most of the people used contour based structural features. Different edge based features including edge direction distribution, edge-hinge distribution were proposed by Bulacu *et al.* in [13]. Maaten *et al.* introduced multiscale edge-hinge feature in [14]. Grid Micro Structure was proposed by Li *et al.* [15]. Normalized and resampled contours of connected components was used by Schomaker *et al.* and Ghiasi *et al.* in [16] and [17] respectively. Siddiqi *et al.* implemented a window policy to divide the whole hand writing image into several parts and then used code book based features for writer identifcation [18].

Most of the above approaches have used contour based information which can be hugely affected by aspect ratio and slanting of the hand writing. Those approcaches also ignored the word level features. To overcome these issues, Wu *et al.* used SIFT features[19] in [6]. This application has been built up mostly based on the idea presented in [6] though it uses SURF [20] instead of SIFT features and ELM Classifier [21] instead of Manhatan and Chi-Square distance as dissimilarity measure.

1.4 SURF: Speed Up Robust Features

Speed Up Robust Feature (SURF) is a local feature detector and descriptor proposed by Bay *et al.* in [20]. It is partially inspired by Scale Invarient Feature Transform (SIFT) proposed by Lowe in [19]. SURF detectors detects local unique key points of an image, and SURF descriptor describes those points with set of features. Just like SIFT, SURF is also scale and rotation invarient. Only difference between SURF and SIFT is SURF uses integral box filter approximation instead of Difference of Gaussian (DoG) as used by SIFT and this makes SURF several time faster than SIFT. This SURF technique has find its wide range of application in the domain of computer vision, starting from Object Recognition, Object Tracking, Object Localization etc. Here in my project, I have used this technique to identify and describe charecteristic key points in the word regions of the hand writtent documents.

1.5 Problem definition

Off-line text-independent writer identification from hand writing images has a wide area of application. Here I have to build a GUI application which takes a handwriting image sample as an input and predicts its author id from a pre-registered set of authors.

Chapter 2

Software Requirements Specification

2.1 System Requirements

Prerequisite software and hard ware requirements are as follows:

2.1.1 Software Requirements

- *Operating System*: Windows 7 with service pack installed, Windows 8 and above, Windows 10 recomended.
- *Host software*: Matlab, version 2014Ra is recomended.

2.1.2 Hardware Requirements

- *Processor*:Any Intel or AMD x86-64 processor, AVX2 instruction set support is recommended.
- *RAM*: 2 GB Minimum, 4 GB recomended.
- *Disk Space*: Atleast 5 GB, this also depends on the number of the enrolled author i.e. the size of the database.

2.2 Functional Requirements

2.2.1 R1: Admin Login

- *Input*: Admin id and password.
- *Output*: On successful Login code book generation and author enrollment functionalities will be enabled. On wrong input a pop-up message box will come to notify the error.

2.2.2 R2: Code Book Generation

- *Input*: Path to the training image folder.
- *Output*: Code Book table of size 300 X 64
- *Processing*: It takes some time. During this time a message will be shown in the display areay of the GUI that "Code Book is Being Generated". On successful generation of the code book it will display "Code Book is Ready".
- *Constraint*: To access this functionality an Admin must be logged in to this software with valid Admin id and password. Any violation of these constraints will result a popup dialogue box to notify the error.

2.2.3 R3: Enroll Author

- *Input*: Image sample and known author id. Image sample file will be selected with a *browse button*. Author id has to be typed in a text box.
- *Output*: Author will be enrolled in the feature database and GUI display will show "Author Enrolled". This event will turn the state of the data base from clean to dirty.

- *Processing*: During enrollment process "Author is being Enrolled" message will be shwon in the GUI display section.

- *Constraint*: To access this functionality an Admin must be logged in to this software with valid Admin id and password. Also a valid code book needs to be present to enroll a author. Any violation of these constraints will result a popup dialogue box to notify the error.

2.2.4 R4: Train System

- *Input*: User need not to provide any explicit input. System itself will take dirty feature database as an implecit input.

- *Output*: On succesful training a messgage will be displayed in the GUI display area that "System is trained in t seconds", where t denotes the training time. Successful training will change the state of the database from dirty to clean.

- *Processing*: During training process a message will be shown in the GUI display section that "System is being trained".

- *Constraint*: To access this functionality an Admin must be logged in to this software with valid Admin id and password. Also the state of the database is to be dirty (modified), i.e. no redundant training is possible. Any violation of these constraints will result a popup dialogue box to notify the error.

2.2.5 R5: Identify Author

- *Input*: New hand writing image sample. This is to be selected using a Browse button. Code book, Author Database and trained classifier model will be taken as implicit input.

- *Output*: Predicted author id will be shown in the GUI display area.

- *Processing*: During the processing of the sample image, "Processing Input for Verification" message will be shown in the GUI display area.

- *Constraint*: User need not be loggedin as admin, but code book, author database and trained classifier model has to be present. Also the author database is to be in clean state. Any violation of these constraints will result a popup dialogue box to notify the error.

2.3 Non-Functional Requirements

Among non-functional requirements (i) Performance (ii) Safety and (iii) Security requirements are most important for this project.

2.3.1 Performance Requirements

- *Event Log*: Each activity with the system is to be logged in a log file. Log file should keep atleast past 24 hours log records.

- *Scalability*: The system should be scalable and atleast it should support 1000 records in the author database. There should not be any significant difference in response time for the functionalities (especially enroll and Identify functionalities) with the increase in the number of authors in the database.

2.3.2 Safety Requirements

Code book and Author database are two most important data file used in this project and so these two files needs utmost safety precotions while handeling. System should keep time-to-time back up of this two files so that in case of any system failure these files can be recovered with minimum, if not zero, data loss.

2.3.3 Security Requirements

There are two major security issues involved here: (i) Regarding Public availability and access of the database and (ii) Regarding enrollment of the new author

- *Regarding Public availability and access of the database* The author hand written samples and the feature database should not be available and accissable by public especially this system is to be used in forensic analysis.

- *Regarding Enrollment of new author* Enrollment of an Image sample with an intentionally wrong author id is a big security issue and this should not happen. To ensure this and to maintain accountability, enroll author functionality should be accessible only to logged in admin users.

Chapter 3

Analysis and Design Document

3.1 Overall Framework

The project has five modules in a broad sense. In the following subsections those are discussed in brief. Fig. ?? overall frame work of the project is shown.

