OFFLINE MODE AUTOMATED SIGNATURE VERIFICATION

USMAN AMJAD

03-243201-012

A thesis submitted in the fulfilment of the
requirements for the award of the degree of
Masters of Science in Computer Science

Department of Computer Science

BAHRIA UNIVERSITY LAHORE CAMPUS

OCTOBER 2022

Approval of Examination

Scholar's Name:	Usman Amjad	Registration No:	67332

Program of Study: Master of Science in Computer Science

Thesis Title: Offline Mode Automated Signature Verification

It is to certify that the above student's thesis has been completed to my satisfaction and, to my belief, its standard is appropriate for submission for Evaluation. I have also conducted plagiarism test of this thesis using HEC prescribed software and found similarity index at ____4%_____ that is within the permissible limit set by the HEC for the MS/MPhil degree thesis. I have also found the thesis in a format recognized by the BU for the MS/MPhil thesis.

Principal Supervisor's Signature: _____

Name: Dr. Iram Noreen

Date:15th November, 2022

Author's Declaration

I, Usman Amjad hereby state that my MS thesis titled "Offline Mode Automated Signature Verification" is my own work and has not been submitted previously by me for taking any degree from Bahria University Lahore Campus or anywhere else in the country/world.

At any time if my statement is found to be incorrect even after my graduation, the University has the right to withdraw/cancel my MS degree.

Name of student: Usman Amjad
Date: 15th November, 2022.

Plagiarism Undertaking

I, solemnly declare that research work presented in the thesis titled “Offline Mode Automated Signature Verification” is solely my research work with no significant contribution from any other person. Small contribution / help wherever taken has been duly acknowledged and that complete thesis has been written by me.

I understand the zero-tolerance policy of the HEC and Bahria University towards plagiarism. Therefore, I as an Author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/ cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of MS degree, the university reserves the right to withdraw / revoke my MS degree and that HEC and the University has the right to publish my name on the HEC / University website on which names of students are placed who submitted plagiarized thesis.

Student / Author’s Sign: _____

Name of the Student: _____Usman Amjad_____

DEDICATION

To

Prophet Muhammad (PBUH)

My Loving Mother (Late)

Wife, Sister, Brothers

&

Worthy Father

Amjad Ali

Who taught me the first word I speak

The first alphabet I write

And

First step I walked

May ALLAH give them a Long Happy Life

ACKNOWLEDGMENTS

To Almighty ALLAH I have no words to express my deepest gratitude (The Merciful and Gracious), who enabled me to make this manuscript complete. I would like to express my gratitude to my supervisor Dr. Iram Noreen Senior Assistant professor at Department of Computer Science Bahria University Lahore Campus for her guidance, continuous encouragement and untiring supervision in bringing this thesis into its present form.

I want to thank my family over the years, for their love and encouragement. Special thanks to my wife during my research for her continuous support.

I am profoundly grateful to everyone who has always like to see me smiley and cheering. May ALLAH bless them with long lives and good health and be a source of prayers for me.

ABSTRACT

Handwritten signature verification has got community attention among numerous biometric systems during this decade. It is widely used as identification and verification of a person, transaction or document in organizations, banks, law courts, business processes. Offline signatures are mere images of signatures and are mostly managed by Computer Vision techniques such as template matching, and statistical methods. Recently, Hidden Markov models and Neural Networks based approaches are used in the problem domain. However, despite wide-ranging work by research community signature verification still remains open to the research due to the diverse challenges such as intra-class variability among signature of same individual. Further, extraction of discriminative visual features is another challenge while using machine learning approaches. Deep learning approaches are not highly explored in this domain due to non-availability of large datasets due to privacy restrictions. This study aims to propose a latent deep learning-based approach for automatic signature verification to be used on mobiles and other less resourceful devices In our approach we have explored an adhoc MobileNet-V2 to learn weights using offline triplet loss. SVM, Random Forest, MLP and Adaboost classifiers are used to generate result, the best being Random Forest on Bengali handwritten signature dataset. I am able to achieve 86% accuracy score with skilled forgeries.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CNN	-	Convolutional Neural Network
DL	-	Deep Learning
DTW	-	Dynamic Time Warping
HOG	-	Histogram of Gradient
HMM	-	Hidden Markov Model
LSTM	-	Long Short-Term Memory
NN	-	Neural Network
RNN	-	Recurrent Neural Network
OfSV	-	Offline Signature Verification
EER	-	Equal Error Rate
WI	-	Writer Independent
WD	-	Writer Dependent

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

Signature Verification has gained popularity for the last few decades as the number of populations grows the need to identify and verify a person on the basis of presented document also grows. Xamxidin et al. [1] has described it as the innate habit of a person evolved with the passage of time which makes it very difficult to imitate or detect. Financial institutions, Forensics, Real Estate Industry and other organization consider signature verification as the primary source of person and document verification. Lopes et al. [2] describe signatures as a powerful tool for identification of a person against a responsibility he is assigned or performed. Hafemann et al. [3] considered OfSV as a behavioural biometric characteristic of a person to verify person and documents. Although there are other parameters and measures as well like thumb, palm etc. Generally, the reason behind signature forgery is to pose the identity of a person or document as genuine, and such mischievous act is also kept secret from original person till the achievement of the nefarious motive.

Bibi et al. [4] categorized biometrics in two major groups as defined by as Physiological and Behavioural. The prior is related to physical features of a person like thumb, palm, finger, iris, retina etc and later is related to behavioural features which may get affected by age, mood, time, muscle strength and other related elements. Unlike face recognition where similar faces have less or no similarity, signature of a same person may tend to vary each time. Generally, signature verification is considered part of biometric information systems and categorized in two parts known as Online (dynamic) and Offline (static). Radhika et al. [5] further discussed data acquisition mechanism for Static (offline) data refers to the images of signatures which are acquired using an A4 size page on which the users put their signatures. The images are created using camera or scanners. The users

are required to sign with a pen whereas the dynamic (online) is obtained through stylus and finger using pressure sensitive devices (tablets, smart phones etc) and also WEBCAMs placed at specific angels to capture the video while performing signatures. Offline signatures have static features which pertains length, height, slant, baseline, pressure, size and coordinates of x and y axis. Similarly, dynamic signatures have additional dynamic properties apart from its shape and size. These properties include velocity, acceleration, sequence of strokes, pressure, stylus tip direction of signatures. There is also a limitation in signature verification task as the signatures vary depending upon conditions, time and environment. Chugh et al. [6] shed light on further challenges like an author's signatures changes depending upon its age, physical and psychological conditions. Similarly, Tolosana et al. [7] illustrates the technological advancements with the passage of time has boosted the data acquisition techniques and deep learning progression has further cushioned the verification tasks.

OSV is categorized in two main approaches namely Writer Dependent (WD) and Writer Independent (WI) [8]. In prior case the model is learned specific to a writer and in later the model is learned independent of the writer. Now some authors also explored hybrid approach as well, [9],[10] have proposed a hybrid approach in which the model is learned in WI way and the classification is performed through WD approach. There are some pros and cons of both approaches; WD is more complex as a separate model is trained for each author and inclined to overfitting as signature data is usually small in size, on the contrary WI overcome the signature count problem but many writers related features may not get enlightened.

Offline signature verification is considered a relatively inspiring task due to scarcity of features as compared to online signature verification. Usually, the state-of-the-art approaches leverage algorithmic or manual feature extraction. Feature's extraction is done applying statistical approaches, pattern recognition is one of them and is used with a fixed size vector. These vectors are further applied to extract local and global features. Local features include Gaussian features, binary patterns, gradient histograms and contours [11]. Whereas global features comprise Fourier transformations, projections, shape related features, directions etc [12],[13]. There are some other methods as well including distance measure among signatures [14], shape features [15], Hidden Markov Model [16], Contourlet [17], wavelet [18], curvelet [19] and Radon [20] transform

features. Nevertheless, deep learning has shifted the feature extraction from hand-crafted features to automatic feature extraction straight from the images with Convolutional Neural Network (CNN) [21],[22]. This research is based upon Offline Signature Verification due to the fact that most of the financial transactions are still being performed on papers.

1.1 Stages of Signature Verification

Signature verification task inherits the steps followed in hand written text images related research and methodologies. These fields are very closely corelated and some researchers have resort to hand written text images datasets (often named as auxiliary data) for training deep networks. Although signature verification is complex and require more computational resources. The general criteria of offline signature verification are depicted in Figure: 1.1. These steps are usually followed in hand-crafted feature extraction.

1.1.1 Data Acquisition

Data Acquisition is the basic ingredient of signature verification for either Online or Offline mode. For Offline data, users are required to sign with a drawing or ball pens with usually black ink on a specific sized paper within a single session or multiple sessions. These signatures containing papers are then scanned using cameras or scanners, transformed into grayscale with a suitable resolution before saving into the databases. These images have PNG, JPEG and JPG. These datasets consist of genuine, random forgeries, simple forgeries and/or skilled forged signatures for training and testing. In online case the data is acquired using smart phones and special devices that may capture dynamic features like speed, acceleration, force, position. The features that are captured using online method are rich as compared to offline. This is why the offline signature verification is complex in comparison with online and hence with rich features online has better accuracy [4].

1.1.2 Pre-Processing

Pre-processing of acquired data is very critical. It is performed to enhance and prepare it for the feature extraction step. This process revolves around removing blurriness, noise, reducing background complexity and heighten foreground. The general procedure involved are depicted by [5] in Figure: 1.2.

1.1.3 Feature Extraction

Feature extraction phase is the pivotal step in getting better performance and accuracy. This step is categorized into two main categories (i) Manual Features (ii) Model Based or Automatic Feature Extraction. In the former type the features in offline signature verification would be the slant, baseline, size, pressure etc. These features are further divided in Global and Local. Global features transpire size of the signatures, overall orientation of the signatures etc. whereas Local features deals with a specific location feature in overall signature image like pixel values at a specific place. Model based or Automatic feature selection is made by the model applied. Convolutional Neural Network (CNN) or Recurrent Neural Network (RNN) based model learn features during their execution.

1.1.4 Feature Selection

Some researchers have also applied feature selection module before classification in manual feature extraction perspective. This is usually applied when researcher try to extract global or local or both features and later try to amalgamate the results with some merging techniques. The researchers try to select the most prominent features among all for better classification. Feature selection is generally performed through any sort of scoring mechanism. Automatic feature extraction techniques avoid feature selection.

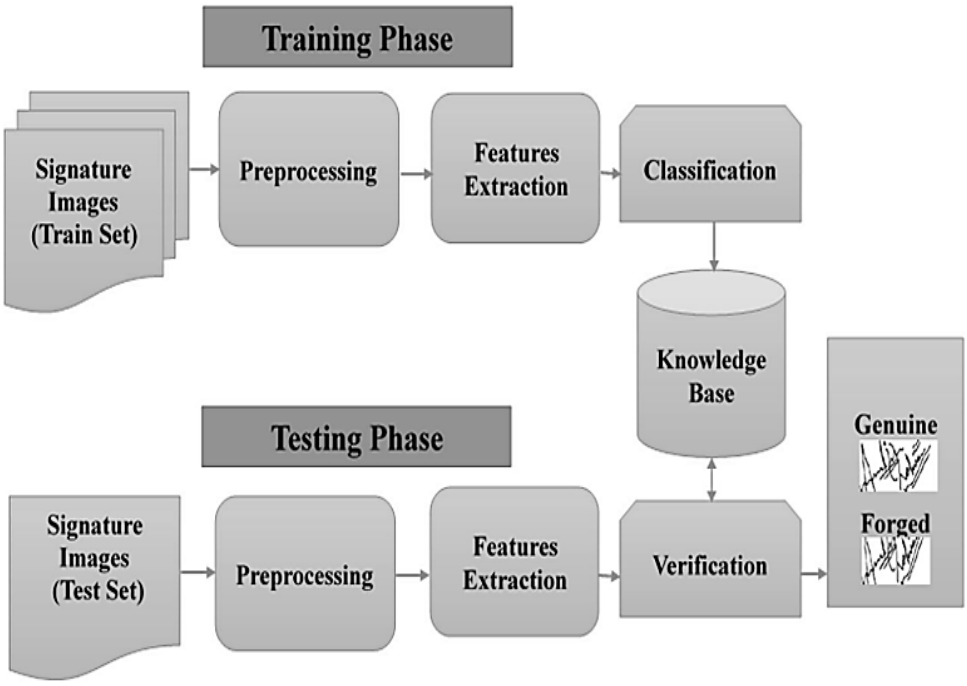


Figure 1.1: Signature verification stages [1]

1.1.5 Classification

The classification or verification process authenticates the query signature on the basis of reference signature knowledge base which is usually learned by the model applied on a large dataset as training dataset. A binary classifier is used which identifies the query signature as either genuine or forged. Apart from data acquisition hurdles and the advancements in a data acquisition techniques and technologies, most of the signature verification applications and systems uses simple machine learning algorithms and approaches such as Support Vector Machine (SVM), Dynamic Time Warping (DTW), Hidden Markov Models (HMM) and Neural Networks. Although other biometric features such as face and finger prints have achieved quite significant results through deep learning as compared to old and tradition techniques. Signature verification is very similar to hand writing recognition in many ways as both are behavioural biometric.

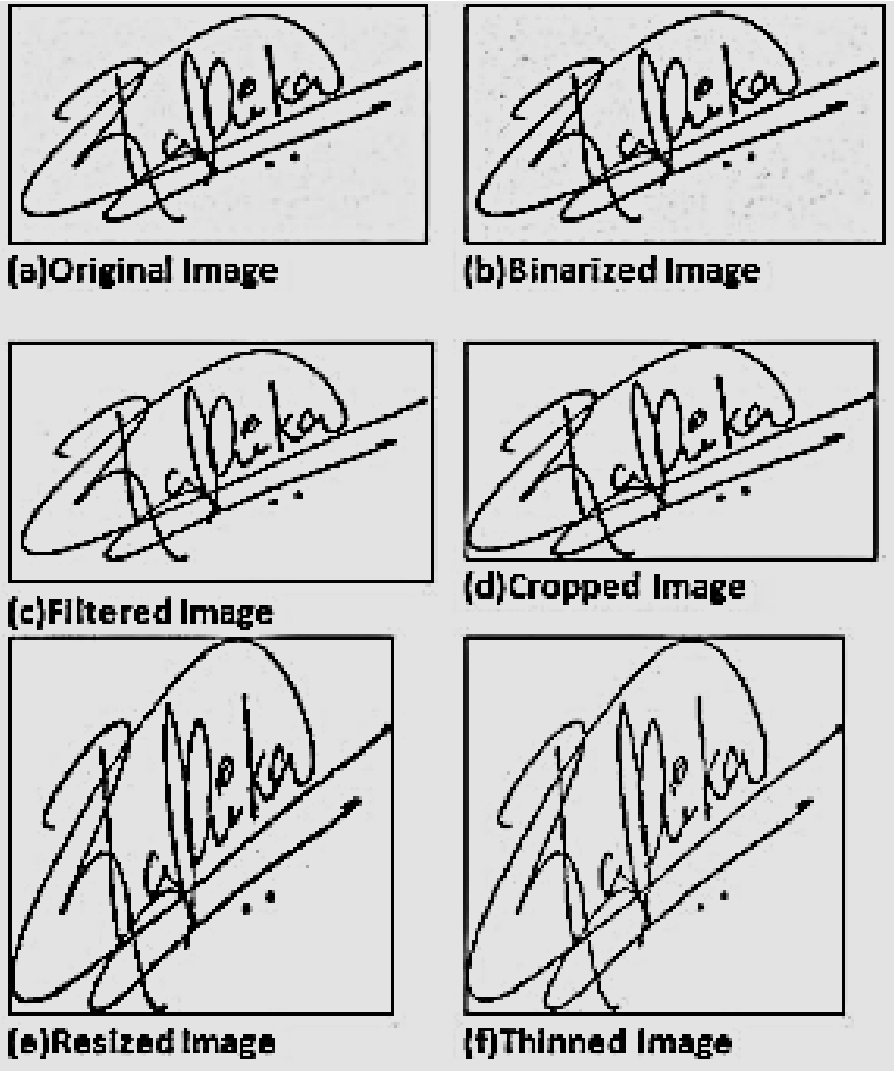


Figure 1.2: Stages of OfSV pre-processing [23]

1.2 Significance of the Research

Signature verification is a very common issue in financial, forensics and real estate organizations as legal documents and cheques are frequently been verified for forgery detection and manual verification is a gigantic task if the quantity of legal and financial instruments become large. This gives rise to automatic signature verification in recent times. Nevertheless, a lot of work has previously been done on this field using traditional measures. State of the art deep learning techniques are yet to be explored on

more sophisticated grounds. This research is typically based upon application of deep learning techniques in this field.

1.3 Motivation

Signatures of a person is considered as behavioural biometrics. Signatures ensures the sanctity of a content of the document as well as the physical presence of a person. The documents signed by authorized person has a legal value. Signature forgery not only ruin the purity of the document but be deemed as identity theft of the person whom signatures are being forged. Manual signature forgery detection is done by signature experts since the inception handwritten documents but it took time and effort to detect forgery and also the margin of error was very high.

Thus, this gives rise to automatic signature verification in recent times. Nevertheless, a lot of work has previously been done on this field using traditional measures. State of the art deep and transfer learning techniques are yet to be explored on more sophisticated grounds. This research is typically based upon application of deep and transfer learning techniques in this field.

1.4 Research Gap

Though research has been performed in signature forgery detection domain however, potential of deep learning and its allied field of transfer learning approaches is not explored for offline signature verifications and forgery detection in signatures. The main problem addressed in this research is that signature attributes related rich information is lost during signature generation process in offline mode mainly due to intra-class variability and other elements like resolution of device, heat, stylus and finger variation, and sampling rate variation etc. Moreover, currently discriminative visual features extraction is mostly performed using hand crafted and manual methods. Trainable feature set for neural networks is also kept high.