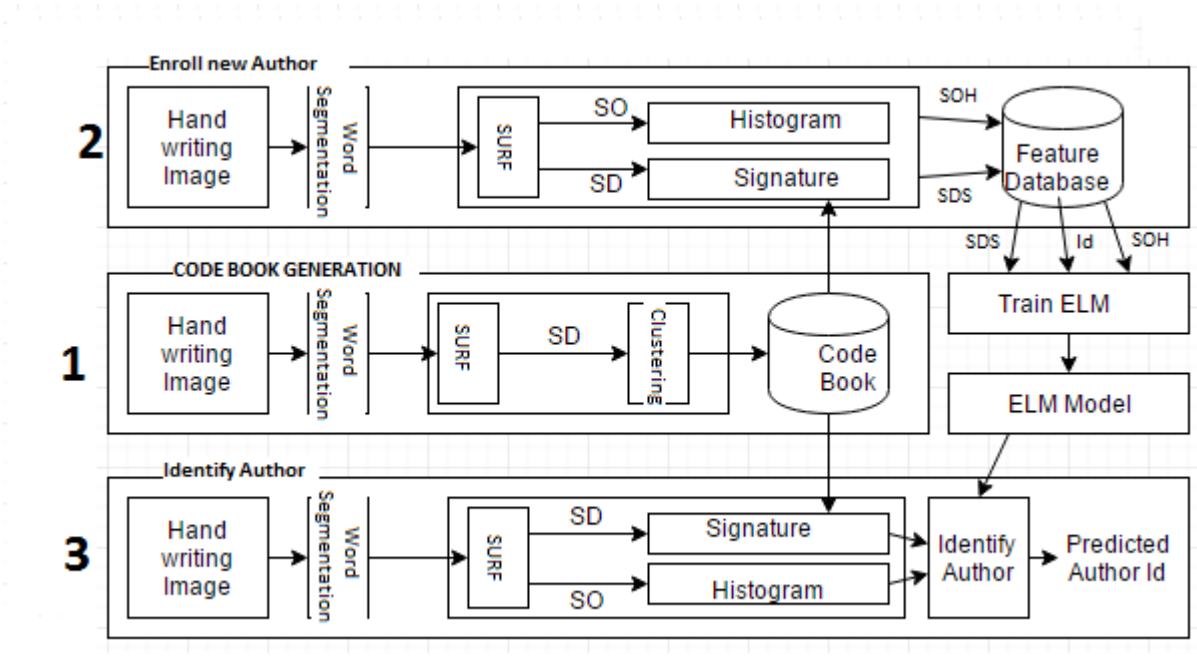


Figure 3.1: Overall Framework of the System

3.1.1 Admin Login

As the name suggests this module is for Admin log in with valid admin id and password. This module is the gateway for accessing restricted features of the system like *Code Book Generation*, *Enroll Author* and *System Train*

3.1.2 Code Book Generation

This module is to generate SURF feature code book for future reference by the feature extraction process. Only a logged in admin has access to this module.

3.1.3 Enroll Author

This module allows one to enroll a new image sample to the database with the known author id. Invocation of this module needs presence of code book as a prerequisite. On each new enrollment this module alter the state of the database to dirty. Only a logged in admin has access to this module.

3.1.4 Train System

This module needs to be invoked after new author enrollment to alter the state of the feature database from dirty to clean. This module can only be invoked if the state of the feature database is dirty. Only a logged in admin has access to this module.

3.1.5 Identify Author

This is the basic module of the system and can be accessed by any user. But, this module invocation needs code book to be present, feature database is to be present in clean state.

3.2 Three Layer Design

The project follows three layer design some what similar to MVC design. The layers, from outer most to inner most in the user perspective, are: (i) UI Layer (ii) UI Callback and (iii) Core function layer. This three layer architecture is shown in fig. 3.2

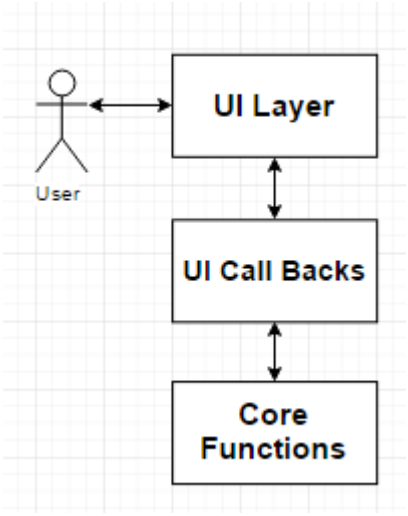


Figure 3.2: Three Layer Architecture

3.2.1 UI Layer

This is the outer most layer of the system. This is nothing but the user interface through which user interacts with the system. This is the gate way through which user can send user input to the core function layer under the control of UI Callback layer and also can get system response from this layer. This UI Layer is similar to *View layer* of MVC design paradigm.

3.2.2 UI Call Back Layer

This layer, in some sense, is similar to *Controller layer* of MVC design. User interaction with the core function layer through the UI Layer is controlled by this middle layer.

Events in the UI layer invokes function from this layer.

3.2.3 Core Function Layer

This is the inner most layer of the system. All the image processing, feature extraction and classification algorithms used in this project, are defined here. It is quite easy to visualize that this layer is quite similar to *Model layer* of MVC design.

3.3 Function Oriented Design

This project follows a function oriented approach. All three layers as discussed above consists of functions and data is being passed all around these functions.

3.3.1 DFD

Functionalities have been decomposed upto level 3 and the corresponding DFDs are shown here.

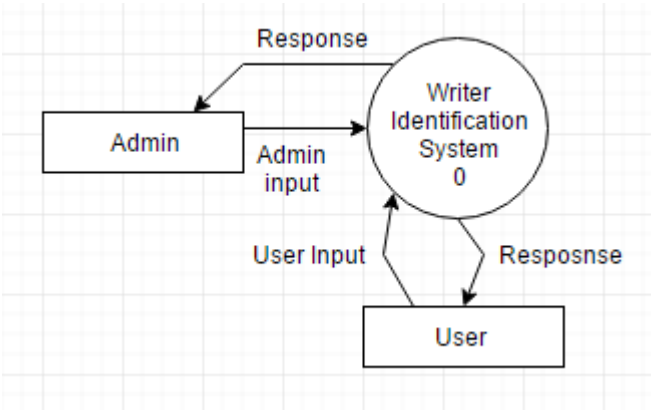


Figure 3.3: Level 0 DFD

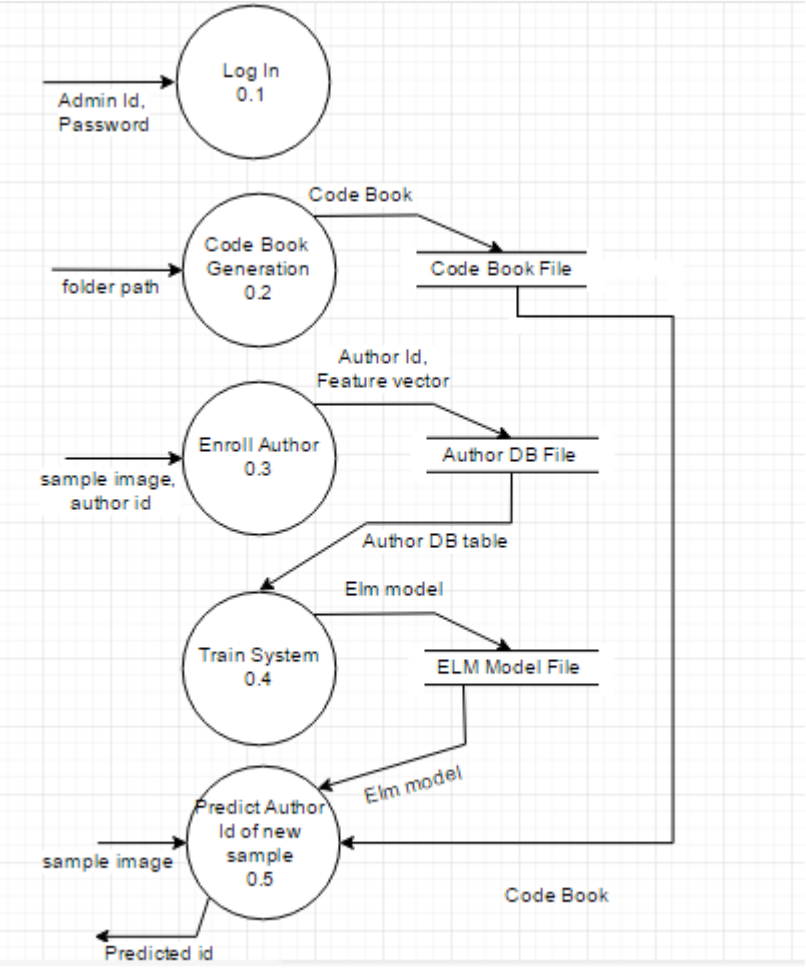


Figure 3.4: Level 1 DFD

3.4 UML Design

As the project is mostly function oriented, in UML design only the use case, sequence and state diagrams are designed and shown here.