1.5 Problem Statement

Signature forgery is a crime and the person committing such thing definitely has dark motives in forging signatures. Automated forgery detection is complex because the signature depends upon an individual's age, gender, muscle strength, mood, writing style, posture and pen. In addition, skilled forgeries are almost impossible to figure out with traditional methods. Despite the fact a huge number of research work has been done on this but deep learning approaches are not widely explored in comparison to statistical, model whereas structural approaches which have performance related issues in verification of signatures. This research is primarily focused towards investigation of deep learning model to address the issue of information loss during offline signature generation and automated discriminative visual features extraction with highest performance and accuracy.

1.6 The Aims and Objectives

Aim of this study is to investigate automatic static signature verification using deep learning paradigm with reduced computational dependency.

- To identify core difference among genuine, random and skilled signature forgeries.
- To explore existing datasets and models.
- To improve accuracy and performance in signature verification system.
- To investigate the best approach, Writer Dependent (WD), Writer Independent (WI) or Hybrid.

1.7 Key Research Question

- How features of offline signature verification be extracted?

- What existing techniques have been used to identify signature forgery?
- How a neural network can improve the accuracy of signature verification?
- Which datasets are being employed for the tasks?
- How transfer learning aspect being applied for OfSV?

1.8 Main Contributions

The main contribution of the thesis can be summarized as follows:

- Identification of forgery among genuine and forged embeddings of signatures.
- Construction of a novel deep learning approach with fewer trainable parameters in order to construct a mobile based application which require smaller amount of computational resources.

1.9 Thesis Methodology

The layout of our research is based upon the steps listed in Figure 1.3. The literature is reviewed in a way to figure out the latest trends and approaches in signature verification task. Identification of the problem and solution proposed by various researchers. Which datasets are used and publicly available or need to build own dataset. Exploration for evaluation criteria and the metrices used for results comparison are also made part of this research endeavour.

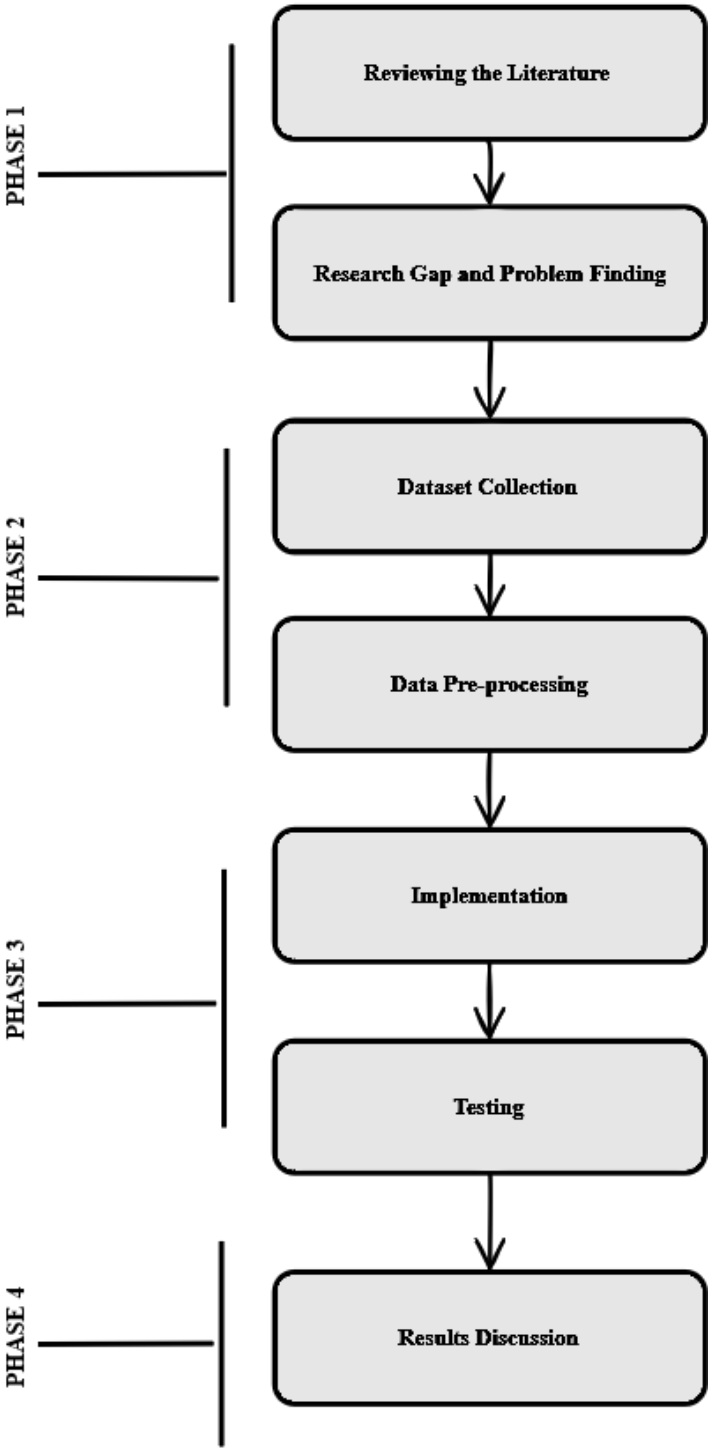


Figure 1.3: Research methodology

CHAPTER 2

LITERATURE REVIEW

Currently, massive research has already been done in this area and it is still a very fertile area for future endeavours. This literature review is caried out in a way to cover all the aspects of offline Signature verification. For such, search and keyword criterion are defined and shown in Table 1.1. Initially starting with a general term “Offline Signature” or “Handwritten Signature” and then adding additional terms to make it specific search sentence has really paid off in filtration of research articles. Resultantly, more than 50 research terms were used for initial categorisation of the article. Further filtration was carried out on the basis of journals/conferences, year, duplication etc.

The general disparity among these approaches is difference in learning models and methodology. This section is distributed in 4 categories. 1) Common datasets used in research, 2) Computer Vision approaches, 3) Deep Learning methods, and 4) Transfer Learning. Discussion regarding the approaches used by researchers is categorised in each section.

Bibi et al. [4] explored the Online and offline signature verification techniques in depth. Authors have provided with a survey of different verification techniques applied by various authors. The paper is illustrating the terms related to behavioural biometrics and further elaborates the offline verification models as well as online verification models and the results achieved by those researchers. The author also shed light on publicly available datasets for both online and offline.

Table 2.1: Searching Criteria

Bi-Gram	Append	Append	Phrases
Offline Signature	Verification, identification, classification	Machine/Deep Learning, Deep Transfer, Transfer Learning	Offline Signature Verification/identification/classification using Machine/Deep Learning/ Deep Transfer/ Transfer Learning (~12 phrases)
Handwritten Signature	Verification, identification, classification	Machine/ Deep Learning, Deep Transfer, Transfer Learning	Handwritten Signature Verification/identification/classification using Machine/Deep Learning/ Deep Transfer/ Transfer Learning (~12 phrases)
Forgery Detection	Offline Signature, Handwritten Signature	Machine/ Deep Learning, Deep Transfer, Transfer Learning	Forgery Detection in Handwritten Signature/Offline Signature Verification/identification/classification using Machine/Deep Learning/ Deep Transfer/ Transfer Learning (~22 phrases)

Online Signature Verification is quite different from Offline Signature Verification. The set of significant features distinguish the both approaches. Radhika et al. [5] provided a hybrid approach for the two signature acquisition techniques i.e., online and offline. Data is collected from 13 different writers, which consists of 390 genuine and 325 forged signatures. Online signature data is acquired using a webcam placed at left side to a right-handed writer in order to capture the pen tip and similarly for left-handed writer camera was placed at right side. Captured video data is used as online data and images of signatures are used as offline data. The author used similar steps i.e., data acquisition, pre-processing, feature extraction and classification.

The noise is removed from the video data and scanned images using various methods for feature extraction. For online pen tracks are analysed and for offline gradient and projection features are used. Unique and most suitable features are extracted. For online data dynamic time warping (DTW) algorithm is used for classification and Euclidian Distance is used in offline case. Lastly, the author used Support Vector Machines (SVM) for combined classification. This approach illustrated in this article a traditional approach and features are learned and extracted separately from model.

Tolosana et al. [24] proposed deep learning approach along with a new signatures database. He further emphasized on proposong a new database for signatures, devising a standard experimental protocol to accomplish an impartial comparison between various signature databases. Initially the author presented a view of different signature databases and data acquisition medium for each of them. He combined all the databases and named it as Deep Signs DB, which consists of 70K signatures using two different channels i.e., finger and stylus from 1526 users. The author used dynamic time warping (DTW) and Recurrent Neural Network (RNN) and named it as Time-Aligned Recurrent Neural Network (TA-RNN) presented by same author [25].

2.1 Datasets

Most of the studies pertaining to transfer learning used a handwritten text due to its similarity with the handwritten signature images. In this sort of formation, the model is trained on a handwritten text dataset owing to the fact that these datasets are rather richer in form of samples as compared to signature datasets and due to constraints of not available publicly [26],[27]. GPDS synthetic dataset [28], which consists of 4000 users signatures with 24 genuine and 30 forged signatures is till date considered the largest non-public dataset available.

However, there are other signature datasets as well which are also used for state of art results and comparisons. Most of the authors [26],[27] and [29] have used multiple signature dataset to evaluate their transfer learning results. This is due to the model being biased towards a single dataset [30], and researchers always try to generalize their results in order to prove the authenticity of their proposed models.

Generally the most common datasets used are MCYT-75 [31], CEDAR , ICDAR, UTSig, BHSig, Kaggle, GPDS-960, GPDS synthetic. Some of these datasets are not available without contract however, datasets other than English are easily available.

Although the open signature datasets are not very large except BHSig-260 but models trained on them can be generalized for other larger datasets. A brief illustration of common handwritten datasets along with their division regarding genuine and

forgeries, number of users is given in Table 2.1. The forgeries mentioned in these datasets are skilled forgeries.

Table 2.2: Various datasets used for offline signature verification

Dataset and Language	Language	Users	Distribution
GPDS-synthetic [32], 2015	English	4000	24Genuine 30-Forgery
GPDS-960 [33], 2007	English	881	24-Genuine 30-Forgery
GPDS [34], 2004	English	160	24-Genuine 30-Forgery
MYCT-75 (sub-corpus) [35], 2003	Spanish	75	15-Genuine 15-Forgery
UTSig [36], 2016	Persian	115	27-Genuine 45-Forgeries
BHSig-260 [37], 2016	Bengali-Hindi	100 160	24-Genuine 30-Forgery
Kaggle	English	30	5-Genuine 5-Forgery
CEDAR [8], 2004	English	55	24-Genuine 24-Forgery
ICDAR [38], 2011	Dutch, Japanese,	10	23-Genuine 12-Forgery

2.2 Computer Vision (CV) Approaches

According to Hameed et al. [39] the process of remodelling raw pixels into a single feature vector by generating new features from existing features and dumping the original ones is known as feature extraction. Computer Vision is based upon hand-crafted feature extraction from the images. Hand-crafted features are majorly categorized in global and local features as shown in Figure 2.1.

Although there has significant amount of research been performed in this domain. In this section various computer vision and machine learning models are discussed with respect to offline signature verification.

Sharif et al. [40] used hand-crafted feature extraction and selection along with Support Vector Machine (SVM) classifier to distinguish between genuine and forged signatures. Authors used hand-crafted feature set from signatures datasets including CEDAR, MCYT and GPDS synthetic. A novel generic algorithm (G.A) is used to reduce error rate from previous literature. Local and global features are extracted and formed as chromosome. ANN cost function is applied to get least expensive chromosomes are fed to SVM for classification. A set of 156 extracted features are then passed through a Generic Algorithm (GA) for selection of best features. The result for MCYT is 5.0, on GPDS it is 5.42 and on CEDAR it is 4.67 with 12G samples.

Batool et al. [41] used 22 Grey Level GLCM along with 8 Geometric measures using pre-processing for feature extraction. The authors also used a novel technique HFPI for fusion of features. The final features for classification is selected through PCA (SKcPCA) with skewness+kutosis approach.

Aravinda et al. [42] used HSVR for signature verification and recognition for personal authentication. Initially the image pre-processing is done by removing noise and images are converted into 16*16 size images to reduce computational overload. The sample consists of 25000 signatures consisting of actual and fake signatures out of 100 users. The sample is saved in separate folders consisting of 25 samples dividing 15 original and 10 forged signatures. All the age groups are included to add diversity. Different pens and languages are used. The results show 81.5% accuracy.

Skilled forgeries as discussed earlier are hard to detect due to its resemblance with original signatures. In skilled forgeries the forger has knowledge of everything related to the original signatory i.e., his name, signature impression and also the way the signature was done. Intraclass invariability among multiple signatures of a signatory creates more problems in detection of genuineness of the signature. The common way in machine learning technique is to divide a signature into a grid and then measure each stroke line etc. for genuine signatures in training and then a distance-based classifier is used to track the changes in reference signature and given signature. Fang et al. [43] researched on

skilled forgeries in offline signature forgery using hand-crafted feature extraction techniques. The writers are of the view that there may exist variations among signatures drawn by same individual. These variations are tracked as features and a range of authentic signatures is actualized during training and can be applied to any test signature. Two separate methods are proposed.

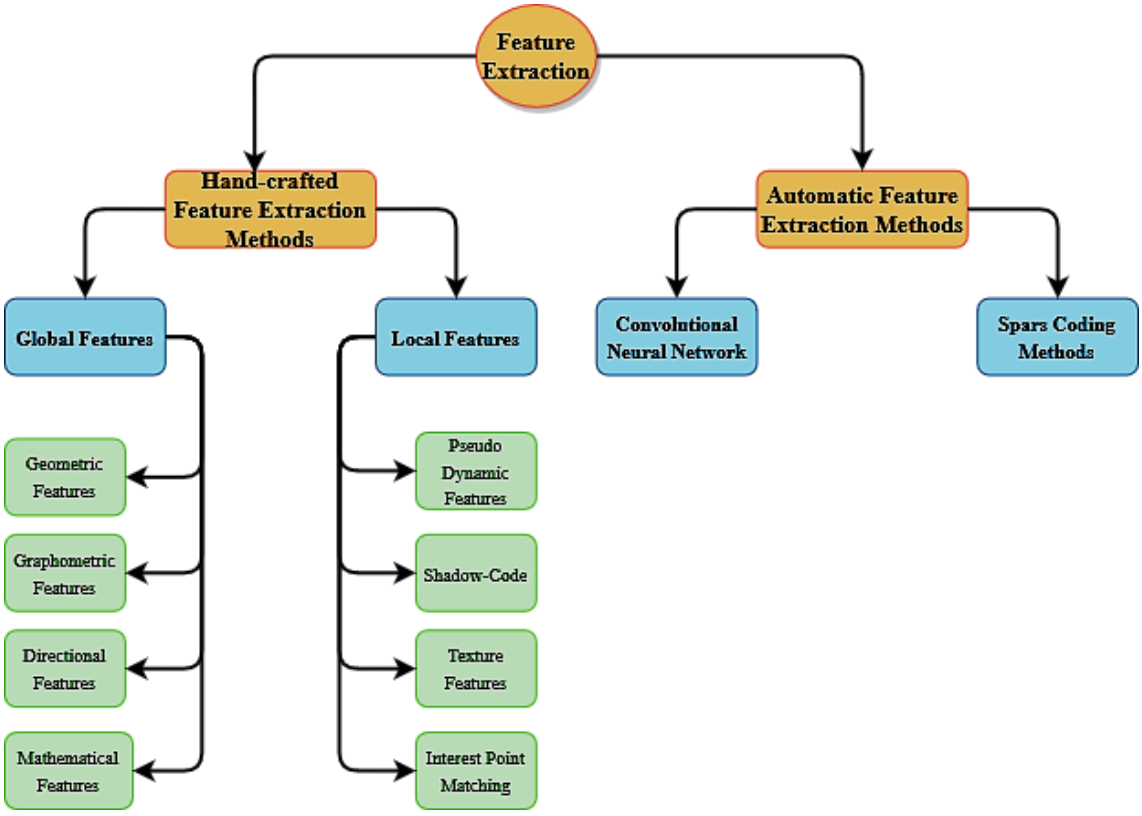


Figure 2.1: Hierarchy of feature learning techniques

In first method, warping function is applied to patterns of projection profiles (along horizontal and vertical axis) and consequently positional variations is derived from warping function. Distance measures are used to find the genuine/forged signatures having FAR on both vertical and horizontal projections 23.2% and FRR percentage 21.4% with binary signature image. Second method involves stroke segments instead of projection profiles. After pre-processing an approximated short line (element) is drawn against skeleton signatures. Method-2 achieves an FAR of 23.3 and FRR percentage of 24.3 for both and on binary image. Wen et al. [44] proposed a novel technique of RPF

framework to detect invariance among the genuine signatures of a writer. RPF model has two components ring external feature (REF) and ring internal features (RIF), but according to writer these features are not sufficient for classification. So Mahalanobis distance model and ring HMM models are used separately. The classification is marked by a threshold score. If the signature threshold remains within the settled threshold the signature is genuine. MCYT-75 dataset is used for experimentation. The results are compared with both techniques HMM and Mahalanobis distance measure and they are 15.02 and 15.3 respectively.

Yilmaz et al. [11] has applied local histogram features. The signatures are fragmented into both cartesian and polar spaces and then a dictionary is created from both coordinate systems. Authors used both WI and WD approach through SVM. User-dependant SVM is for writer identification and Global SVM is for signature verification. Local features are extracted using Histogram of Oriented Gradients (HOG) and Local Binary Patterns (LBP) and Support Vector Machines (SVMs) for WI and WD verification. These are features fall in the category of texture features.

Singh et al. [45] have created it own dataset and used machine learning algorithm for skilled forgery detection. Signatures from 48 individuals were collected and labels were created through overlapping. Pre-processing was done to create Grey Level Co-occurrence Matrix GLCM and Red, Green and Yellow (RGY) images. Later, RGY images are passed through a CNN. GLCM matrices give the textual properties of an image and provides a dictionary of co-occurrences of grey level values over an offset of an image. Classification of both feature sets is performed through SVM.