3.4.1 Use Case Diagram

This diagram of UML shows the view of the software from the user perspective. Fig 3.7 describes the use cases of this project.

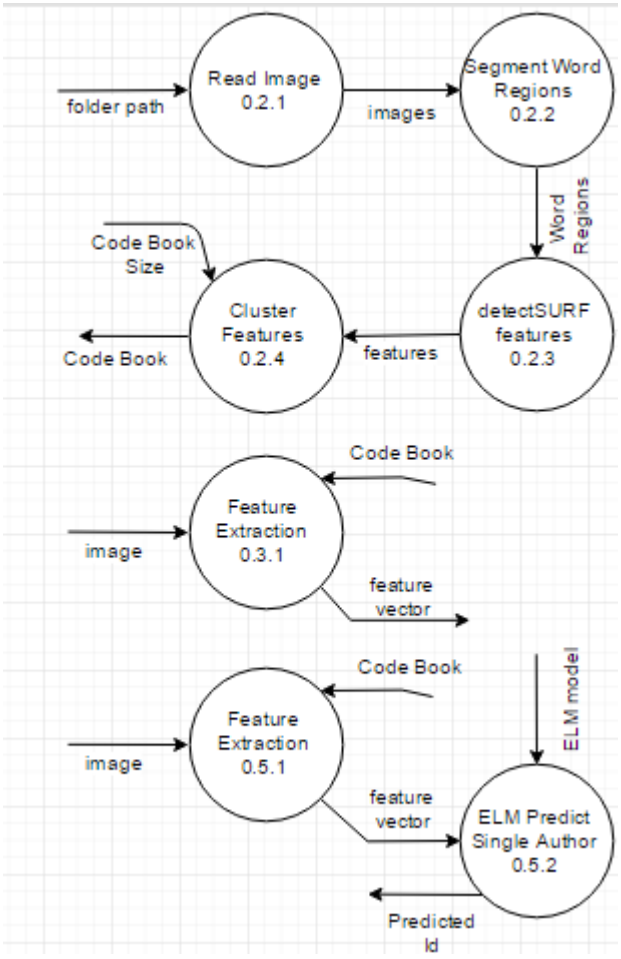


Figure 3.5: Level 2 DFD

3.4.2 Sequence Diagrams

Sequence diagrams shows the timing sequences of steps involved in each use case. So the number of Sequence Diagrams should be the same as the number of the use cases. Fig. 3.8, 3.9, 3.10, 3.11 and 3.12 describes the sequence fo steps to be done for the five use cases as shown in use case diagram (fig. 3.7).

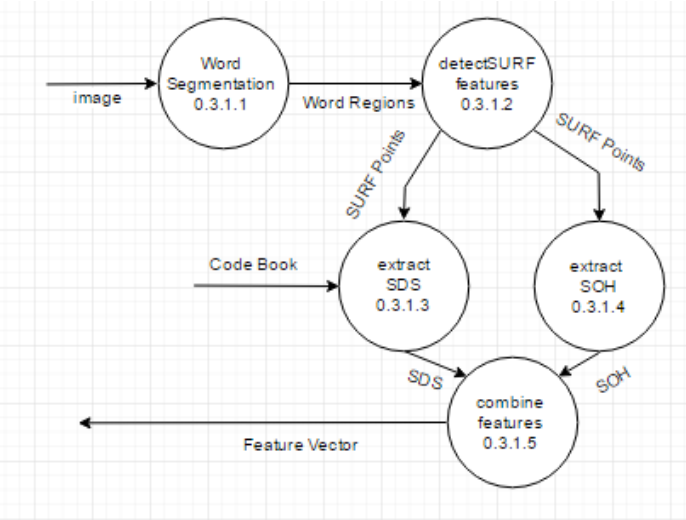


Figure 3.6: Level 3 DFD

3.4.3 State Diagram

State diagram shows the transition between different states of the system based on different activities on the system. In this project System State based on login status and feature database state are maintained.

System State Based on login status

This keeps tracks whether the system is logged in with an Admin id. The involved states are *Logged Out state* and *Logged In state*. In *Logged Out state* functionalities of code book generation, enrolling new author and train system are disabled. Only the author identification is enabled. In *Logged In state* all the functionalities are enabled. State transition is shown in the FSM model in fig. 3.13.

Feature Database State

This keeps tracks whether the feature database is updated but system not trained with updated features. Two states of the feature database are *Clean State* and *Dirty State*. *Clean State* refers to the state where system is trained with the updated feature database.

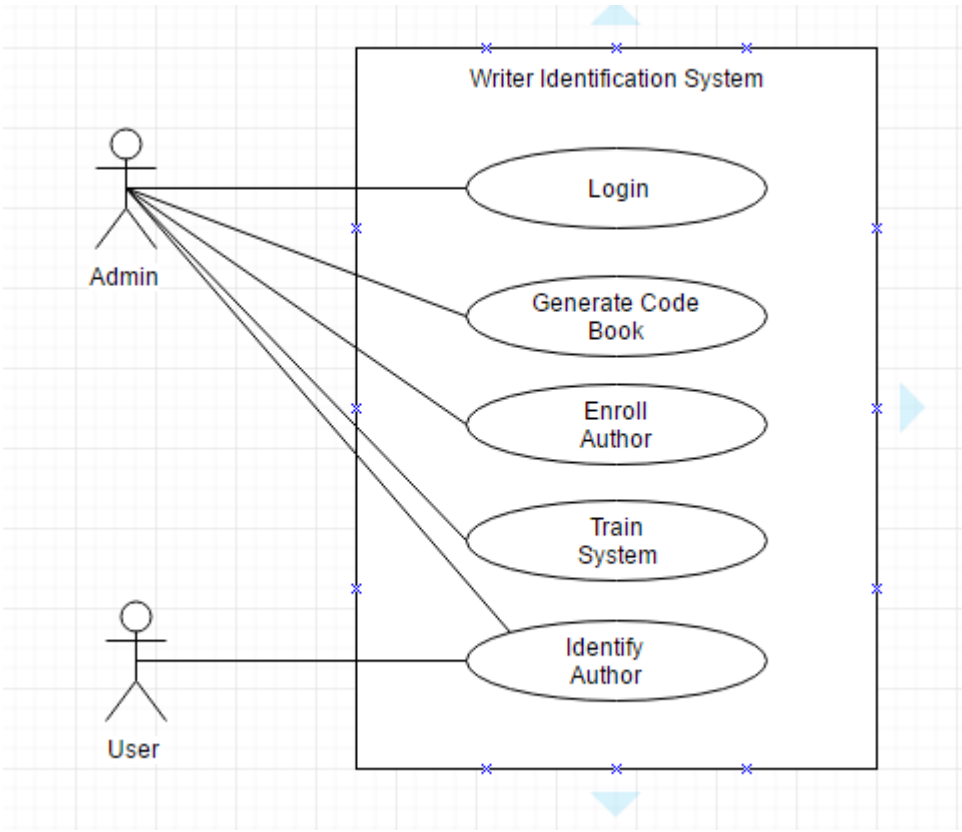


Figure 3.7: Use Case Diagram

Dirty State refers to the state where the database is updated with newly enrolled authors but system is not trained with the latest updated version of the feature database. State transition is shown in the FSM model in fig. 3.14.

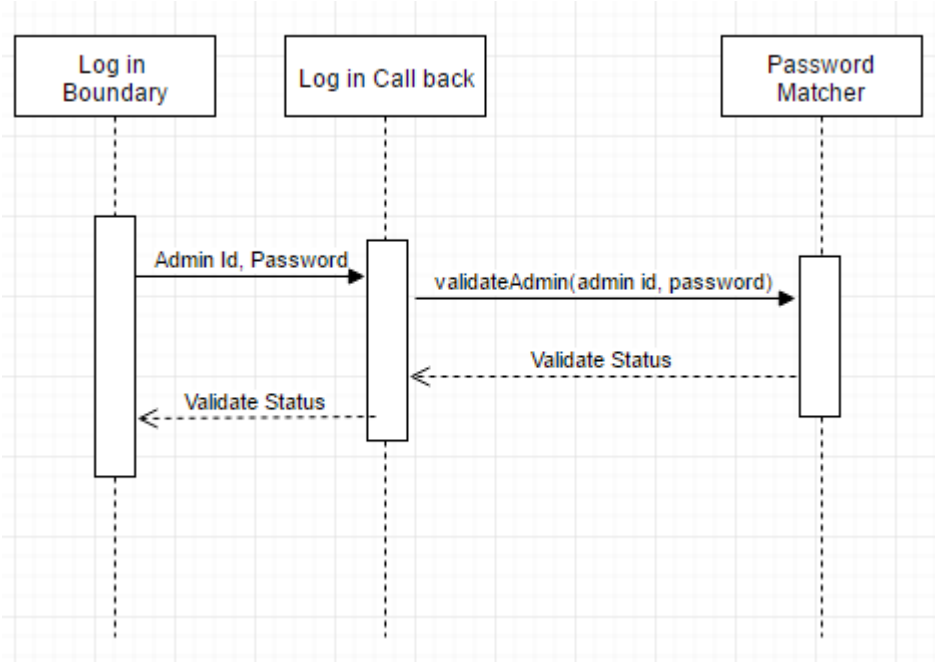


Figure 3.8: Sequence Diagram for Admin Log in

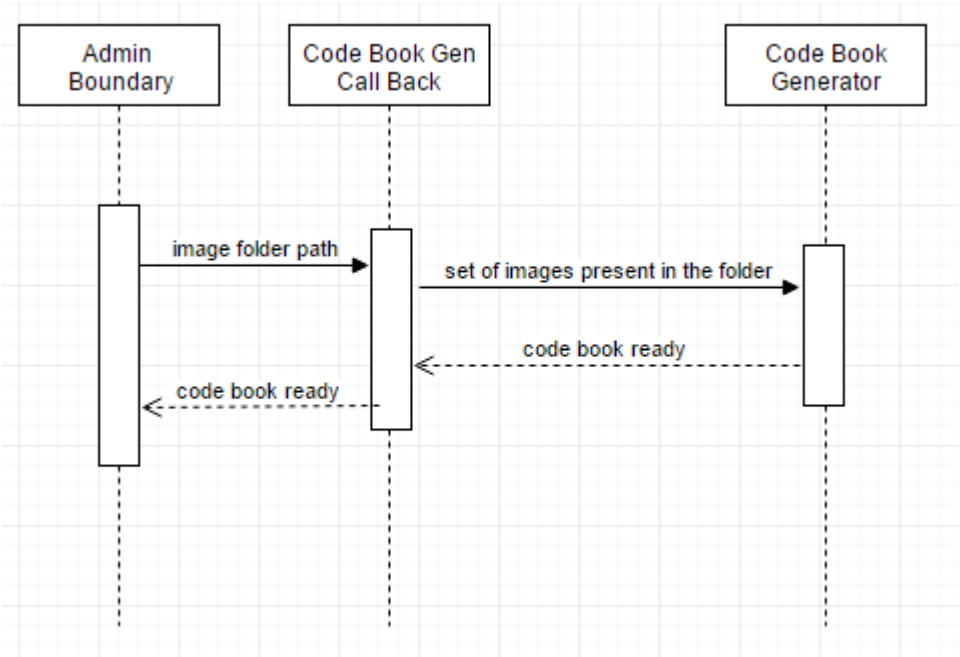


Figure 3.9: Secuence Diagram for Code Book Generation

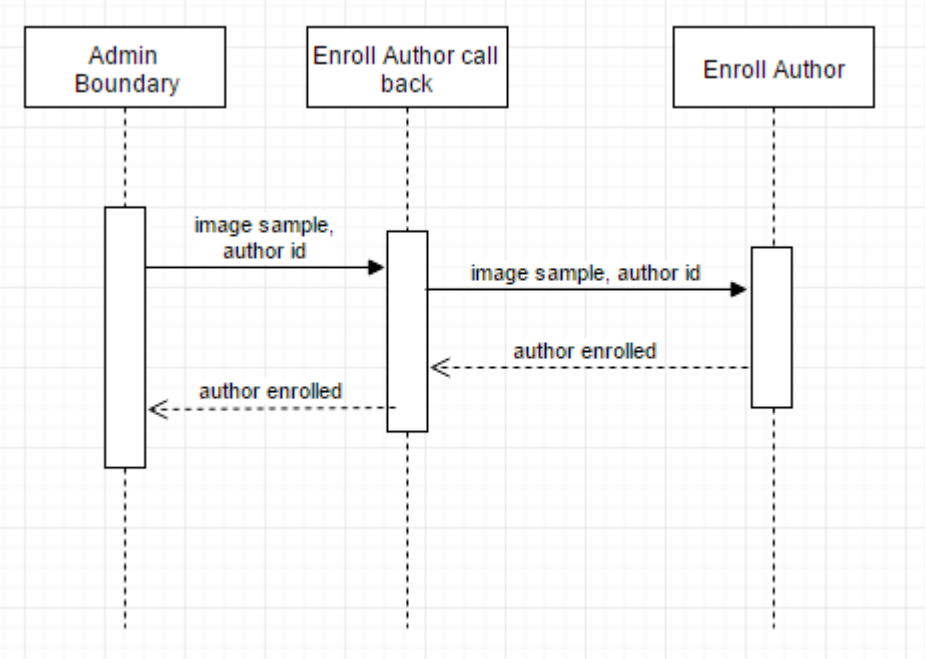


Figure 3.10: Sequence Diagram for Enroll New Author

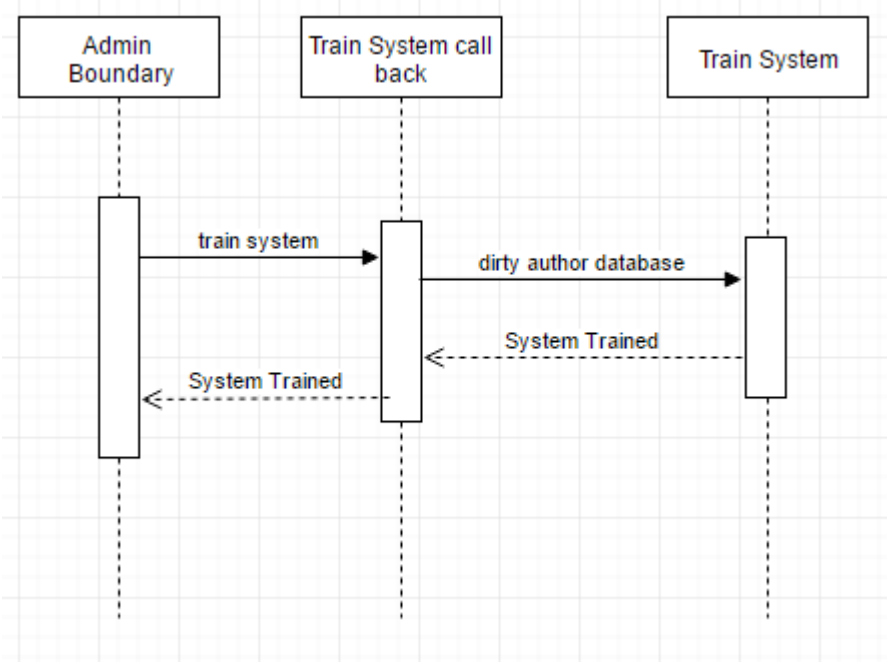


Figure 3.11: Sequence Diagram for Train System

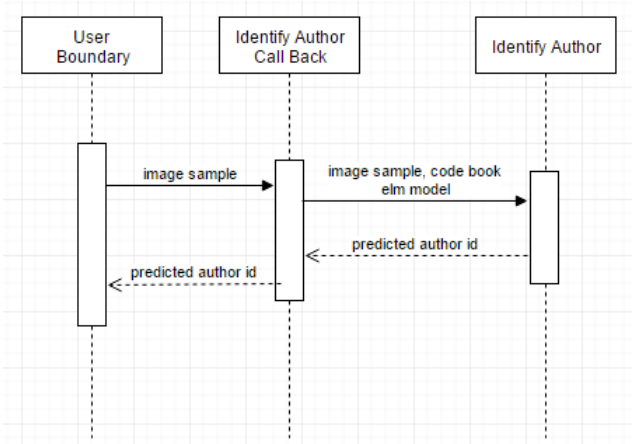


Figure 3.12: Sequence Diagram for Identify Author

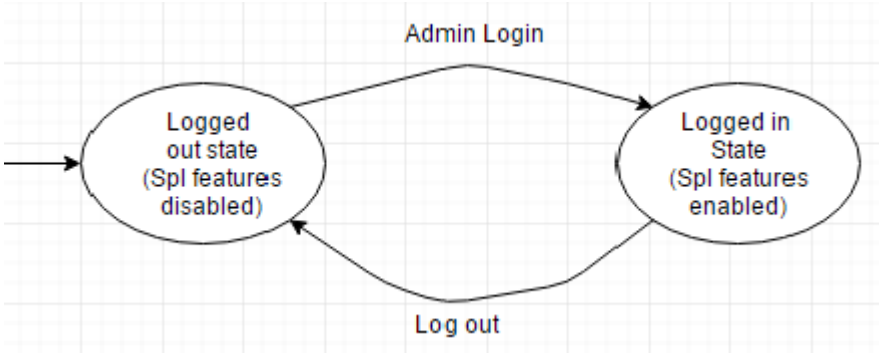


Figure 3.13: System states based on Log in Status

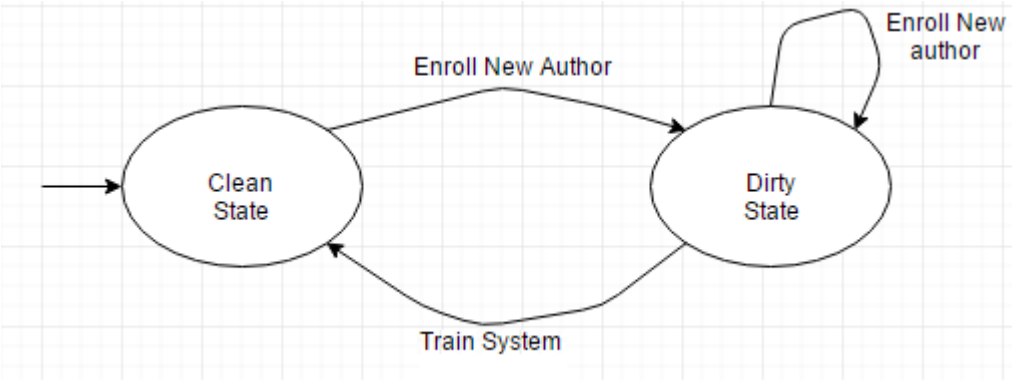


Figure 3.14: Feature Database States

Chapter 4

Implementation

In this chapter algorithimic details of the project will be discussed in implementation aspect. This will also describe the GUI with the help of snapshots.

4.1 Procedure in brief

Among different core functionalities of this project, (i) Word Segmentation (ii) Code Book Generation (iii) Feature Extraction and (iv) Author Id Prediction are most important ones and need in depth discussion in implementation point of view.

4.1.1 Word Segmentation

This is used to cut out each word regions for further processing for feature extraction. Wu *et al.* in [6] showed in case of off-line writer identification, word-level features perform better than that of page-level or allograph-level features. This is intutive because page-level feature extraction might produce huge redundant features to confuse the system, where as allograph-level features might miss some key feature with in a whole word. So word segmentation is used here before further processing.