Okawa [23] proposed KAZE algorithm as local feature extractor, clustered them using K-Means algorithm and then BoVW/VLAD is used to convert these features into a single vector for classification using SVM. VLAD feature extraction would require normalization and dimensionality reduction. Although results been shown with several variations in BoVW and VLAD in terms of features count the best would be 1.6% EER for KAZE and BoVW (512) are 1% EER for VLAD with KAZE and L2 (64).

Bouamra et al. [46] has proposed one-class SVM with run-length algorithm as feature extractor. Research is caried out to learn features from only genuine class and classify skilled forgeries. Run-Length algorithm is also a textual feature extraction

technique which tracks black and white pixels of a segment of an image, hence this technique requires binary image. Further One-Class SVM discriminate between one class and all/any other class and it is also emphasised by Hafemann et al. [8] and suggested that single class SVM phenomenon is very generic to signature classification problem The proposed model achieves an accuracy of 93.23%.

Oliveira et al. [47] has proposed an approach for feature extraction based upon graphology. Author also discussed various features with sample signatures for better understanding of the features. Four statistical and 3 pseudo dynamic features are short listed for verification. To calculate these features a grid with a box size of 16X40 is overlayed and features are extracted from these segments. The dataset is created with total of 5600 signature gathered from 60 users. 60 genuine signature per author, the forgeries used are random (genuine signatures of other authors or any genuine signature which are not enrolled in dataset), simple forgeries and simulated forgeries. Author have used various HMM models for each type of features extracted. The graphometry features which are based upon statistical methods used for evaluation are Density of Pixels, distribution of pixels, slant, progression and form and the results produced are 7.87, 7.65, 7.92, 9.15 and 11.3 respectively in terms of average error on random forgeries for each of the mentioned features. Average error although not used frequently in literature as an evaluation measure but it is the mean of false negative and false positive.

These researchers have performed significantly in the field of signature verification. The features extracted mostly belong to the family of textual features. These features are usually extracted by dividing the image into fixed size grid and then hand-crafted feature extraction algorithm are applied to collect features from a grid cell and also establishing relation with neighbouring grid cells. Textual features hence proved to be the top most applied algorithm in the field of manual feature extraction, which is apparent from this section of literature review.

Table 2.3: State of art comparison for computer vision techniques

SrNo.	Author.	Dataset	Technique	Performace (EER, FAR, FRR, Accuracy)	Critical Review
1.	Sharif et al. [40], 2020	CEDAR, MCYT and GPDS synthetic	Generic Algorithm (G.A), SVM	4.67 AER, 5 AER, 5.42 AER	The method adopted is prone to noise. The pixel with non-zero value may exist outside signature regions.
2.	Batool et al. [41], 2020	GPDS synthetic, MCYT, CEDAR	GLCM, Skewness and Kurtosis, SVM	10- Folds (FRR=10.0, FAR=9.17), (FRR=2.00, FAR=2.66), (FRR=3.75, FAR=3.34)	GLCM technique falls under statistical feature extraction technique. GLCM work on the relative frequencies of neighboring pixels and hard to generalize.
3.	Aravinda et al. [42], 2019	Self-Acquired Dataset	ANN	81.5% Acc.	Author has acquired 25000 signatures, but did not mention the criteria. Hence there may exist 4. discrepancies in signature acquisition.
4.	Fang et al. [43], 2003	Self-Acquired Dataset	Non-linear dynamic warping	M1: 22.3% AER M2: 23.8% AER	DTW is mostly used in Online signature verification as it is related to temporal variations. This algorithm is used to extract the variation among signatures of an individual.
5.	Yilmaz et al. [11], 2011	GPDS-160	(HOG), (LBP), SVM	15.41% EER (12 reference signatures)	Author have used skilled forgeries and apply both statistical and textual type features but they have inbuilt drawback of lower accuracy, bias and noise.
6.	Wen et al. [44], 2009	MCYT-75	RPF, Mahalanobis distance measure, HMM	15.3 EER 15.02 EER	The technique comprised of textual and structural feature and are more application oriented. The results show promising EER but may have low accuracy.
7.	Singh et al. [45], 2021	Self-Acquired Dataset, ICDAR (Dutch)	GLCM, RGY through CNN, SVM	(DUTCH) CNN: P=0.61, R = 0.69, F1-score = 0.65 GLCM: P = 0.84, R = 0.77, F1-score = 0.80	The technique used in this paper is a combination of both hand-crafted feature extraction and CNN. The overlapping of images may occur in case of forgeries as skilled forgeries are very close to real signatures and noise and distortion may also add-in.

8.	Okawa [23], 2018	CEDAR, MCYT-75	KAZE, BoVW, VLAD SVM	1.6% EER BoVW 1% EER VLAD	BoVW require a large code book so a large computation is involved in it. Although VLAD entails less computations than BoVW but it requires some sort of dimensionality reduction to enhance its productivity.
9.	Bouamra et al. [46], 2018	GPDS-960	RunLength One Class-SVM	93.23% acc.	The model works on one class SVM which is a good way to distinguish other classes but run length algorithm is noise prone and distortion. The threshold for classification is done through another set of signatures which hinders the generalization mechanism of the technique.
10.	Oliveira et al. [47]	Self	HMM	7.87, 7.65, 7.92, 9.15 and 11.3 Avg. Error	The textual and statistical features are used for model training. Author also used pseudo dynamic features as these features are usually available in online signature verification. The features selected are inclined towards noise, distorted form and hard to generalize.

2.3 Deep Learning (DL) Approaches

Jahandad et al. [21] have used Google Net architectures of CNN known as Inception-v1 and Inception-v3. The author used keras a wrapper over TensorFlow, to tune the model and used gradual increase in training set by adding different values of users that is initially the author used 20 user's data learned the model and used call-back epoch if the accuracy stops or reduces. Then again on 30, 100 and finally 1000 user's data. The model also used call-back and Reduce Learning Rate functions for lining up the model. The author claimed EER level of 22 in Inception-v1 case and also F1-score of 0.75 with Validation Accuracy of 77% whereas the values in case of Inception-v3 EER of 26, F1-score of 0.72 and Validation accuracy of 73% is achieved.

Navid et al. [22] proposed Siamese based Convolutional Neural Network to distinguish between forged and genuine signature. Scholars set a two-point theme to build their experimental model: 1- Enhance precision for differentiation of forged signatures and 2- reduced the precision decrease for time interval. The authors used a pretrained

VGG-19 architecture to address the computational and data scarcity issues and then connected the pretrained model to 256 layers of CNN, gradually reducing to 128 layers and then to 64 layers. These 64 layers are further connected to 512 layers of output, then to 256 to 128 and then to a compact layer of 2 outputs with weights of the model made unfrozen. The dataset used is a combination of three publicly available dataset Kaggle, ICDAR and CEDAR. Images are RGB and in .PNG format. The results are quite unrealistic for ICDAR, a 100% is reached in classification but the model performed well for CEDAR at 88% test accuracy and Kaggle at 94% test accuracy.

Maergner et al. [31] also applied CNN with triplet loss function combined with structural approach. Scholar applied graph-based model to calculate Graph Edit Distance (GED) to measure cost. Author also reduces the computational complexity of GED by bipartite approximation framework. Cost of substitution, insertion and deletion is described as Euclidean distance of node labels, constant cost C_{node} and zero respectively. Each GED's calculated is normalised with maximum value. CNN is applied with distance measure between two signatures. Finally, the model is passed a triplet function with a genuine signature, a forget signature and a reference signature. The authors achieved an EER measure of 1.05 for random forgery and 9.15 for skilled forgery on MCYT dataset. On GPDS dataset the scores for above are 0.41 and 6.49 respectively.

Alajrami et al. [48] proposed CNN based approached for offline signature verifications. Pre-processing is applied to each image in the dataset. The dataset is obtained from Kaggle which consists of 300 images from 30 users (5 genuine and 5 forged). Results obtained are quite impressive but the author did not provide details of forgery types. Scholars obtained 99.9% accuracy on training with 80-20 ratio. However, validation accuracy remained 99.7%.

Yapıcı et al. [49] also applied CNN on GPDS dataset. Author used a separate technique for training by splitting the training data into two datasets. Each dataset is allied with Writer Independent (WI) and Writer Dependent (WD) models. The accuracy scores of each were calculated as 62.5% for WI and 75% for WD. Computational complexity of the algorithm is reduced by reducing the number of samples to 30 in training to WD.

Poddar et al. [50] has performed forgery detection and also signature/writer recognition tasks and applied CNN for signature verification and recognition. Author did

not provide any details regarding data acquisition or dataset. The signatures are treated as images and pre-processing techniques are applied before applying CNN. The proposed solution is based on WI technique as the whole dataset is passed to series of pre-processing phases. Test signature is identified on the basis of training set with both CNN and Crest-trough techniques. Forgery is detected using Harris and Surf algorithms working in a sequence. Crest-trough algorithm and CNN determines whether the test signature is part of the training set family or not. Once it is decided Harris and Surf algorithms further find forgery in the given test signature. The author claimed accuracy score for identification is 94% but its forgery detection accuracy score is 85-89%.

Rantzsch et al. [51] proposed offline signature verification through measuring triplet loss and applying Euclidean distance in order to minimize the distance among genuine and maximizing the distance among forgeries. The whole process is carried out through VGG-16 architecture of CNN. Model is trained by calculating loss and back propagating it through the network. Author has also used a self-modified version of VGG-16 by reducing a pooling layer, one fully connected layer and three convolutional layers.

Parcham et al. [52], has proposed a CapsuleNet (a framework of CNN) with a little modification and named it as Composite Backbone CapsNet. This is a combination of CNN and CapsNet. Two Conv layers followed by average pooling of previous conv layer and then Conv 2^{nd} and 3^{rd} layers and concatenated in which is again followed by avg. pooling layer before passed it to CapsNet and then three FC layers with 512, 256 and 128 vector size. The embeddings generated by this network is of size 128. The dataset used are CEDAR, GPDS-300, GPDS-synthetic, BHSig-260. The CBCapsNet results are 100, 92.94, 90.87 and BHSig-260 94.3 and 100. The cost of computation is quite heavy in training such models.

Generative Adversarial Network (GAN) has also been employed for signature verification by Wang et al. [53]. According to the approach defined in article SIGAN (ad-hoc GAN) is used for handwriting verification which in-turn verifies a signature. The results yielded 91.2% accuracy without generalizing it for other datasets.

Table 2.4: State of art comparison for deep learning techniques

Sr. No.	Author	Dataset	Technique	Performance (EER, FAR, FRR, Accuracy)	Critical Review
1.	Tolosana et al. [19], 2021	DeepSign DB	DTW, RNN	EER (4.2, 13.8)	DTW is a time series function and to align the time functions.
2.	Navid et al. [17], 2020	ICDAR Kaggle CEDAR	VGG-19	100 94.44 88	Author have applied VGG-19 through transfer learning with a supposition that this might not affect final classification as weights learnt can be used for further training. The model will likely to overfit on smaller datasets and the author did not bother to pre-train it models to a similar task as hand written text, that is why the accuracy is dropping.
3.	Poddar et al.[46], 2020	Any dataset	CNN Crest Trough Harris, SURF	85-89 Acc.	The technique pertaining to image pre-processing is vague as to how the author always finds and upward slant. No generalization results are given and no validation set used.
4.	Jahandad et al. [16], 2019	GPDS-Synthetic	GoogleNet Inception-v1 Inception-v2	Iv1-83% Acc. Iv3-82% Acc	Author has tested its system on only one dataset and did not generalize the results on other datasets.
5.	Alajrami et al.[44], 2019	Kaggle	Ad-hoc CNN	99.7% Acc.	The author did not provide generalized results and dataset selected is very small.
6.	Yapıcı et al. [45], 2019	GPDS-synthetic	Ad-hoc CNN	62.5% Acc WI 75% Acc WD	No generalization of model to other datasets is provided.
7.	Maergner et al. [26], 2018	GPDS MCYT	Graph Edit Distance (GED) Triplet Loss (CNN)	RF 0.41, SF 6.49 RF 1.05, SF 9.15	The computational cost of such system is quite high as both model require significant amount of heavy computations.
8.	Parcham et al. [48], 2021	CEDAR GPDS-300 GPDS-synthetic Bengali Hindi	CNN-CapsNet CB-CapsNet	100 92.94 90.87 94.3 100	Author has proposed a very unique formulation of the CapsNet model but training such models add computational cost.
9.	Rantzsch et al. [47], 2016	ICDAR (Dutch, Japanese)	VGG-16 Tripplet Loss	81.76 Acc, 93.39% Acc	VGG-16 is quite a heave network for such a small signature verification task with Tripplet Loss as the loss function.

10.	Wang et al [49], 2019	Chinese	GAN (SIGAN)	91.2	The model is complex and the author did not mention about the redundance for training the GANs twice. The method is not generalized for other datasets.

2.4 Transfer Learning (TL) Approaches

Transfer Learning is a type of Deep Learning which is used to avoid feature learning process from scratch. Researchers try to implement a pre-trained model which resembles their current problem to avoid extra burden of learning features. This technique also enables them to limit the learned features by manipulating the layers. There are several pre-trained models available; Res-Net [54], Alex-Net [55], Google-Net [56], VGG-Net [57], Mobile-Net [58] etc. All these models are specific to their datasets, solution and no. of parameters learned.

Most of the studies pertaining to transfer learning used a handwritten text due to its similarity with the handwritten signature images. In this sort of formation, the model is trained on a handwritten text dataset owing to the fact that these datasets are rather richer in form of samples as compared to signature datasets and due to constraints of not available publicly [26],[27]. GPDS synthetic dataset [28], which consists of 4000 users signatures with 24 genuine and 30 forged signatures is till date considered the largest non-public dataset available.

However, there are other signature datasets as well which are also used for state of art results and comparisons. Most of the authors [26],[27] and [29] have used multiple signature dataset to evaluate their transfer learning results. This is due to the model being biased towards a single dataset [30], and researchers always try to generalize their results in order to prove the authenticity of their proposed models. ResNet is one of the most used deep learning model as used by [27],[30],[59]–[62].

This section illustrates the use of TL for offline signature classifications. Younesian et al. [62] have combined active learning and transfer learning to cope with the small sample size of signature dataset. Transfer Learning is done using WI approach and Active Learning through SVM is classified opting WD approach. He opted transfer

learning for feature extraction through ResNet pre-trained on ImageNet dataset. He further managed intra-class variability though active learning approach by employing SVM to minimize confusion. A ResNet-50 is applied with identity shortcuts X for bypassing 3 convolutional layers each. This scheme results in a 7x7x2048 feature vector for each image. A query selection criterion is also opted for selection of the most significant instances among signatures and the annotating is using an oracle. There are three algorithms proposed for query signature selection, these are; Distance based sampling, Maximum Entropy and K-NN.

Author did not pick state of the art experimental settings and ignored validation set due to scarcity of signature data. Though author has calculated 83.60% accuracy but the confusion matrix is not given.

Hafemann et al. [29] has provided insight into the difficulties in the feature extraction process for signature verification. The author is of the view that the model is only trained on the positive (genuine) samples in the dataset and it is almost impossible to train a model for skilled forgeries. A novel approach is used based on Convolutional Neural Network. The model is trained on the basis of Writer Independent method, in which signatures are enrolled against users. While for classification between genuine and forged signatures, Writers Dependent approach is used. Each query signature is classified as genuine or forged. A double loss function (cross entropy) is used for this purpose. Initially to determine users and the second function for classification. Datasets of GPDS, CEDAR and Brazilian (PUC-PR) is used. GPDS has provided the better results than CEDAR and Brazilian (PUC-PR) due to its larger size.

Mersa et al. [27], use transfer learning approach using ResNet trained on Persian handwriting dataset and then fine-tuned for signature dataset and for classification SVM is used as it generalize well for small datasets. He has achieved a minimum of 9.02, 6.81 and 3.98 Equal Error Rate (EER) with 10 genuine samples of each signer using SVM as classifier.

Tsourounis et al. [26] used transfer learning model for the task and chose the hand-written text data due to its similarity with the hand-written signatures. Author used two separate terms for each type of dataset. One is called Auxiliary Domain (this domain is considered close with Handwritten signatures), consist of handwritten text from CVL

dataset for learning the model in first phase. Then target domain (consists of signature images) a signature is passed to another CNN model for learning features and finally the third model is used for genuine and forgery classification for the query signature.

Transfer learning in signature verification task still remined less explored due to uniqueness of the data. Researchers though explored the possibilities of training the model from scratch or the pre-tuned weights of the model on a separate dataset of handwritten text. This separate dataset are named as supporting or auxiliary dataset by some authors.

Table 2.5: State of the art comparison for deep transfer learning

Sr. No.	Author	Dataset	Technique	Performace (EER, FAR, FRR, Accuracy)	Critical Review
1.	Younesian et al. [62], 2019	UTSig	Pre-trained ResNet-50 Active Learning	85.3% Acc.	Author did not pick state of the art experimental settings and ignored validation set due to scarcity of signature data. Though author has calculated 83.60% accuracy but the confusion matrix is not given.
2.	Tsourounis et al [26], 2022	CVL Text Dataset CEDAR MCYT-75 GPDS-300	SigNet	EER 0.99 1.26 1.98	Although the results produced are impressive but insead using continuous stripes of text, words dataset can be used.
3.	Hafemann et al [29], 2017	GPDS-160 GPDS-300 MCYT-75 CEDAR Brazilian	SigNet	1.72 1.69 2.87 4.63 2.01	Model is trained only on genuine signatures and forgeries are random.
4.	Mersa et al [27], 2019	Persian Text MCYT-75 UTSig GPDS-synthetic	ResNet-8 SVM	EER 3.98 9.02 6.81	Persian text to learn weights, again the words choice would better for such task.

CHAPTER 3

METHODOLOGY AND IMPLEMENTATION

Our proposed approach is based on deep neural network for feature extraction from offline signature dataset and classify using distance based approach using application oriented mechanism as proposed by Hafemann et al. [8],[29]. This is a novel technique and no research has been done using this technique to the extent of our knowledge. An overview of the proposed methodology is shown in Figure 3.1.

Signatures are passed to the network in the form of triplets. Two genuine and one skilled forgery are selected from single author and pushed into the network. Network consist of MobileNet-V2 and it is trained using triplet loss. The outcome of this network is a feature map and sole purpose of this network is to minimize the distance between positive or genuine sample (Z_p) and genuine anchor (Z_a) of an author and maximizing the distance among the genuine anchor and skilled forgery (Z_n) of the same author in case of skilled forgery and genuine signature of other author for random forgery. This relationship is formulated in equation 3.1.

$$L_{Triplet} = \left[\left\| \int(Z_a) - \int(Z_p) \right\|_2^2 - \left\| \int(Z_a) - \int(Z_n) \right\|_2^2 + \alpha \right]_+ \qquad (3.1)$$

The result of this equation is always negative due to the factor that anchor-negative distances are desirably maximized. To counter this a margin α is added and the + sign at equation end indicate that the result should always be positive.

Figure 3.1 shows the training of MobileNet-V2 with triplet-loss function. First, the triplets (anchor, positive, negative) pre-processed images passes through the same MobileNet-V2 model. The weight and structure of the models are the shared among all the triplets.

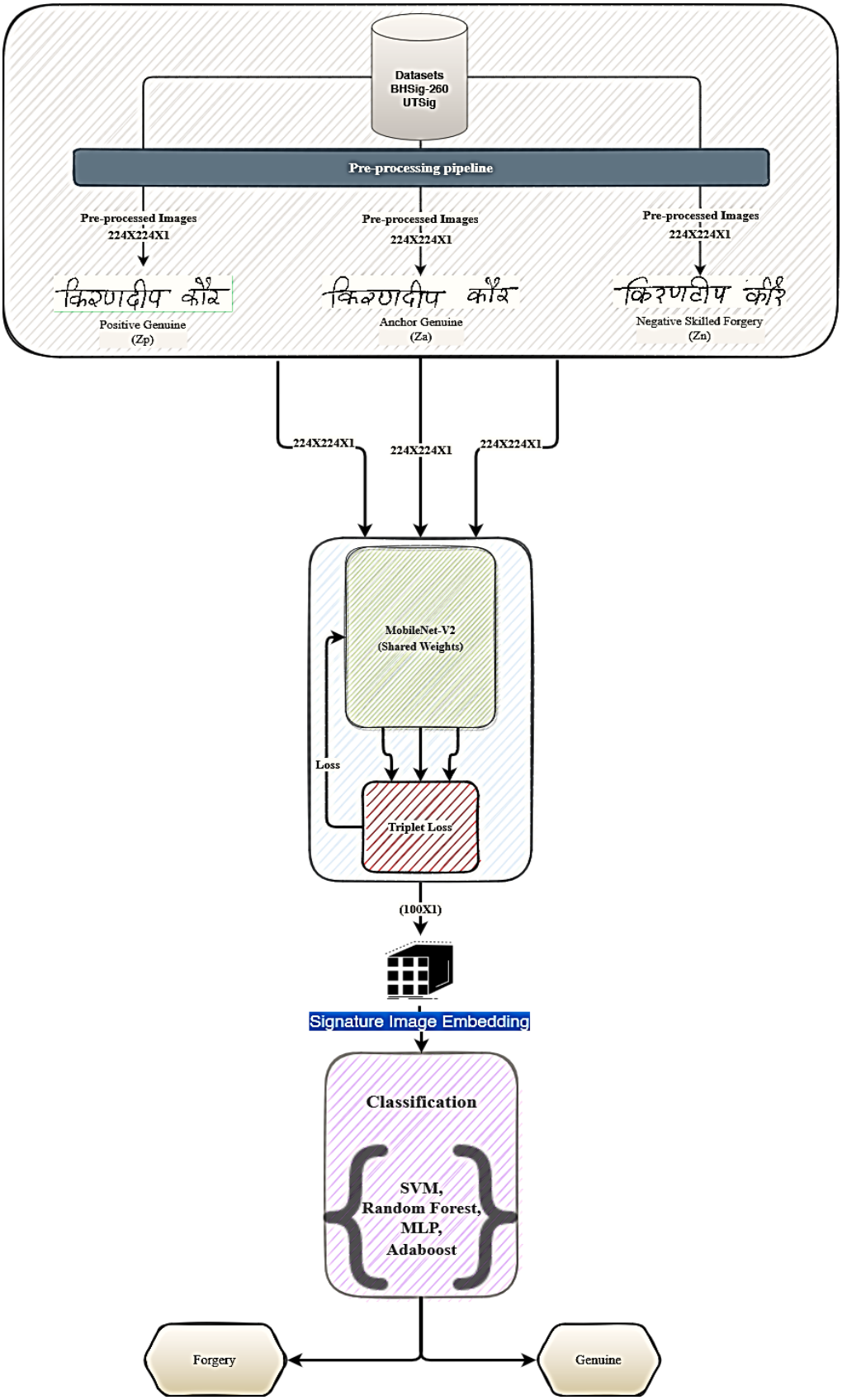


Figure 3.1: Proposed Methodology

This formation produces a feature vector of 100x1 for each image and this corresponding feature vector is now pushed into triplet loss after training. Euclidean distance among anchor and positives are reduced and negative points are squeezed out from the anchor point. There, it helps to form each individually corresponding cluster class in the embedded space.

Nguyen et al. [63] has proposed a similar model for detection of ailment in plants. Although there is no similarity between his dataset and our problem, but Rantzsch et al. [51] have used reduced VGG-16 with triplet loss and leaned feature map in an embedded space. He also used SVM for classification.

Signature verification through our system is segmented into four major steps, these are 1) dataset selection, 2) data pre-processing, 3) MobileNet, 4) Triplet Loss. Later part of this chapter discuss these segments one by one.

3.1 Dataset Selection

There are many datasets available for offline signatures such as GPDS (synthetic), BHSig-260 (Hindi-Bengali), CEDAR (English), UTSig (Persian) etc. A lot of work has previously been done on English datasets. Our work is focussed towards the local languages and a lightweight DNN model with better accuracy that can be application friendly and could be used on any device with minimum computational capacity. The decision of dataset selection is very tough as which language and amount of data is required for improved signature verification task. Various datasets are searched out and many of them are not even publicly available. So, this reduces the search to only available datasets other than English language.

3.1.1 Local Language Datasets

Most common among these datasets are Bengali and Hindi language dataset BHSig-260 is a largest available local language dataset other than English, which contains

24 genuine and 30 skilled forged signatures for each user for both datasets with 160 users are enrolled in Hindi and 100 users for Bengali. BHSig dataset [37], [64]. is used for this research based upon its size.

UTSig is another eminent dataset, which is acquired by University of Tehran. This dataset belongs to Persian language and it contains data of 115 users with 27 genuine, 6 skilled forgeries, 36 simple forgeries and 3 opposite hand forgeries.

3.2 Data Pre-processing

Image pre-processing is a pre-requisite for any machine learning task. In our methodology signature images are initially converted to grey scale from RGB. This will reduce the computational load over the network. Colour inversion is done to detect the noise easily. Otsu denoising is used with 7x7 Gaussian blur for smoothing and denoising the image. In the end, image resizing is carried out to mitigate the complexity of calculations through a network. Resizing is performed to reduce the largeness of DNN model. The resized imaged used is of size (224x224x1). Each step is depicted in Figure 3.2 along with its input and output.

3.3 Mobile Net

MobileNets are basically designed to attain better accuracy with less computational resources. The design phenomena of MobileNets are inherited from XceptionNet. The most important element of this network is Depthwise Seperable Convolution as described in [58]. Depthwise Seperable Convolution is also a type factorized convolution which consist of a 1x1 pointwise convolution. To understand this a normal convolution works on width, height and depth (in 3D) at the same time, but in MobileNet the details of all the channels is captured along with their corelation among them. So this is a two way information capturing, firstly you gather features from one

channel and then find the relation among other channels features that are also captured separately.

This methodology, however reduce the number of operations by 8-9 times from normal convolution but also the accuracy as claimed by the researcher itself. Two main hyper parameters it used to further reduce the computations is Width Multiplier for thinning the model and Resolution Multiplier for reduction in representation.

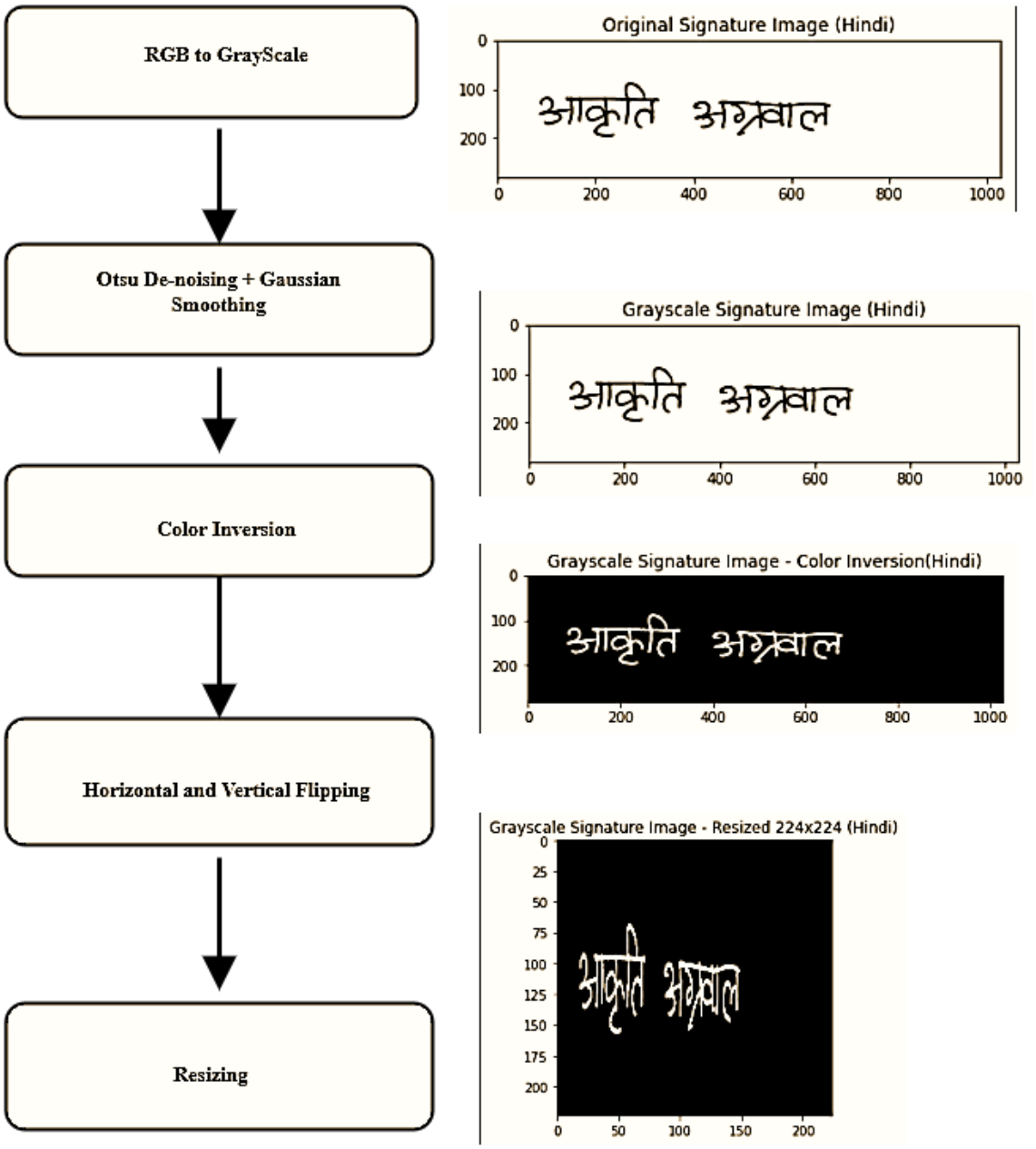


Figure 3.2: Pre-processing Steps

The model is configured to extract only features from the signatures. MobileNet-V2 is employed with triplet loss for the task as proposed by Nguyen et al. [63] for detection of disease in plants, shown in Figure 3.1. Pre-processed images are fed into the model as triplets. In case of skilled forgery, the negative sample is a skilled forgery of the same author for which the positive and the anchors are chosen, but in the case of random forgery, anchor and genuine signatures are from one writer whereas the negative sample is usually considered from genuine signature of other signers.

The selection of DNN model is based upon the lightness of the model in terms of computational complexities. MobileNet-V2 is a light weight depth-wise model and it is very similar to original Mobile-Net which is trained of ImageNet dataset. In our scenario Mobile-Net is applied from scratch. Mobile-Net is designed for working on less computationally resourceful devices.

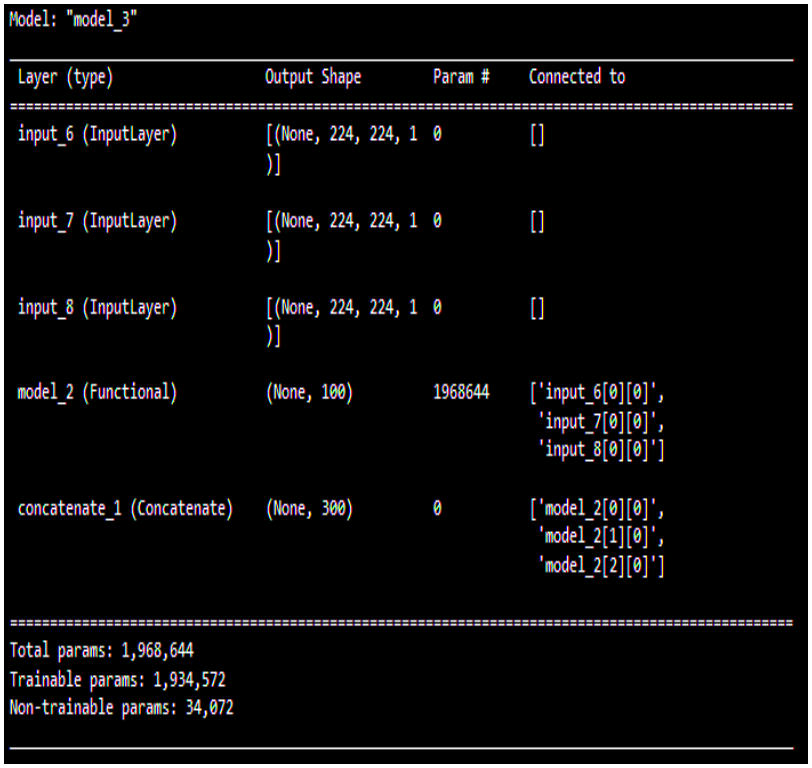
The weights are tuned by training the model and through back propagation. The loss function holds the responsibility to bring positive and anchor closer and in the meanwhile pushing negative away from anchor. In this way the loss is calculated and propagated back to the model for tuning of weights after each epoch. A *softmax* function at the end distribute the probabilities of distances between 0 and 1. Our model is fed with genuine signatures as positive samples, skilled forgeries as negative samples and for anchor samples the not only genuine signatures are used but also the flipped images too randomly to increase the variation during training. Flipping is done in horizontally and also vertically. Primarily MobileNet-V2 uses 3.4 million parameters which are quite less than VGGNet, ResNet, AlexNet, GoogleNet etc. Nevertheless, performances of later are quite higher based upon their size and required computational resources. Our model has just less than 2 million trainable parameters, which is shown in Figure 3.3.

3.4 Triplet Loss

Triplet loss initially proposed by Hoffer et al. [65] and by that author transformed from simple positive and negative classes metric comparison to a new way by introducing a self-inflicted anchor class which is also usually a positive class instance. Negative and

positive distances are measured from anchor with purpose to minimize distance between a positive and anchor and maximizing the distances among negative with anchor in an embedding space and the learned feature vector is called embedding.

Same is used inside our proposed OfSV technique. Our proposed model used offline triplet. These are similar to hard coding the triplets before passing through the network. This implies that triplets are formed before training and these triplets may not be altered at any later stage.



```
Model: "model_3"
_________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #   Connected to
=================================================================================================
input_6 (InputLayer)         [(None, 224, 224, 1       0         []
                             )]

input_7 (InputLayer)         [(None, 224, 224, 1       0         []
                             )]

input_8 (InputLayer)         [(None, 224, 224, 1       0         []
                             )]

model_2 (Functional)         (None, 100)               1968644   ['input_6[0][0]',
                                                                  'input_7[0][0]',
                                                                  'input_8[0][0]']

concatenate_1 (Concatenate)  (None, 300)               0         ['model_2[0][0]',
                                                                  'model_2[1][0]',
                                                                  'model_2[2][0]']

=================================================================================================
Total params: 1,968,644
Trainable params: 1,934,572
Non-trainable params: 34,072
_________________________________________________________________________________________________
```

Figure 3.3: Proposed number of trainable parameters

On the contrary online triplet concept proposed by Schroff et al. [66]. The author described it by shifting from somewhat static triplets to creating triplets during the training process by calculating minimum and maximum inside a minibatch. The model diagram is shown in Figure 3.4.

3.5 Classification

As proposed by Hafemann et al. [29] the DNN model is applied only to extract automatic features from the input image. In this work the features are extracted calculating triplet loss after extracting feature map through MobileNet-v2.