A flow chart explaining the word segmentation process is shown in the fig. 4.2

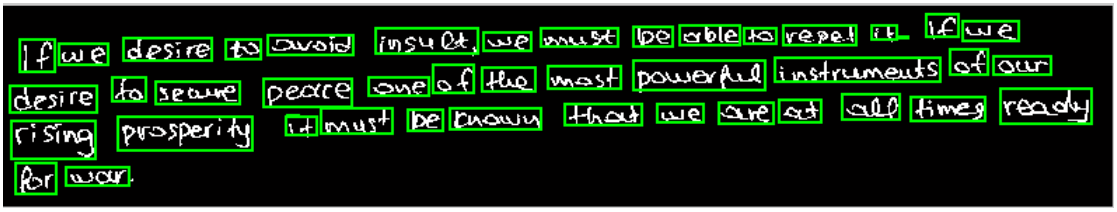


Figure 4.1: Word Segmentation Result

Gausian and Motion Blurs are used to connect the components with in a word region. Morphological opening and closing are done to eliminate very thin connections or dissjointsness. Finally bounding boxes of each connected components are considered as the word regions. Result of word segmentation is shown in fig. 4.1

4.1.2 Code Book Generation

Code book serves the purpose of a reference for histogram based feature extraction. SURF descriptor valuses of all word regions of all training images were clustered into 300 catagories using k-means clustering. This code book approach makes the Surf Descriptor feature length fixed.

4.1.3 Feature Extraction

This project is mostly inspired by the work of Wu *et al.* in [6]. Like there approach I have also used similar kind of two features used there. But, to speed up the process I have used SURF Key points instead of SIFT key points. The features used are: (i) SURF Descriptor Signature (SDS) and (ii) Scale and Orientation Histogram (SOH)

SDS Feature

This is calculated using the SURF descriptor values and the Code Book. For each of the 300 catagories of the code book 300 bins are created and an SURF Descriptor histogram is obtained which is called as SURF Descriptor Signature or SDS. Deatil algorithim is

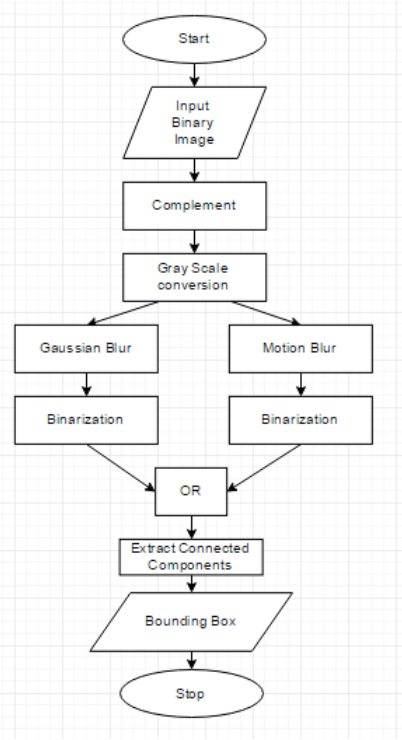


Figure 4.2: Word Segmentation Flow Chart

explained in [6]. Though they have used SIFT Descriptors, the process is same.