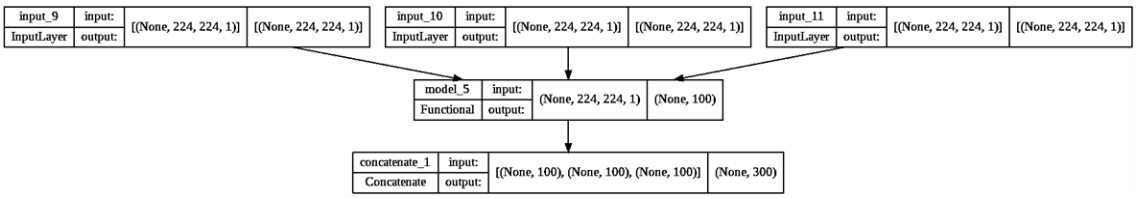


Figure 3.4: Triplet-model diagram

Signature verification is a binary task and most of the task performed provides output as to whether a given signature is genuine or forged. Many researchers [11],[23] and [45] have applied SVM for classification. One class SVM have also been emphasized by Hafemann et al. [8] for model which are trained on only genuine signatures.

We have used different classifiers which is rarely been used previously for such task. Random Forest [67] classifier is an example of ensemble classifier and they are based upon decision trees. Support Vector Machine (SVM), Adaboost and Multi-Layer Perceptron (MLP) along-with ensemble of these classifiers. In pursuit of building an application independent approach, no threshold being set for the results, hence making this approach accessible and applicable in applications without considering the barrier of computational overhead.

CHAPTER 4

EXPERIMENTAL SETTINGS AND RESULTS

As mentioned earlier, the model is passed through triplets of signatures. The triplets are generated *offline*. Offline term refers for the creation of triplets statically before training the model whereas a new phenomenon has evolved recently for creating triplet dynamically and termed as *online* triplets' creation. We have implemented offline triplets to calculate loss and feature extraction due to the fact that data is quite scares in case of local languages and this is too done without replacement. In order to calculate total number of triplets formed for a specific dataset following formulation is used as in equation 1.

$$L_{Triplets} = \left[\frac{g}{2} + f\right] \quad (4.1)$$

Where 'g' denotes genuine signatures which belongs to the set of genuine signatures 'G' and 'f' is a skilled forgery which is from forgeries set 'F'.

This formulation is universal for all datasets irrespective of their genuine and forged samples, but count of the triplets created differ from one dataset to another depending upon their genuine sample size. As this formulation creates offline triplets and without replacement, number of samples across datasets remains constant for that specific dataset.

Another noteworthy thing regarding model training is that the model is trained in a writer independent fashion, the model is passed images (2 genuine and one forged) in APN form and MobileNet-V2 generates feature vectors which is then passed to triplet loss. Triplet loss would be reducing the distance between anchor and positive sample and

at the same instance stretching the negative sample away from anchor. It then calculates the loss and back propagating it to tune the weights for the model.

Model creates embeddings for each batch it processes for further classification. The embeddings are a vector of length 100. These embedding are then passed through ensemble algorithm for classification against given images of signatures from genuine and forgery from same dataset to predict whether the given signature is genuine or forged. The forgeries used in these testing are skilled forgeries which are extremely similar to genuine signature. The reason of these similarities is mainly due to intra-class variability meaning that signature of same author may differ with each instance and skilled forgery is actually imitating the original signatures by knowledge of author name and signatures. This makes the detection quite difficult to detect. We have discussed such similarities later in this section.

4.1 Experimentation and Configurations

The evaluation matrices being used by researchers are False Acceptance Rate (FAR), False Rejection Rate (FRR), Equal Error Rate (EER), accuracy, F1-score, precision and recall. We have opted for accuracy, F1-score, recall and precision.

Several settings and tests being conducted on different local language datasets. The main hindrance in OfSV is the scarcity of data. The data is so small that heavy DNN models and architectures overloads while training. MobileNet-V2 proposed by Howard et al. [58] performs two operations while training. According to the author the big trade-offs of this model is latency versus accuracy. If you opt for speed the accuracy drops and vice versa, increased accuracy will decrease speed and may tend to overfit. Another thing that is pertinent to mention here that MobileNets architectures are suitable for satellite imageries and object detection. Our proposed model will try to balance between both latency and accuracy through loss function. Local language signature dataset is quite unique in its nature and the overhead of writing separate data pipelines for each dataset.

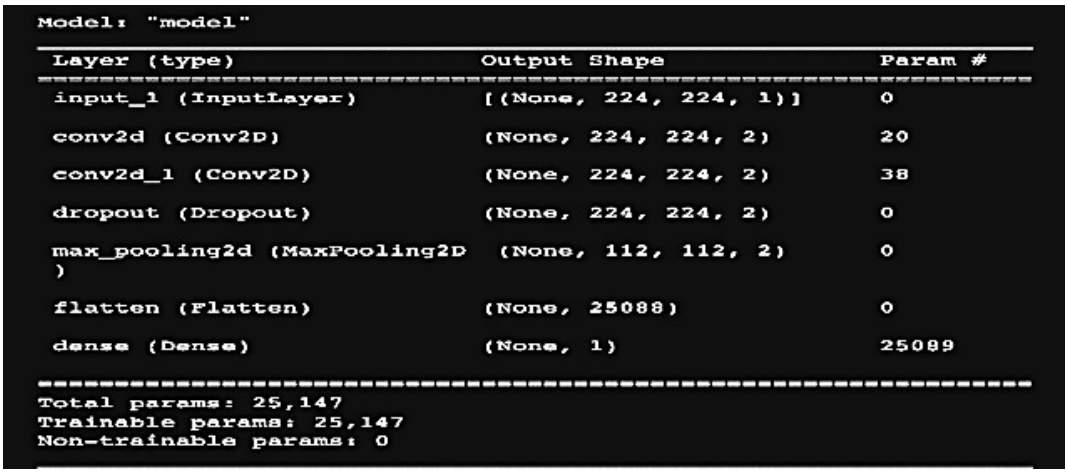
Initially in testing phase genuine forgeries are only considered and applied MobileNets with loss functions proposed by Avola et al. [68]. Skilled forgeries are quite

hard to detect as a writer genuine signature may vary each time and such intra-class variability reduces the chances for detection of a skilled forgery.

Avola et al. [68], proposed a reduced Signet model with dual loss functions for two separate outcomes. One loss function is to identify the author and the other to classify the signatures as being genuine or forged. BHSig-260 is used for evaluation.

In our setting, MobileNet-V2 with same loss function as proposed by [68], is trained and the model started overfitting with training accuracy after only 5 epochs was 71% and validation accuracy remained at 45% for signature classification task.

Similarly, VGG-19 is also trained using the same loss functions but the results remained the same. Consequently, only one block of VGG-19 is trained with RSigNet loss functions. It produces the training accuracy of 87% with validation accuracy at 72%. The single block is shown in Figure 4.1.



```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 1)]     0

 conv2d (Conv2D)             (None, 224, 224, 2)       20

 conv2d_1 (Conv2D)           (None, 224, 224, 2)       38

 dropout (Dropout)           (None, 224, 224, 2)       0

 max_pooling2d (MaxPooling2D  (None, 112, 112, 2)      0
 )

 flatten (Flatten)           (None, 25088)             0

 dense (Dense)               (None, 1)                 25089

=================================================================
Total params: 25,147
Trainable params: 25,147
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.1: Single block of VGG-19

It is quite apparent, that any deeper network would get overfit due to scarcity of data. On the other side a simple and reduced model might produce better results without overfitting. Another setting that can be taken into consideration is dropouts at various level and grid-search approach for hyper parameter tunning.

Our proposed model reduces the chances of overfitting and the training process will only produce signature embeddings and classification is done through various

classifiers. Table 4.1 will show the results of our model with BHSig dataset on various classifiers. As local languages are employed for evaluation, the model is trained on all the datasets used separately. Generally, in English language datasets the authors exploit the facility of training the model once by loading all datasets if using multiple datasets.

We have tried to execute as many epochs as possible by increasing epoch to 300 and patience to 10. With these setting the model is trained after executing 73 epochs before the loss did not improve and early stopping triggered. We have tried these setting with batchsize equals to 32 and also with author-based batch size meaning the batch size is equal to the number of authors in a dataset. But this did not produce any significant improvement in the accuracy.

Table 4.1: Test results of Bengali Dataset with different classifiers

Classifiers	Accuracy	Signatures	Precision	Recall	F1-score
MLP	77%	Forged	81%	77%	79%
		Genuine	72%	77%	74%
SVM	76%	Forged	84%	72%	77%
		Genuine	70%	82%	75%
Adaboost	78%	Forged	78%	84%	81%
		Genuine	78%	70%	74%
Ensemble (MLP, SVM, RF, Adaboost)	81%	Forged	84%	82%	83%
		Genuine	78%	80%	79%
Random Forest	84%	Forged	84%	88%	86%
		Genuine	84%	79%	81%

The split in the above results is 90-10 due to scarcity of data. In Bengali language the data of users is contained at 100 users whereas in Hindi dataset the user count is 160 with same 24 genuine samples and 30 skilled forgeries. The above results are produced against skilled forgery as negative example. Skilled forgeries are the hardest to detect as they performed by the person whom has known the name and signature pattern of the original signatory. Figure 4.2 shows the confusion matrix based upon above settings.

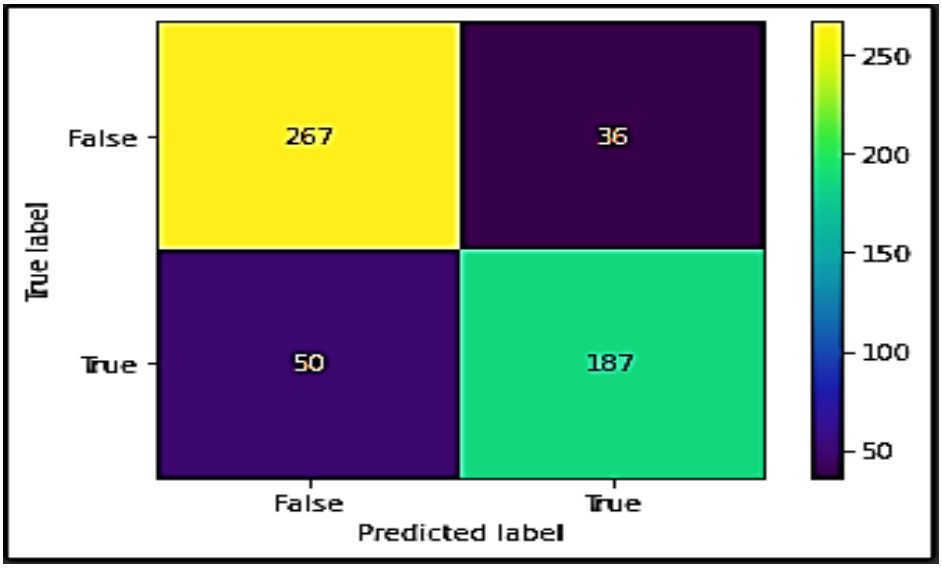


Figure 4.2: Confusion matrix for Random Forest on Bengali Dataset

Hindi dataset is very unique due to its shape of alphabets and writing style. Resemblance between genuine and skilled forgery signatures is very high. In order to generalize the model on other datasets Hindi dataset is used which produced following results shown in Table 4.2.

Table 4.2: Test results of Hindi Dataset with different classifiers

Classifiers	Accuracy	Signatures	Precision	Recall	F1-score
MLP	74%	Forged	72%	85%	78%
		Genuine	77%	60%	67%
SVM	74%	Forged	73%	86%	79%
		Genuine	77%	61%	68%
Adaboost	73%	Forged	71%	86%	78%
		Genuine	77%	58%	66%
Ensemble (MLP, SVM, RF, Adaboost)	72%	Forged	71%	84%	77%
		Genuine	74%	57%	64%
Random Forest	73%	Forged	72%	82%	77%
		Genuine	73%	61%	66%

Although the results are not that significant due to the fact similarity among skilled forgeries and genuine signatures is very high, as skilled forgeries are imitated from genuine signatures and signer knows the name and signature flow of the original person.

Commonality among Bengali and Hindi datasets is that for every user there are 24 genuine and 30 skilled forgeries which results in 12 triplets for each user. The model is trained in writer independent way and the results are oriented to predict whether a signature in genuine or forgery. The split is 90-10, 90 being training part and 10 as testing. The split is performed on the learned embedding of the signatures with vector size fixed to 100 for each triplet. A confusion matrix for Hindi dataset is shown in Figure 4.3.

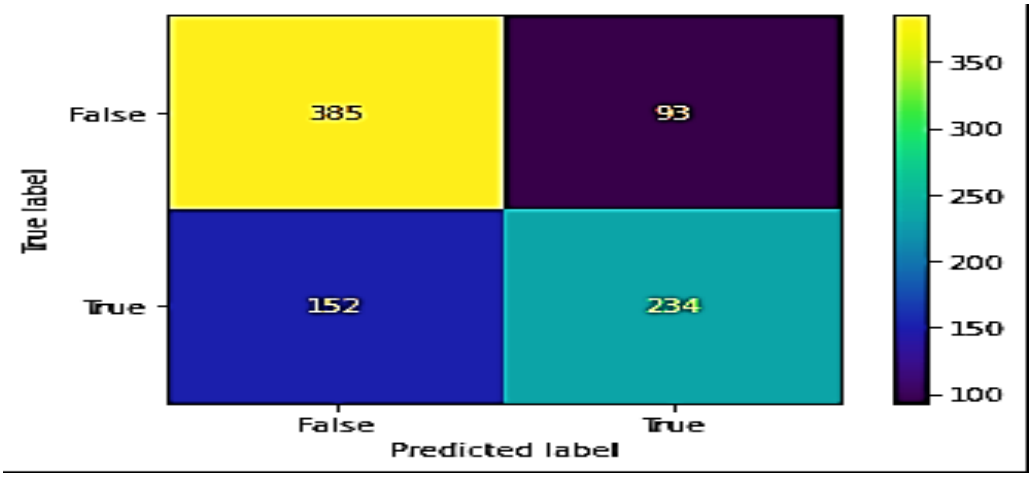


Figure 4.3: Confusion matrix for Random Forest on Hindi Dataset

As our estimation the results predicted are not as comparable with heavy neural network models, but proposed model is definitely reduced the computational cost and the results have shown potential in it own class.

4.2 Model Analysis and Comparison

Subsequently after performing sufficient number of variations, we have concluded that number of epochs though reduce the loss but accuracy almost remained the same.

Otsu denoising and Gaussian blur with 7x7 size is performed instead of eliminating noise with pixel values. We have flipped the images of genuine signatures to be used as only anchor sample on random basis.

Model was trained on only first 80 users with 32 epochs per dataset, learning rate was fixed at 0.0001, patience is set to 5, although it seemed highly unlikely from previous experimentations that the model would exhaust any patience value greater than 1. Minibatch size is set to 4 and the model only monitors loss.

We have performed separate blocks of codes on the local signature datasets. The model only gives signature embedding of each image. These images are then classified with multiple classifiers and their ensemble. K-validation is executed using 10 folds with 90:10 ratio of split. Each classifier is given separate model parameters after grid search.

We have trained and classified two major states of the art local language signature datasets using the above setting. While using BHSig-260, Hindi and Bengali language datasets are trained and evaluated separately.

Bengali dataset showed relatively better results than the rest. The losses evaluation of these datasets with the model training which is shown in Figure 4.4, Figure 4.5 and Figure 4.6.

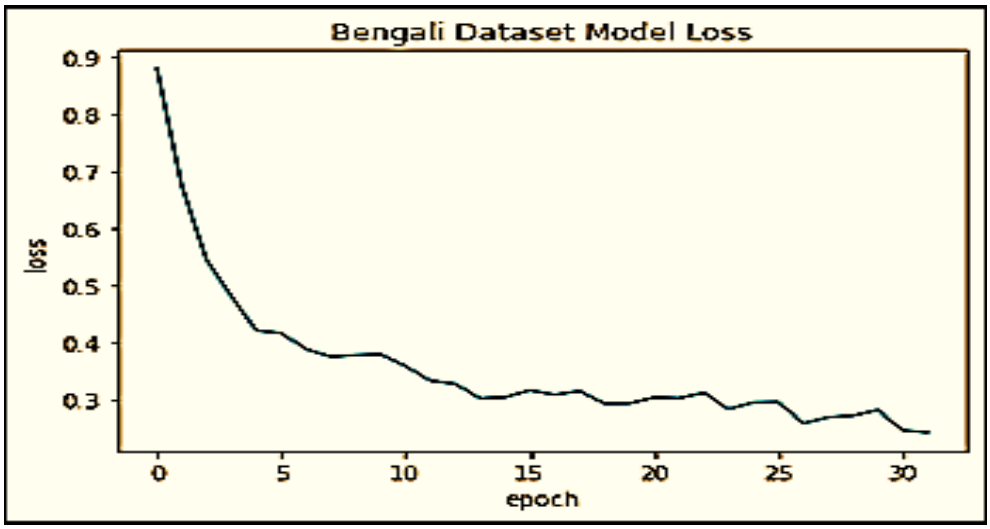


Figure 4.4: Bengali dataset loss

In Figure 4.4, loss of the model started dipping initially till the 15 epoch on Benagali dataset and then it started fluctuating between 0.2 and 0.3. Although the loss keeps on declining gradually as depicted in the above graph but this has not significantly improved the test results. That is why the number of epoch are reduced to 32 for all the datasets.

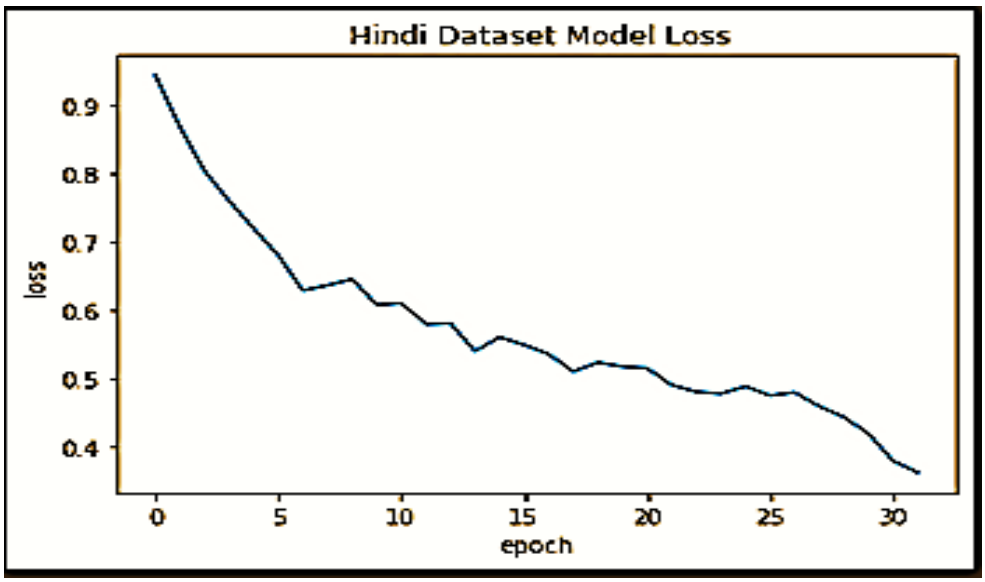


Figure 4.5: Hindi dataset loss

Hindi dataset though contains more users than Benagli but the loss did not stated dipping immediately as in the case of Bengali. However, the inclination of loss is a lesser steep but the loss keeps on decreasing till last epoch (32) and it is also evident that the loss suddenly decreased after 25th epoch, but again number of epoch beyond 32 has not produced any noticeable outcome from testing.

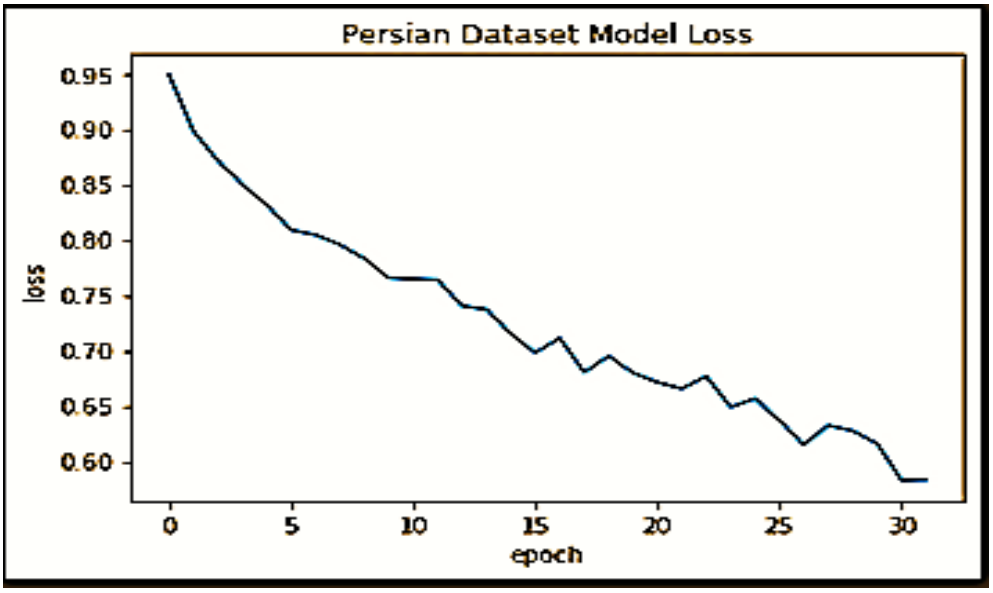


Figure 4.6: Persian dataset loss

Classification is carried out on 90:10 ration and k-fold validation is also done with k = 10 on all the images in that particular dataset. The count of total images in Bengali dataset are 100 x (24 + 30) = 5400 and the 90:10 ratio becomes 4860 as training part and 540 as test. The significance of this it that usually research select a random user pool from dataset and perform classification. This approach is called writer dependent, whereas our approach is completely based upon writer independent. Table 4.3, Table 4.4 and Table 4.5 show the evaluation on different datasets.

Table 4.3: Classifiers comparison test result from Bengali

Classifiers	Accuracy	Signatures	Precision	Recall	F1-score
MLP	74%	Forged	81%	70%	75%
		Genuine	67%	79%	73%
SVM	83%	Forged	82%	89%	86%
		Genuine	85%	80%	80%
Adaboost	81%	Forged	80%	87%	83%

		Genuine	81%	73%	77%
Ensemble (MLP, SVM, RF, Adaboost)	84%	Forged	83%	89%	86%
		Genuine	85%	77%	81%
Random Forest	86%	Forged	85%	93%	89%
		Genuine	89%	78%	84%

Table 4.4: Classifiers Comparison test result from Hindi

Classifiers	**Accuracy**	**Signatures**	**Precision**	**Recall**	**F1-score**
MLP	74%	Forged	73%	84%	78%
		Genuine	76%	62%	68%
SVM	76%	Forged	75%	85%	80%
		Genuine	78%	64%	70%
Adaboost	74%	Forged	74%	83%	78%
		Genuine	75%	64%	69%
Ensemble (MLP, SVM, RF, Adaboost)	75%	Forged	74%	85%	79%
		Genuine	77%	63%	70%
Random Forest	77%	Forged	70%	55%	62%
		Genuine	80%	88%	84%

Table 4.5: Classifiers comparison test result from UTSig

Classifiers	Accuracy	Signatures	Precision	Recall	F1-score
MLP	62%	Forged	63%	88%	74%
		Genuine	58%	25%	35%
SVM	63%	Forged	63%	92%	75%
		Genuine	63%	20%	30%
Adaboost	62%	Forged	63%	91%	74%
		Genuine	60%	20%	30%
Ensemble (MLP, SVM, RF, Adaboost)	62%	Forged	64%	84%	72%
		Genuine	55%	29%	38%
Random Forest	63%	Forged	63%	92%	75%
		Genuine	63%	21%	31%

Confusion matrix is very unique way to represent the classification problem. It provides an easy overview of actual and predicted values. Confusion matrix is a good measure in cases where accuracy can be deceptive in cases where dataset is not poised. It is a visualization technique for evaluating performance of a classifier. Similar to above approach separate confusion matrices for every classifier is visualized and shown in Figure 4.7, Figure 4.8 and Figure 4.9.

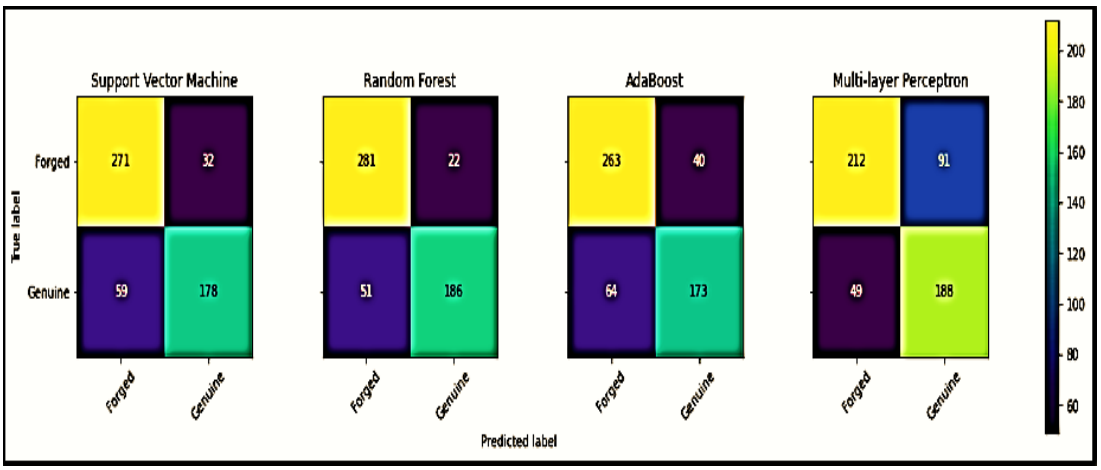


Figure 4.7: Confusion matrices for test results of Bengali

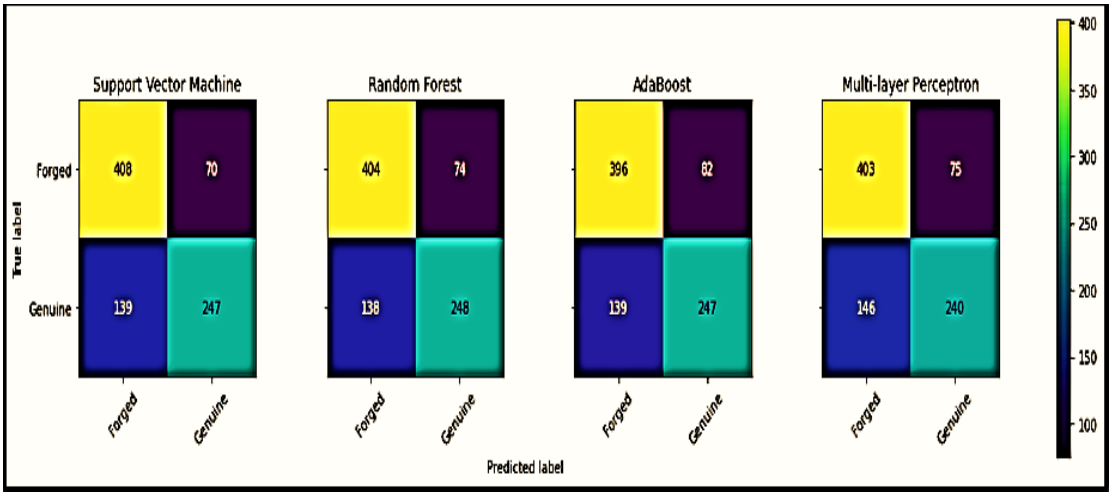


Figure 4.8: Confusion matrices for test results of Hindi

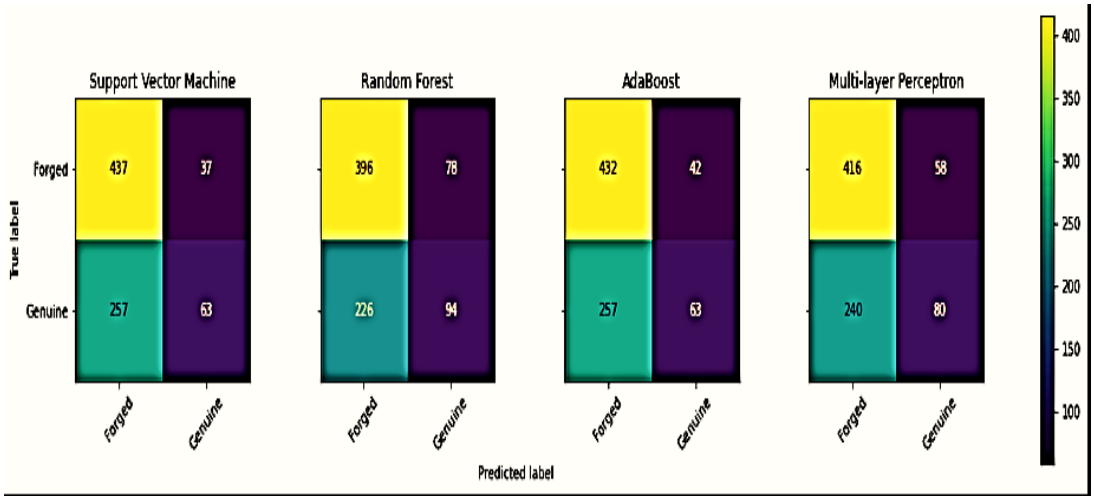


Figure 4.9: Confusion matrix for test results of UTSig

ROC-curve is a curve that indicates the classification performance of a model at several thresholds of classification. It is a metric that is usually more indicative of the performance in the case of binary classification. The graph is actually a plot between true positive rate (TPR) and false positive rate (FPR). Initially, it is evident that by lowering the threshold for classification, the model predicts more samples as positive thus getting more true positives and false positives and vice-versa.

Ideally, moving from left to right, curve should reach to 1.0 value of TPR and should remain there till the value of FPR also equals 1.0. Auc-Roc curve is also plotted for these classifiers. It is also a good matric to visualize the probabilities to demonstrate the model capability to detached classes. Figure 4.10, Figure 4.11 and Figure 4.12 demonstrate these curves for each dataset.

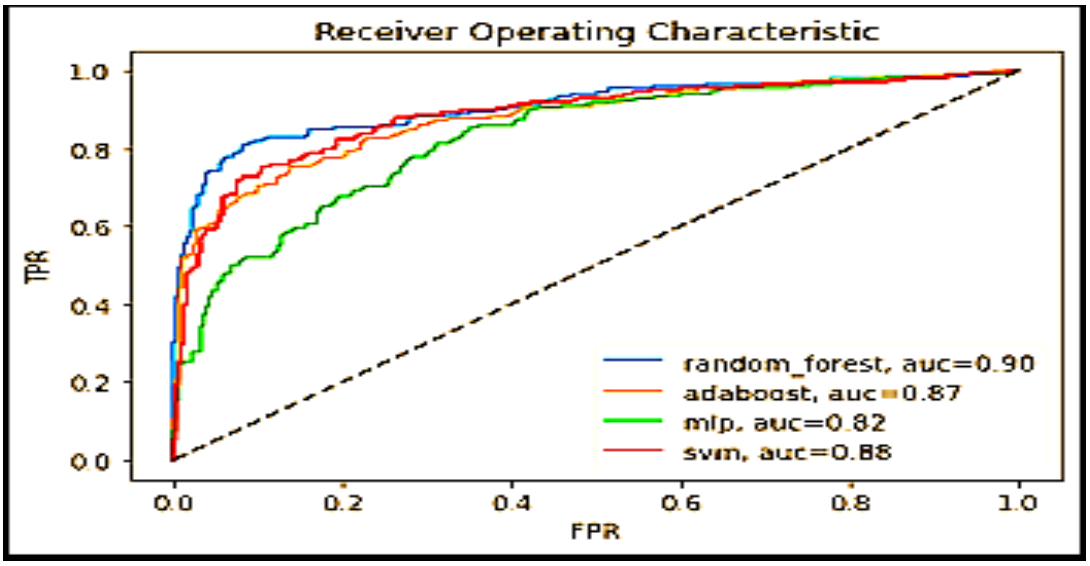


Figure 4.10: AUC-ROC curve for test results of Bengali

As elaborated earlier, in Figure 4.10, for Bengali dataset, the curve has a sharp rise and reaches the desired value of 1 or 0.9 precisely, which is a very good score. Similarly, in Figure 4.11, shows curve for Hindi dataset, the performance of multiple classifiers is very comparative and they tend to achieve very similar scores. The rise of the curve is gradual as compared to Bengali dataset achieving the maximum score of 0.82. However, in Figure 4.12, the AUC value is lower than both Bengali and Hindi datasets. The trend for all the classifiers is quite similar with maximum score 0.63 by MLP classifier.

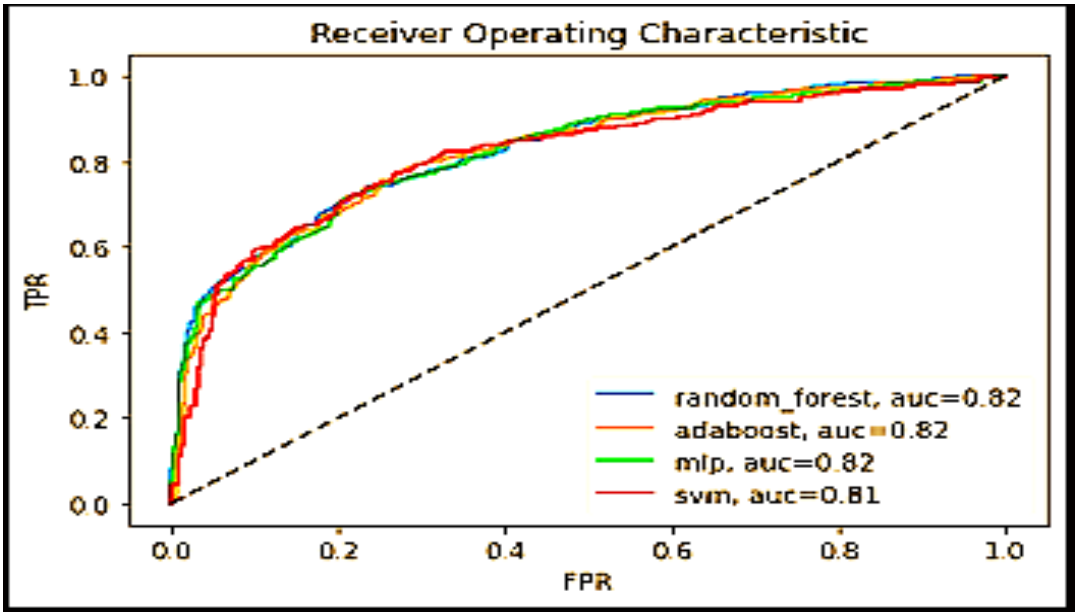


Figure 4.11: AUC-ROC curve for test results of Hindi

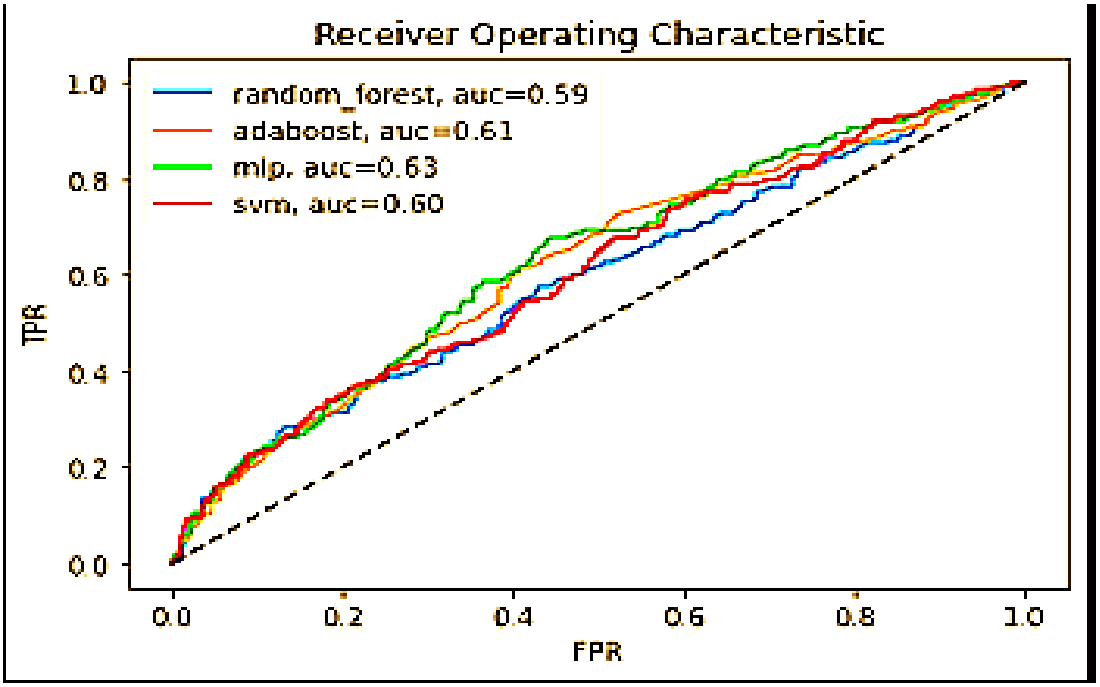


Figure 4.12: AUC-ROC curve for test results of UTSig

The common evaluation metrics used in literature is Equal Error Rate (EER). EER measure is very suitable for verification goals and is a segment on ROC curve where false acceptance rate (FAR) and false rejection rate (FRR) collide. So the EER for each dataset is computed for the state of art comparison with the literature.

Table 4.6: Performance comparison Bengali [37]

Model	Mode	EER/Accuracy
Dutta et al. [69], 2016	WI	14.1 EER
Dey et al. [70], 2017	WI	13.89 EER
Pal et al. [71], 2016	WD	33.82 EER
Proposed	**WI**	**14.4 EER**

Table 4.7: Performance comparison Hindi [37]

Model	Mode	EER/Accuracy
Dutta et al. [69], 2016	WI	14.10 EER
Dey et al. [70], 2017	WI	15.36 EER
Pal et al. [71], 2016	WD	24.47 EER
Proposed	**WI**	**25.6 EER**

Table 4.8; Performance comparison UTSig [36]

Model	Mode	EER/Accuracy %
Mersa et al. [27], 2019	WI	9.8 EER
Maergner et al. [72], 2019	Hybrid	14.9 EER

Soleimani et al. [36], 2017	CV	29.71 EER
Proposed	**WI**	**43.5 EER**

Our proposed methodology has demonstrated promise given the number of parameters. However, in case of UTSig similarity among skilled forgeries and genuine signatures is very high and also the number of samples per user are 6 which is quite low than genuine sample which are 27. It is also important to notice that in Bengali there are 30 skilled forgeries per user and the images are not cropped as well. Intra-class variability poses a very significant challenge in our case of skilled forgeries if the number of samples are less.