SOH Feature

This is calculated by taking the orientation histogram in 24 bins of 15 degree interval each of the SURF Key points across different scales and concatinating those histograms

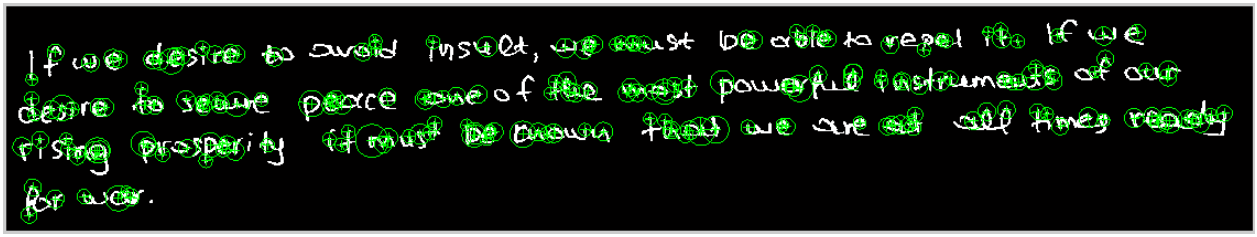


Figure 4.3: Word Level SURF features

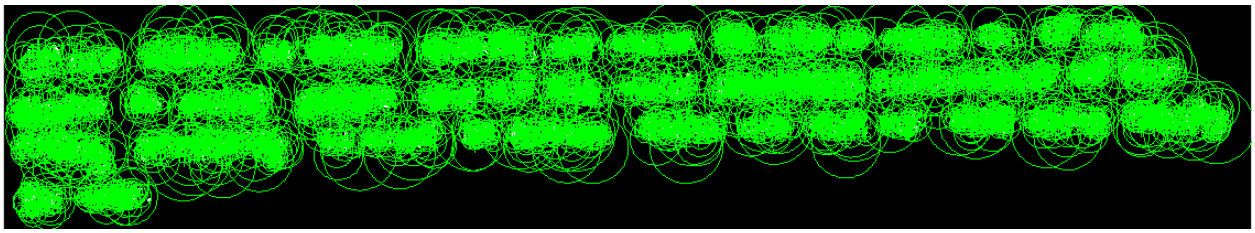


Figure 4.4: Too many redundnt SURF features at Page level

horizontally to produce the larger histogram which is called Scale and Orientation Histogram or SOH. Detail process is described in [6] in terms of SIFT key points.

4.1.4 Author Id Prediction

Unlike, Wu *et al.*[6] here classifier approach is used for predicting new author id. Each individual author is considered as a class. A relatively new classifier ELM [21], [22] is trained with the samples with known author id and a trained ELM model is obtained. During prediction of author id of the new sample, the feature vector and trained elm model are passed to the prediction function. This function, with the help of the trained model, classifies the new sample to one of the pre-registered authors and thus predicts author id.

4.2 GUI

User Interface is built with Matlab GUI Development Environment (GUIDE). It has four pannels (for log in, code book generation, enroll new author and train and Identify Author) and one display area. Outputs are shown in the display area. Success notification and error messages are shown to user in small dialogue boxes.