4.3 Conclusion

We have implemented MobileNet-V2 with triplet loss and trained embeddings of signatures in Euclidean space. Although our model demonstrated promise and achieved reasonable accuracy with fewer trainable parameters but was not able to get desired results. We assume this can be improved further with extensive task specific pre-processing and cropping the region of interest (ROI) from signature images.

Classification process may hamper due to variation of ROI among the signature images of different datasets. A generalized pre-processing technique for local languages signature dataset is quite hard to formulate. We suppose that further probe into finding the standardized pre-processing techniques is mandatory for better feature extraction.

Triplet loss has been used in various research articles with state-of-the-art outcome. It is also been implemented in signature verification methodologies. Nevertheless we expected that online triplet loss as implemented in [66], may improve the classification of signatures. This may be a separate research process as well.

Finally, the we consider Mobile Nets are a very lightweight and desirable approach for this task, considering the real-world scenario where forgeries in signatures

are a common fraud, and providing a fast and efficient technique with comparative accuracy may cause some reduction in such crime.

MobileNet-V3 are theoretically enticing as being fast and requires fewer trainable parameters than MobileNet-V2. MobileNet-V2 has shown great promise with respect to speed and accuracy, and after improving and generalizing MobileNet-V2, we will move to MobileNet-V3 for more speed and fewer trainable parameters.

REFERENCES

[1] N. Xamxidin, Mahpirat, Z. Yao, A. Aysa, and K. Ubul, "Multilingual Offline Signature Verification Based on Improved Inverse Discriminator Network," *Information*, vol. 13, no. 6, p. 293, 2022, doi: 10.3390/info13060293.

[2] J. A. P. Lopes, B. Baptista, N. Lavado, and M. Mendes, "Offline Handwritten Signature Verification Using Deep Neural Networks," *Energies*, vol. 15, no. 20, p. 7611, Oct. 2022, doi: 10.3390/en15207611.

[3] L. G. Hafemann, L. S. Oliveira, and R. Sabourin, "Fixed-sized representation learning from offline handwritten signatures of different sizes," *Int. J. Doc. Anal. Recognit.*, vol. 21, no. 3, pp. 219–232, 2018, doi: 10.1007/s10032-018-0301-6.

[4] K. Bibi, S. Naz, and A. Rehman, "Biometric signature authentication using machine learning techniques: Current trends, challenges and opportunities," *Multimed. Tools Appl.*, vol. 79, no. 1–2, pp. 289–340, 2020, doi: 10.1007/s11042-019-08022-0.

[5] K. S. Radhika and S. Gopika, "Online and offline signature verification: A combined approach," *Procedia Comput. Sci.*, vol. 46, no. Icict 2014, pp. 1593–1600, 2015, doi: 10.1016/j.procs.2015.02.089.

[6] A. Chugh, C. Jain, P. Singh, and P. Rana, "Learning approach for offline signature verification using vector quantization technique," in *Advances in Intelligent Systems and Computing*, 2015, vol. 337, pp. 337–344. doi: 10.1007/978-3-319-13728-5_38.

[7] R. Tolosana *et al.*, "SVC-onGoing: Signature verification competition," *Pattern Recognit.*, vol. 127, p. 108609, 2022, doi: 10.1016/j.patcog.2022.108609.

[8] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Offline handwritten signature

verification - Literature review," *Proc. 7th Int. Conf. Image Process. Theory, Tools Appl. IPTA 2017*, vol. 2018-Janua, pp. 1–8, 2018, doi: 10.1109/IPTA.2017.8310112.

[9] G. S. Eskander, R. Sabourin, and E. Granger, "Hybrid writer-independent writer-dependent offline signature verification system," *IET Biometrics*, vol. 2, no. 4, pp. 169–181, 2013, doi: 10.1049/iet-bmt.2013.0024.

[10] M. B. YIlmaz and B. Yanikoğlu, "Score level fusion of classifiers in off-line signature verification," *Inf. Fusion*, vol. 32, pp. 109–119, 2016, doi: 10.1016/j.inffus.2016.02.003.

[11] M. B. Yilmaz, B. Yanikoglu, C. Tirkaz, and A. Kholmatov, "Offline signature verification using classifier combination of HOG and LBP features," *2011 Int. Jt. Conf. Biometrics, IJCB 2011*, 2011, doi: 10.1109/IJCB.2011.6117473.

[12] D. Impedovo and G. Pirlo, "Automatic signature verification: The state of the art," *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 38, no. 5, pp. 609–635, 2008, doi: 10.1109/TSMCC.2008.923866.

[13] R. Plamondon and G. Lorette, "Automatic signature verification and writer identification - the state of the art," *Pattern Recognit.*, vol. 22, no. 2, pp. 107–131, 1989, doi: 10.1016/0031-3203(89)90059-9.

[14] V. R. X, "Enhanced Digital Human Signature Verification over Web and Mobile Interfaces," *J. Netw. Commun. Emerg. Technol.*, vol. 6, no. 5, pp. 49–52, 2016.

[15] M. Yasmin, M. Sharif, I. Irum, W. Mehmood, and S. L. Fernandes, "Combining multiple color and shape features for image retrieval," *IIOAB J.*, vol. 7, no. 3Special Issue, pp. 97–110, 2016, [Online]. Available: https://www.researchgate.net/publication/307578097_Combining_multiple_color_and_shape_features_for_image_retrieval

[16] K. Rajpal and P. Choudhary, "Handwritten Signature Verification Based on SURF Features Using HMM," *Int. J. Comput. Sci. Trends Technol.*, vol. 3, no. 1, pp. 187–195, 2015.

[17] A. Hamadene, Y. Chibani, and H. Nemmour, "Off-line handwritten signature

verification using contourlet transform and co-occurrence matrix," in *Proceedings - International Workshop on Frontiers in Handwriting Recognition, IWFHR*, 2012, pp. 343–347. doi: 10.1109/ICFHR.2012.245.

[18] P. S. Deng, H. Y. M. Liao, C. W. Ho, and H. R. Tyan, "Wavelet-based off-line handwritten signature verification," *Comput. Vis. Image Underst.*, vol. 76, no. 3, pp. 173–190, 1999, doi: 10.1006/cviu.1999.0799.

[19] M. S. Shirdhonkar and M. B. Kokare, "Off-Line Handwritten Signature Retrieval using Curvelet Transforms," vol. 8, no. 8, Apr. 2011, doi: 10.48550/arXiv.1104.1945.

[20] J. Coetzer, B. M. Herbst, and J. A. Du Preez, "Offline signature verification using the discrete radon transform and a hidden Markov model," *EURASIP J. Appl. Signal Processing*, vol. 2004, no. 4, pp. 559–571, 2004, doi: 10.1155/S1110865704309042.

[21] Jahandad, S. M. Sam, K. Kamardin, N. N. Amir Sjarif, and N. Mohamed, "Offline signature verification using deep learning convolutional Neural network (CNN) architectures GoogLeNet inception-v1 and inception-v3," *Procedia Comput. Sci.*, vol. 161, pp. 475–483, 2019, doi: 10.1016/j.procs.2019.11.147.

[22] S. M. A. Navid, S. H. Priya, N. H. Khandakar, Z. Ferdous, and A. B. Haque, "Signature Verification Using Convolutional Neural Network," in *2019 IEEE International Conference on Robotics, Automation, Artificial-Intelligence and Internet-of-Things, RAAICON 2019*, 2019, pp. 35–39. doi: 10.1109/RAAICON48939.2019.19.

[23] M. Okawa, "From BoVW to VLAD with KAZE features: Offline signature verification considering cognitive processes of forensic experts," *Pattern Recognit. Lett.*, vol. 113, pp. 75–82, 2018, doi: 10.1016/j.patrec.2018.05.019.

[24] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, and J. Ortega-Garcia, "DeepSign: Deep on-line signature verification," *arXiv*, pp. 1–11, 2020, doi: 10.1109/tbiom.2021.3054533.

[25] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, and J. Ortega-Garcia,

"BioTouchPass2: Touchscreen Password Biometrics Using Time-Aligned Recurrent Neural Networks," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 2616–2628, 2020, doi: 10.1109/TIFS.2020.2973832.

[26] D. Tsourounis, I. Theodorakopoulos, E. N. Zois, and G. Economou, "From text to signatures: Knowledge transfer for efficient deep feature learning in offline signature verification," *Expert Syst. Appl.*, vol. 189, no. January 2021, p. 116136, 2022, doi: 10.1016/j.eswa.2021.116136.

[27] O. Mersa, F. Etaati, S. Masoudnia, and B. N. Araabi, "Learning Representations from Persian Handwriting for Offline Signature Verification, a Deep Transfer Learning Approach," *4th Int. Conf. Pattern Recognit. Image Anal. IPRIA 2019*, pp. 268–273, 2019, doi: 10.1109/PRIA.2019.8785979.

[28] M. A. Ferrer, M. Diaz-Cabrera, and A. Morales, "Static signature synthesis: A neuromotor inspired approach for biometrics," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 667–680, 2015, doi: 10.1109/TPAMI.2014.2343981.

[29] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Learning features for offline handwritten signature verification using deep convolutional neural networks," *Pattern Recognit.*, vol. 70, pp. 163–176, 2017, doi: 10.1016/j.patcog.2017.05.012.

[30] C. Ishikawa, J. A. U. Marasigan, and M. V. C. Caya, "Cloud-based Signature Validation Using CNN Inception-ResNet Architecture," *2020 IEEE 12th Int. Conf. Humanoid, Nanotechnology, Inf. Technol. Commun. Control. Environ. Manag. HNICEM 2020*, pp. 5–10, 2020, doi: 10.1109/HNICEM51456.2020.9400027.

[31] P. Maergner *et al.*, "Offline signature verification by combining graph edit distance and triplet networks," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11004 LNCS, pp. 470–480, 2018, doi: 10.1007/978-3-319-97785-0_45.

[32] M. A. Ferrer, M. Diaz-Cabrera, and A. Morales, "Static signature synthesis: A neuromotor inspired approach for biometrics," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 667–680, 2015, doi: 10.1109/TPAMI.2014.2343981.

[33] J. Vargas, F. and Ferrer, M. and Travieso, C. and Alonso, "Off-line Handwritten

Signature GPDS-960 Corpus," *Ninth Int. Conf. Doc. Anal. Recognit. (ICDAR 2007)*, vol. 2, pp. 764–768, 2007, doi: 10.1109/ICDAR.2007.4377018.

[34] L. E. Martínez, C. M. Travieso, J. B. Alonso, and M. A. Ferrer, "Parameterization of a forgery handwritten signature verification system using SVM," *Proc. - Int. Carnahan Conf. Secur. Technol.*, pp. 193–196, 2004, doi: 10.1109/ccst.2004.1405391.

[35] J. Ortega-Garcia *et al.*, "MCYT baseline corpus: a bimodal biometric database," *IEE Proc. - Vision, Image, Signal Process.*, vol. 150, no. 6, p. 395, Jan. 2003, doi: 10.1049/ip-vis:20031078.

[36] A. Soleimani, K. Fouladi, and B. N. Araabi, "UTSig: A Persian offline signature dataset," *IET Biometrics*, vol. 6, no. 1, pp. 1–8, 2017, doi: 10.1049/iet-bmt.2015.0058.

[37] S. Pal, A. Alaei, U. Pal, and M. Blumenstein, "Performance of an Off-Line Signature Verification Method Based on Texture Features on a Large Indic-Script Signature Dataset," *Proc. - 12th IAPR Int. Work. Doc. Anal. Syst. DAS 2016*, pp. 72–77, 2016, doi: 10.1109/DAS.2016.48.

[38] M. Liwicki *et al.*, "Signature Verification Competition for Online and Offline Skilled Forgeries (SigComp2011)," in *2011 International Conference on Document Analysis and Recognition*, 2011, pp. 1480–1484. doi: 10.1109/ICDAR.2011.294.

[39] M. M. Hameed, R. Ahmad, M. L. M. Kiah, and G. Murtaza, "Machine learning-based offline signature verification systems: A systematic review," *Signal Process. Image Commun.*, vol. 93, no. October 2020, p. 116139, 2021, doi: 10.1016/j.image.2021.116139.

[40] M. Sharif, M. A. Khan, M. Faisal, M. Yasmin, and S. L. Fernandes, "A framework for offline signature verification system: Best features selection approach," *Pattern Recognit. Lett.*, vol. 139, pp. 50–59, 2020, doi: 10.1016/j.patrec.2018.01.021.

[41] F. E. Batool *et al.*, "Offline signature verification system: a novel technique of fusion of GLCM and geometric features using SVM," *Multimed. Tools Appl.*,

2020, doi: 10.1007/s11042-020-08851-4.

[42] C. V. Aravinda, L. Meng, and K. R. Uday Kumar Reddy, “An approach for signature recognition using contours based technique,” *Int. Conf. Adv. Mechatron. Syst. ICAMechS*, vol. 2019-Augus, pp. 46–51, 2019, doi: 10.1109/ICAMechS.2019.8861516.

[43] B. Fang, C. H. Leung, Y. Y. Tang, K. W. Tse, P. C. K. Kwok, and Y. K. Wong, “Off-line signature verification by the tracking of feature and stroke positions,” *Pattern Recognit.*, vol. 36, no. 1, pp. 91–101, 2003, doi: 10.1016/S0031-3203(02)00061-4.

[44] J. Wen, B. Fang, Y. Y. Tang, and T. P. Zhang, “Model-based signature verification with rotation invariant features,” *Pattern Recognit.*, vol. 42, no. 7, pp. 1458–1466, 2009, doi: 10.1016/j.patcog.2008.10.006.

[45] P. Singh, P. Verma, and N. Singh, “Offline Signature Verification: An Application of GLCM Features in Machine Learning,” *Ann. Data Sci.*, vol. 9, no. 6, pp. 1309–1321, Dec. 2022, doi: 10.1007/s40745-021-00343-y.

[46] W. Bouamra, C. Djeddi, B. Nini, M. Diaz, and I. Siddiqi, “Towards the design of an offline signature verifier based on a small number of genuine samples for training,” *Expert Syst. Appl.*, vol. 107, pp. 182–195, 2018, doi: 10.1016/j.eswa.2018.04.035.

[47] L. S. Oliveira, E. Justino, C. Freitas, and R. Sabourin, “The Graphology Applied to Signature Verification,” *Proc. 12th Bienn. Conf. Int. Graphonomics Soc.*, pp. 286–290, 2005, [Online]. Available: http://www.inf.ufpr.br/lesoliveira/download/IGS2005.pdf

[48] E. Alajrami *et al.*, “Handwritten Signature Verification using Deep Learning,” *Int. J. Acad. Multidiscip. Res.*, vol. 3, no. 12, pp. 39–44, 2019, [Online]. Available: www.ijeais.org/ijamr

[49] M. M. Yapici, A. Tekerek, and N. Topaloglu, “Convolutional Neural Network Based Offline Signature Verification Application,” *Int. Congr. Big Data, Deep Learn. Fight. Cyber Terror. IBIGDELFT 2018 - Proc.*, pp. 30–34, 2019, doi:

10.1109/IBIGDELFT.2018.8625290.