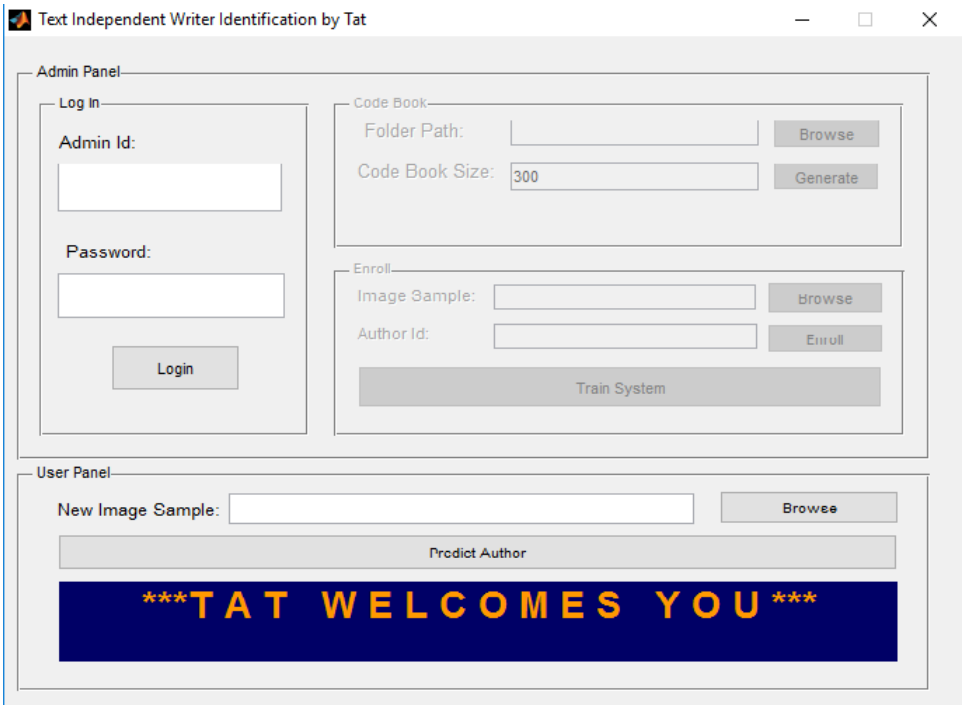


Figure 4.5: GUI: Initial Window

4.3 Hardware and software used

- **Hardware:**

  Processor: Intel Core i5, 2.20GHz

  RAM: 8GB DDR3

  HDD: 1TB

- **Software:**

  Matlab 2014Ra

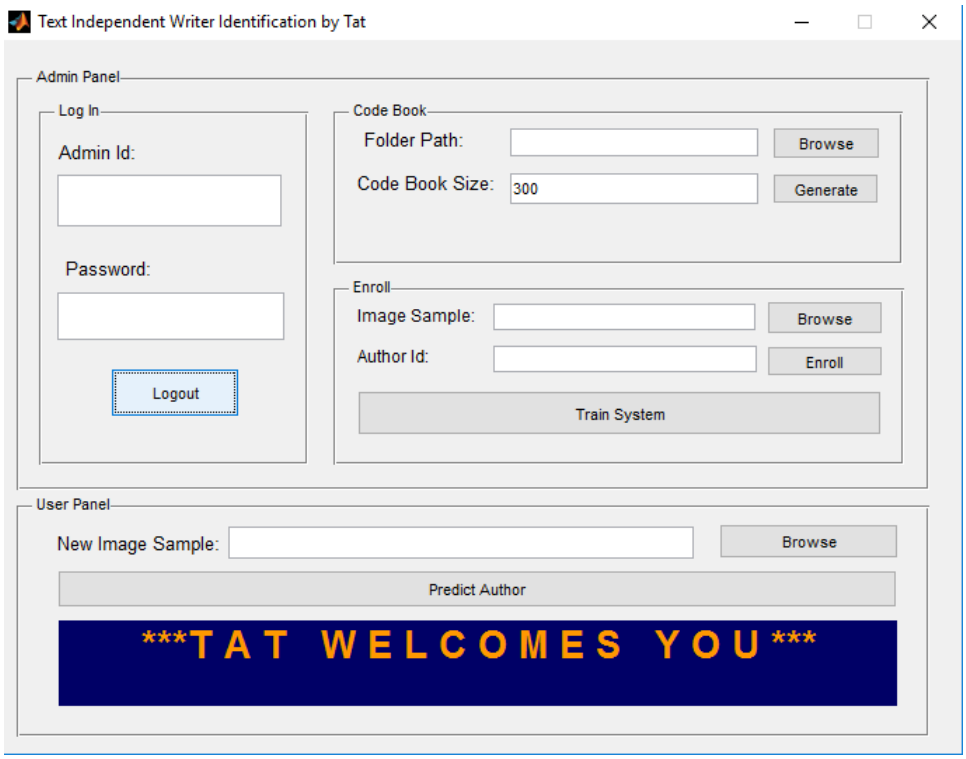


Figure 4.6: GUI: Window after Login

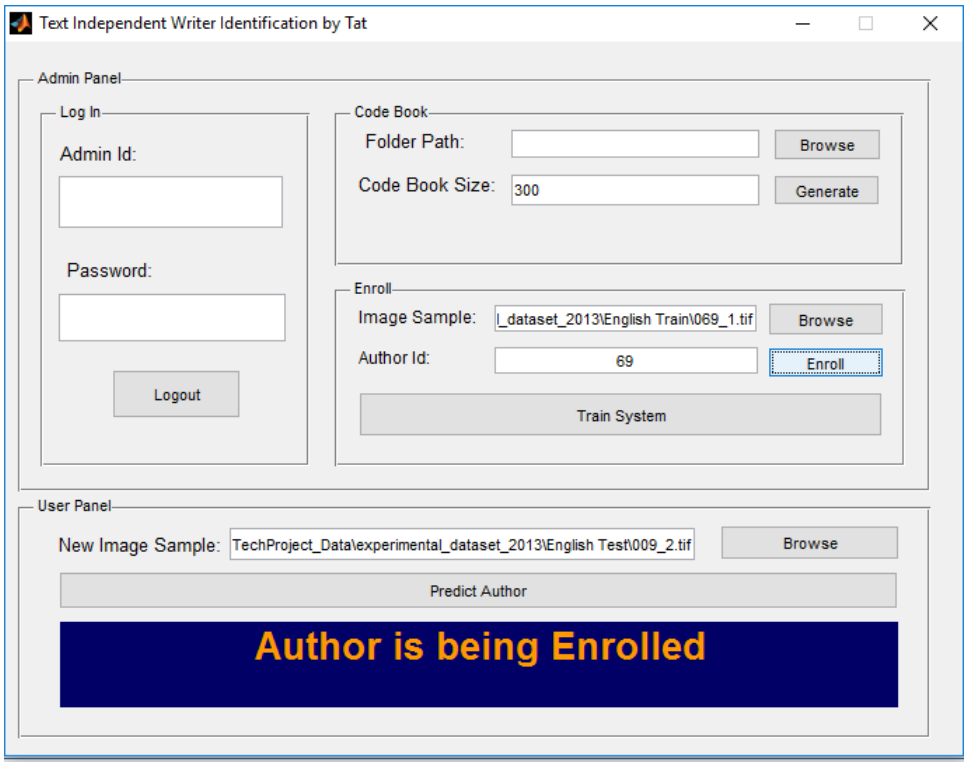


Figure 4.7: GUI: Window showing author is being enrolled

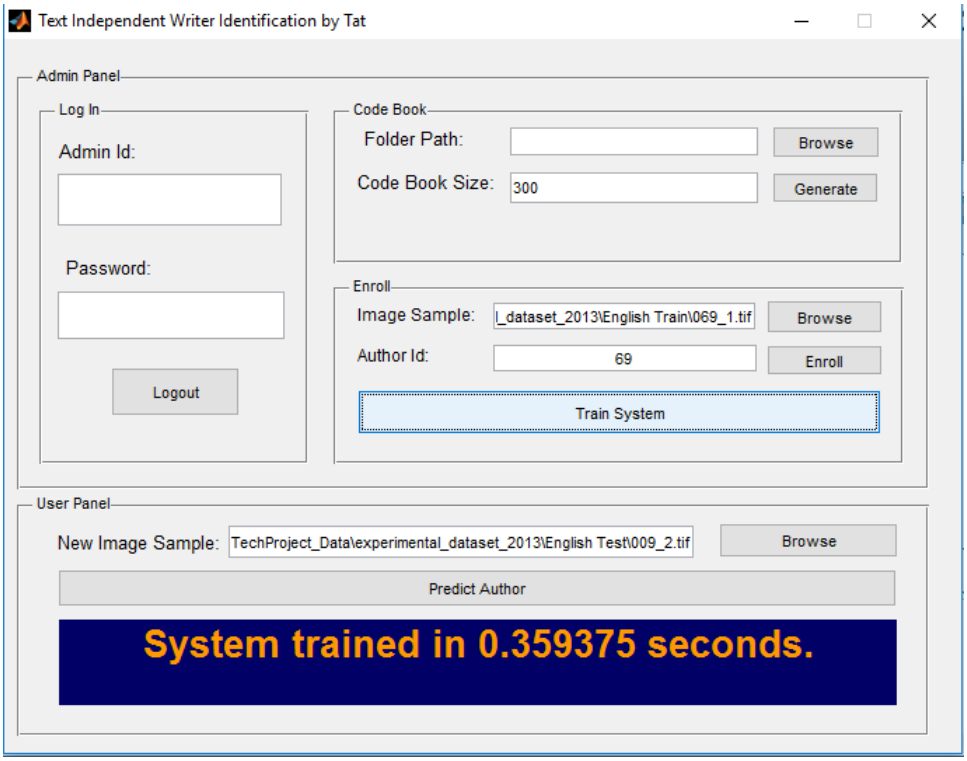


Figure 4.8: GUI: Window after System is trained. Display area showes training time.