[50] J. Poddar, V. Parikh, and S. K. Bharti, “Offline Signature Recognition and Forgery Detection using Deep Learning,” *Procedia Comput. Sci.*, vol. 170, no. 2019, pp. 610–617, 2020, doi: 10.1016/j.procs.2020.03.133.

[51] H. Rantzsch, H. Yang, and C. Meinel, “Signature embedding: Writer independent offline signature verification with deep metric learning,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10073 LNCS, pp. 616–625, 2016, doi: 10.1007/978-3-319-50832-0_60.

[52] E. Parcham, M. Ilbeygi, and M. Amini, “CBCapsNet: A novel writer-independent offline signature verification model using a CNN-based architecture and capsule neural networks,” *Expert Syst. Appl.*, vol. 185, no. July, p. 115649, 2021, doi: 10.1016/j.eswa.2021.115649.

[53] S. Wang and S. Jia, “Signature handwriting identification based on generative adversarial networks,” *J. Phys. Conf. Ser.*, vol. 1187, no. 4, 2019, doi: 10.1088/1742-6596/1187/4/042047.

[54] V. Sangeetha and K. J. R. Prasad, “Syntheses of novel derivatives of 2-acetylfuro[2,3-a]carbazoles, benzo[1,2-b]-1,4-thiazepino[2,3-a]carbazoles and 1-acetyloxycarbazole-2- carbaldehydes,” *Indian J. Chem. - Sect. B Org. Med. Chem.*, vol. 45, no. 8, pp. 1951–1954, 2006, doi: 10.1002/chin.200650130.

[55] T. F. Gonzalez, “Handbook of approximation algorithms and metaheuristics,” *Handb. Approx. Algorithms Metaheuristics*, pp. 1–1432, 2007, doi: 10.1201/9781420010749.

[56] C. Szegedy *et al.*, “Going deeper with convolutions,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, vol. 07-12-June, pp. 1–9. doi: 10.1109/CVPR.2015.7298594.

[57] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, Sep. 2015, doi: 10.48550/arXiv.1409.1556.

[58] A. G. Howard *et al.*, “MobileNets: Efficient Convolutional Neural Networks for

Mobile Vision Applications," 2017, doi: 10.48550/arXiv.1704.04861.

[59] M. Jampour, S. Abbaasi, and M. Javidi, "CapsNet regularization and its conjugation with ResNet for signature identification," *Pattern Recognit.*, vol. 120, p. 107851, 2021, doi: 10.1016/j.patcog.2021.107851.

[60] D. Engin, A. Kantarci, S. Arslan, and H. K. Ekenel, "Offline signature verification on real-world documents," *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, vol. 2020-June, pp. 3518–3526, 2020, doi: 10.1109/CVPRW50498.2020.00412.

[61] S. Masoudnia, O. Mersa, B. N. Araabi, A. H. Vahabie, M. A. Sadeghi, and M. N. Ahmadabadi, "Multi-representational learning for Offline Signature Verification using Multi-Loss Snapshot Ensemble of CNNs," *Expert Syst. Appl.*, vol. 133, pp. 317–330, 2019, doi: 10.1016/j.eswa.2019.03.040.

[62] T. Younesian, S. Masoudnia, R. Hosseini, and B. N. Araabi, "Active Transfer Learning for Persian Offline Signature Verification," *4th Int. Conf. Pattern Recognit. Image Anal. IPRIA 2019*, pp. 234–239, 2019, doi: 10.1109/PRIA.2019.8786013.

[63] T. V. Nguyen and I. Paik, "Feature extraction with triplet loss to classify disease on leaf data," *2020 11th Int. Conf. Aware. Sci. Technol. iCAST 2020*, pp. 2–6, 2020, doi: 10.1109/iCAST51195.2020.9319494.

[64] S. Pal, U. Pal, and M. Blumenstein, "Off-line verification technique for Hindi signatures," *IET Biometrics*, vol. 2, no. 4, pp. 182–190, 2013, doi: 10.1049/iet-bmt.2013.0016.

[65] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9370, no. 1271, pp. 84–92, 2015, doi: 10.1007/978-3-319-24261-3_7.

[66] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 815–823, 2015, doi:

10.1109/CVPR.2015.7298682.

[67] Z. Jin, J. Shang, Q. Zhu, C. Ling, W. Xie, and B. Qiang, “RFRSF: Employee Turnover Prediction Based on Random Forests and Survival Analysis,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12343 LNCS, pp. 503–515, 2020, doi: 10.1007/978-3-030-62008-0_35.

[68] D. Avola, M. J. Bigdello, L. Cinque, A. Fagioli, and M. R. Marini, “R-SigNet: Reduced space writer-independent feature learning for offline writer-dependent signature verification,” *Pattern Recognit. Lett.*, vol. 150, pp. 189–196, 2021, doi: 10.1016/j.patrec.2021.06.033.

[69] A. Dutta, U. Pal, and J. Llados, “Compact correlated features for writer independent signature verification,” in *Proceedings - International Conference on Pattern Recognition*, 2016, vol. 0, pp. 3422–3427. doi: 10.1109/ICPR.2016.7900163.

[70] S. Dey, A. Dutta, J. I. Toledo, S. K. Ghosh, J. Llados, and U. Pal, “SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification,” no. 1, pp. 1–7, Jul. 2017, doi: 10.48550/arXiv:1707.02131.

[71] S. Pal, A. Alaei, U. Pal, and M. Blumenstein, “Performance of an Off-Line Signature Verification Method Based on Texture Features on a Large Indic-Script Signature Dataset,” in *Proceedings - 12th IAPR International Workshop on Document Analysis Systems, DAS 2016*, 2016, pp. 72–77. doi: 10.1109/DAS.2016.48.

[72] P. Maergner *et al.*, “Combining graph edit distance and triplet networks for offline signature verification,” *Pattern Recognit. Lett.*, vol. 125, pp. 527–533, 2019, doi: 10.1016/j.patrec.2019.06.024.

APPENDIX A

Appendix A consist of the proposed model summary with respect to layers, output shape and number of parameters at each layer.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 1)]	0
conv2d (Conv2D)	(None, 112, 112, 32)	288
conv1_bn (BatchNormalization)	(None, 112, 112, 32)	128
conv1_relu (ReLU)	(None, 112, 112, 32)	0
block_1_depthwise_conv (DepthwiseConv2D)	(None, 112, 112, 32)	288
block_1_dw_bn (BatchNormalization)	(None, 112, 112, 32)	128
block_1_dw_relu (ReLU)	(None, 112, 112, 32)	0
block_1_compress (Conv2D)	(None, 112, 112, 16)	512
block_1_compress_bn (BatchNormalization)	(None, 112, 112, 16)	64
block_2_expand (Conv2D)	(None, 112, 112, 96)	1536
block_2_expand_bn (BatchNormalization)	(None, 112, 112, 96)	384
block_2_expand_relu (ReLU)	(None, 112, 112, 96)	0
block_2_depthwise_conv (DepthwiseConv2D)	(None, 56, 56, 96)	864
block_2_dw_bn (BatchNormalization)	(None, 56, 56, 96)	384
block_2_dw_relu (ReLU)	(None, 56, 56, 96)	0
block_2_compress (Conv2D)	(None, 56, 56, 24)	2304
block_2_compress_bn (BatchNor alization)	(None, 56, 56, 24)	96
block_3_expand (Conv2D)	(None, 56, 56, 144)	3456
block_3_expand_bn (BatchNormalization)	(None, 56, 56, 144)	576
block_3_expand_relu (ReLU)	(None, 56, 56, 144)	0
block_3_depthwise_conv (DepthwiseConv2D)	(None, 56, 56, 144)	1296
block_3_dw_bn (BatchNormalization)	(None, 56, 56, 144)	576
block_3_dw_relu (ReLU)	(None, 56, 56, 144)	0
block_3_compress (Conv2D)	(None, 56, 56, 24)	3456
block_3_compress_bn (BatchNormalization)	(None, 56, 56, 24)	96
add (Add)	(None, 56, 56, 24)	0
block_4_expand (Conv2D)	(None, 56, 56, 144)	3456
block_4_expand_bn (BatchNormalization)	(None, 56, 56, 144)	576
block_4_expand_relu (ReLU)	(None, 56, 56, 144)	0
block_4_depthwise_conv (DepthwiseConv2D)	(None, 28, 28, 144)	1296
block_4_dw_bn (BatchNormalization)	(None, 28, 28, 144)	576

block_4_dw_relu (ReLU)	(None, 28, 28, 144)	0
block_4_compress (Conv2D)	(None, 28, 28, 32)	4608
block_4_compress_bn (BatchNormalization)	(None, 28, 28, 32)	128
block_5_expand (Conv2D)	(None, 28, 28, 192)	6144
block_5_expand_bn (BatchNormalization)	(None, 28, 28, 192)	768
block_5_expand_relu (ReLU)	(None, 28, 28, 192)	0
block_5_depthwise_conv (DepthwiseConv2D)	(None, 28, 28, 192)	1728
block_5_dw_bn (BatchNormalization)	(None, 28, 28, 192)	768
block_5_dw_relu (ReLU)	(None, 28, 28, 192)	0
block_5_compress (Conv2D)	(None, 28, 28, 32)	6144
block_5_compress_bn (BatchNormalization)	(None, 28, 28, 32)	128
add_1 (Add)	(None, 28, 28, 32)	0
block_6_expand (Conv2D)	(None, 28, 28, 192)	6144
block_6_expand_bn (BatchNormalization)	(None, 28, 28, 192)	768
block_6_expand_relu (ReLU)	(None, 28, 28, 192)	0
block_6_depthwise_conv (DepthwiseConv2D)	(None, 28, 28, 192)	1728
block_6_dw_bn (BatchNormalization)	(None, 28, 28, 192)	768
block_6_dw_relu (ReLU)	(None, 28, 28, 192)	0
block_6_compress (Conv2D)	(None, 28, 28, 32)	6144
block_6_compress_bn (BatchNormalization)	(None, 28, 28, 32)	128
add_2 (Add)	(None, 28, 28, 32)	0
block_7_expand (Conv2D)	(None, 28, 28, 192)	6144
block_7_expand_bn (BatchNormalization)	(None, 28, 28, 192)	768
block_7_expand_relu (ReLU)	(None, 28, 28, 192)	0
block_7_depthwise_conv (DepthwiseConv2D)	(None, 14, 14, 192)	1728
block_7_dw_bn (BatchNormalization)	(None, 14, 14, 192)	768
block_7_dw_relu (ReLU)	(None, 14, 14, 192)	0
block_7_compress (Conv2D)	(None, 14, 14, 64)	12288
block_7_compress_bn (BatchNormalization)	(None, 14, 14, 64)	256
block_8_expand (Conv2D)	(None, 14, 14, 384)	24576
block_8_expand_bn (BatchNormalization)	(None, 14, 14, 384)	1536
block_8_expand_relu (ReLU)	(None, 14, 14, 384)	0
block_8_depthwise_conv (DepthwiseConv2D)	(None, 14, 14, 384)	3456
block_8_dw_bn (BatchNormalization)	(None, 14, 14, 384)	1536
block_8_dw_relu (ReLU)	(None, 14, 14, 384)	0
block_8_compress (Conv2D)	(None, 14, 14, 64)	24576
block_8_compress_bn (BatchNormalization)	(None, 14, 14, 64)	256
add_3 (Add)	(None, 14, 14, 64)	0
block_9_expand (Conv2D)	(None, 14, 14, 384)	24576
block_9_expand_bn (BatchNormalization)	(None, 14, 14, 384)	1536
block_9_expand_relu (ReLU)	(None, 14, 14, 384)	0
block_9_depthwise_conv (DepthwiseConv2D)	(None, 14, 14, 384)	3456
block_9_dw_bn (BatchNormalization)	(None, 14, 14, 384)	1536
block_9_dw_relu (ReLU)	(None, 14, 14, 384)	0
block_9_compress (Conv2D)	(None, 14, 14, 64)	24576

block_9_compress_bn (BatchNormalization)	(None, 14, 14, 64)	256
add_4 (Add)	(None, 14, 14, 64)	0
block_10_expand (Conv2D)	(None, 14, 14, 384)	24576
block_10_expand_bn (BatchNormalization)	(None, 14, 14, 384)	1536
block_10_expand_relu (ReLU)	(None, 14, 14, 384)	0
block_10_depthwise_conv (DepthwiseConv2D)	(None, 14, 14, 384)	3456
block_10_dw_bn (BatchNormalization)	(None, 14, 14, 384)	1536
block_10_dw_relu (ReLU)	(None, 14, 14, 384)	0
block_10_compress (Conv2D)	(None, 14, 14, 64)	24576
block_10_compress_bn (BatchNormalization)	(None, 14, 14, 64)	256
add_5 (Add)	(None, 14, 14, 64)	0
block_11_expand (Conv2D)	(None, 14, 14, 384)	24576
block_11_expand_bn (BatchNormalization)	(None, 14, 14, 384)	1536
block_11_expand_relu (ReLU)	(None, 14, 14, 384)	0
block_11_depthwise_conv (DepthwiseConv2D)	(None, 14, 14, 384)	3456
block_11_dw_bn (BatchNormalization)	(None, 14, 14, 384)	1536
block_11_dw_relu (ReLU)	(None, 14, 14, 384)	0
block_11_compress (Conv2D)	(None, 14, 14, 96)	36864
block_11_compress_bn (BatchNormalization)	(None, 14, 14, 96)	384
block_12_expand (Conv2D)	(None, 14, 14, 576)	55296
block_12_expand_bn (BatchNormalization)	(None, 14, 14, 576)	2304
block_12_expand_relu (ReLU)	(None, 14, 14, 576)	0
block_12_depthwise_conv (DepthwiseConv2D)	(None, 14, 14, 576)	5184
block_12_dw_bn (BatchNormalization)	(None, 14, 14, 576)	2304
block_12_dw_relu (ReLU)	(None, 14, 14, 576)	0
block_12_compress (Conv2D)	(None, 14, 14, 96)	55296
block_12_compress_bn (BatchNormalization)	(None, 14, 14, 96)	384
add_6 (Add)	(None, 14, 14, 96)	0
block_13_expand (Conv2D)	(None, 14, 14, 576)	55296
block_13_expand_bn (BatchNormalization)	(None, 14, 14, 576)	2304
block_13_expand_relu (ReLU)	(None, 14, 14, 576)	0
block_13_depthwise_conv (DepthwiseConv2D)	(None, 14, 14, 576)	5184
block_13_dw_bn (BatchNormalization)	(None, 14, 14, 576)	2304
block_13_dw_relu (ReLU)	(None, 14, 14, 576)	0
block_13_compress (Conv2D)	(None, 14, 14, 96)	55296
block_13_compress_bn (BatchNormalization)	(None, 14, 14, 96)	384
add_7 (Add)	(None, 14, 14, 96)	0
block_14_expand (Conv2D)	(None, 14, 14, 576)	55296
block_14_expand_bn (BatchNormalization)	(None, 14, 14, 576)	2304
block_14_expand_relu (ReLU)	(None, 14, 14, 576)	0
block_14_depthwise_conv (DepthwiseConv2D)	(None, 7, 7, 576)	5184
block_14_dw_bn (BatchNormalization)	(None, 7, 7, 576)	2304
block_14_dw_relu (ReLU)	(None, 7, 7, 576)	0
block_14_compress (Conv2D)	(None, 7, 7, 160)	92160

block_14_compress_bn (BatchNormalization)	(None, 7, 7, 160)	640
block_15_expand (Conv2D)	(None, 7, 7, 960)	153600
block_15_expand_bn (BatchNormalization)	(None, 7, 7, 960)	3840
block_15_expand_relu (ReLU)	(None, 7, 7, 960)	0
block_15_depthwise_conv (DepthwiseConv2D)	(None, 7, 7, 960)	8640
block_15_dw_bn (BatchNormalization)	(None, 7, 7, 960)	3840
block_15_dw_relu (ReLU)	(None, 7, 7, 960)	0
block_15_compress (Conv2D)	(None, 7, 7, 160)	153600
block_15_compress_bn (BatchNormalization)	(None, 7, 7, 160)	640
add_8 (Add)	(None, 7, 7, 160)	0
block_16_expand (Conv2D)	(None, 7, 7, 960)	153600
block_16_expand_bn (BatchNormalization)	(None, 7, 7, 960)	3840
block_16_expand_relu (ReLU)	(None, 7, 7, 960)	0
block_16_depthwise_conv (DepthwiseConv2D)	(None, 7, 7, 960)	8640
block_16_dw_bn (BatchNormalization)	(None, 7, 7, 960)	3840
block_16_dw_relu (ReLU)	(None, 7, 7, 960)	0
block_16_compress (Conv2D)	(None, 7, 7, 160)	153600
block_16_compress_bn (BatchNormalization)	(None, 7, 7, 160)	640
add_9 (Add)	(None, 7, 7, 160)	0
block_17_expand (Conv2D)	(None, 7, 7, 960)	153600
block_17_expand_bn (BatchNormalization)	(None, 7, 7, 960)	3840
block_17_expand_relu (ReLU)	(None, 7, 7, 960)	0
block_17_depthwise_conv (DepthwiseConv2D)	(None, 7, 7, 960)	8640
block_17_dw_bn (BatchNormalization)	(None, 7, 7, 960)	3840
block_17_dw_relu (ReLU)	(None, 7, 7, 960)	0
block_17_compress (Conv2D)	(None, 7, 7, 320)	307200
block_17_compress_bn (BatchNormalization)	(None, 7, 7, 320)	1280
last_conv (Conv2D)	(None, 7, 7, 12)	3840
last_bn (BatchNormalization)	(None, 7, 7, 12)	48
last_relu (ReLU)	(None, 7, 7, 12)	0
global_average_pool (GlobalAveragePooling2D)	(None, 12)	0
dense (Dense)	(None, 100)	1300

Total params: 1,968,644

Trainable params: 1,934,572

Non-trainable params: 34,072