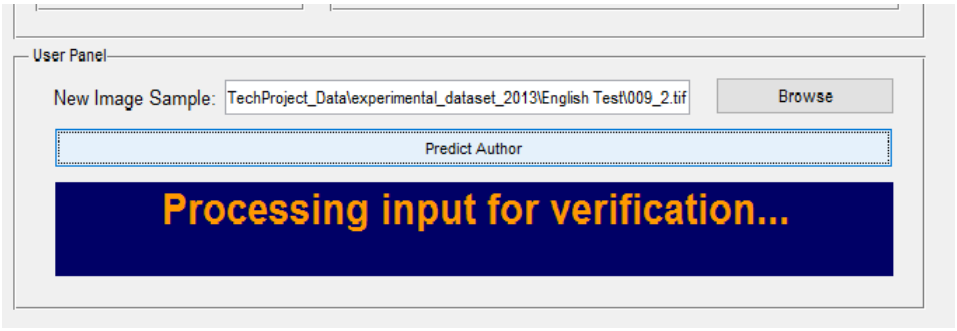


Figure 4.9: GUI: Input is being processed to serve the Predict Author request

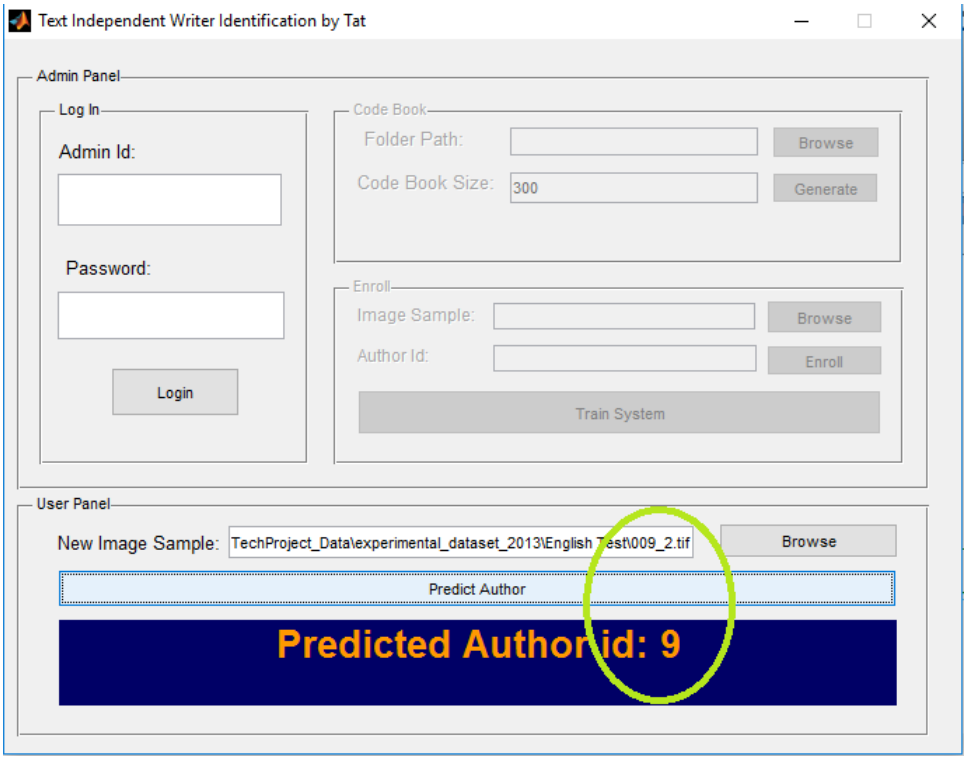


Figure 4.10: GUI: Window Showing Correct Prediction

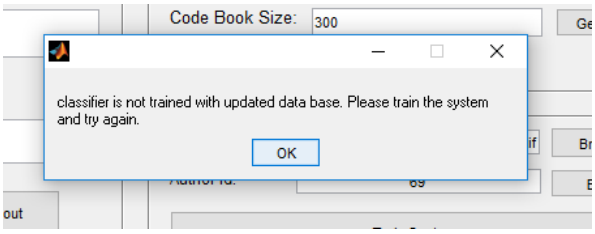
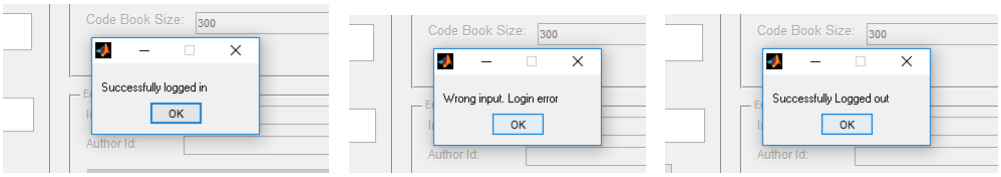


Figure 4.11: GUI: Dirty Database Error. This error message is popped up if the author database is updated but system not trained with the updated database and a predict request comes.



(a) Login Success Message (b) Login Error Message (c) Logout Message

Figure 4.12: Different Notifications and Error Messages popped up in Dialogue Box

Chapter 5

Conclusion and Future Scope

The working model of this project meets most of the requirements specified in SRS. System has reached a steady state as most of the bugs have been removed. Use of SURF and ELM speeds up the identification process as compared to other approaches of writer identification.

This project is a generic structure of a writer identification system. Many more functionalities can be added to this syetem based on specific requirements. For, instance, it can be linked with criminal dabases for use in forensic analysis.

However, this project is the working prototype version of the whole idea. So some of the functionalities need more detail implementations. Password encryption techniques to be adopted to hide the password even from the source code. Register New Admin functionality is to be implemented. SURF is patented under US Patents. So comercialization of this product needs paid permission form the patent owner. Currently this prototype has been developed using Matlab, but for public use of the software, it is better to build it with open source technologies.

Bibliography

[1] Haizhou Li, Liyuan Li, and Kar-Ann Toh. *Advanced topics in biometrics*. World Scientific, 2012.

[2] James Wayman, Anil Jain, Davide Maltoni, and Dario Maio. An introduction to biometric authentication systems. *Biometric Systems*, pages 1–20, 2005.

[3] Ameur Bensefia, Thierry Paquet, and Laurent Heutte. A writer identification and verification system. *Pattern Recognition Letters*, 26(13):2080–2092, 2005.

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

[5] Rajiv Jain and David Doermann. Offline writer identification using k-adjacent segments. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 769–773. IEEE, 2011.

[6] Xiangqian Wu, Youbao Tang, and Wei Bu. Offline text-independent writer identification based on scale invariant feature transform. *IEEE Transactions on Information Forensics and Security*, 9(3):526–536, 2014.

[7] Diego Bertolini, Luiz S Oliveira, E Justino, and Robert Sabourin. Texture-based descriptors for writer identification and verification. *Expert Systems with Applications*, 40(6):2069–2080, 2013.

[8] Zhenyu He, Xinge You, and Yuan Yan Tang. Writer identification using global wavelet-based features. *Neurocomputing*, 71(10):1832–1841, 2008.

[9] Liang Du, Xinge You, Huihui Xu, Zhifan Gao, and Yuanyan Tang. Wavelet domain local binary pattern features for writer identification. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 3691–3694. IEEE, 2010.

[10] Huwida ES Said, Tienniu N Tan, and Keith D Baker. Personal identification based on handwriting. *Pattern Recognition*, 33(1):149–160, 2000.

[11] Yong Zhu, Tieniu Tan, and Yunhong Wang. Biometric personal identification based on handwriting. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 2, pages 797–800. IEEE, 2000.

[12] Regiane Kowalek Hanusiak, Luiz S Oliveira, E Justino, and Robert Sabourin. Writer verification using texture-based features. *International Journal on Document Analysis and Recognition*, pages 1–14, 2012.

[13] Marius Bulacu and Lambert Schomaker. Text-independent writer identification and verification using textural and allographic features. *IEEE transactions on pattern analysis and machine intelligence*, 29(4), 2007.

[14] Laurens Van Der Maaten and Eric O Postma. Improving automatic writer identification. In *BNAIC*, pages 260–266, 2005.

[15] Xin Li and Xiaoqing Ding. Writer identification of chinese handwriting using grid microstructure feature. *Advances in Biometrics*, pages 1230–1239, 2009.

[16] Lambert Schomaker and Marius Bulacu. Automatic writer identification using connected-component contours and edge-based features of uppercase western script. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):787–798, 2004.

[17] Golnaz Ghiasi and Reza Safabakhsh. Offline text-independent writer identification using codebook and efficient code extraction methods. *Image and Vision Computing*, 31(5):379–391, 2013.

[18] Imran Siddiqi and Nicole Vincent. Text independent writer recognition using redundant writing patterns with contour-based orientation and curvature features. *Pattern Recognition*, 43(11):3853–3865, 2010.

[19] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[20] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[21] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1):489–501, 2006.

[22] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: a new learning scheme of feedforward neural networks. In *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, volume 2, pages 985–990. IEEE, 2004.