

A Real-time Signature Verification System using Deep Learning

by

**Khadiza Tul Kubra Heme (181014031)
Ashiqur Rahman Prince (191014054)
Mahfuzur Rahman (193014011)
Sanjib Biswas (193014012)**

*Capstone Project Report (CSE499) submitted in partial fulfillment of the
requirements for the degree of*

Bachelor of Science in Computer Science and Engineering

Under the supervision of

Suravi Akhter, Lecturer



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF LIBERAL ARTS BANGLADESH**

FALL 2024

© *Khadiza Tul Kubra Heme, Ashiqur Rahman Prince , Mahfuzur Rahman ,Sanjib Biswas*
All rights reserved

DECLARATION

Project Title A Real-time Signature Verification System using Deep Learning
Authors *Khadiza Tul Kubra Heme, Ashiqur Rahman Prince, Mahfuzur Rahman*
, and Sanjib Biswas
Student IDs 181014031, 191014054, 193014011, and 193014012
Supervisor Suravi Akhter, Lecturer

We declare that this capstone project report entitled *A Real-time Signature Verification System using Deep Learning* is the result of our own work except as cited in the references. The capstone project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Khadiza Tul Kubra Heme
181014031

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Ashiqur Rahman Prince
191014054

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Mahfuzur Rahman
193014011

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Sanjib Biswas
193014012

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Date: January 30, 2025



Department of Computer Science and Engineering
University of Liberal Arts Bangladesh
Mohammadpur, Dhaka - 1207

CERTIFICATE

This is to certify that the capstone project report entitled **A Real-time Signature Verification System using Deep Learning**, submitted by **Khadiza Tul Kubra Heme** (181014031), **Ashiqur Rahman Prince** (191014054), **Mahfuzur Rahman** (193014011), **Sanjib Biswas** (193014012) is a undergraduate students of the **Department of Computer Science and Engineering** has been examined. Upon recommendation by the examination committee, we hereby accord our approval of it as the presented work and submitted report fulfill the requirements for its acceptance in partial fulfillment for the degree of *Bachelor of Science in Computer Science and Engineering*.

Suravi Akhter , Lecturer

Dept. of CSE, University of Liberal Arts Bangladesh

Prof. Muhammad Golam Kibria, PhD, SMIEEE

Dept. of CSE, University of Liberal Arts Bangladesh

Raihan Kibria, Senior Lecturer

Dept. of CSE, University of Liberal Arts Bangladesh

Rubaiya Hafiz , Lecturer

Dept. of CSE, University of Liberal Arts Bangladesh

Place: Dhaka

Date: January 30, 2025

ACKNOWLEDGEMENTS

We are most grateful to our research supervisor *Suravi Akhter, Lecturer*, for providing us the opportunity to perform research under him. His incredible leadership, wise feedback, and steadfast support have been a major driver of the work we do. His commitment and encouragement taught us to aim high and tackle our research with concise clarity and confidence. It has been a pleasure to learn from him and be mentored by him throughout my time.

We owe our special thanks to the honorable faculty members of the Department of Computer Science and Engineering at the University of Liberal Arts Bangladesh. They have helped shape our lives through their teaching, guidance, and encouragement.

To our parents, thank you for your unconditional love and support without which, this would not be possible. A big thank you to our friends and family as well for their perpetual support and encouragement on this journey.

Most importantly, we would like to thank all of the people who have helped us learn and succeed in this pursuit.

Khadiza Tul Kubra Heme, Ashiqur Rahman Prince, Mahfuzur Rahman, Sanjib Biswas

University of Liberal Arts Bangladesh

Date: January 30, 2025

Dedicated to
To my beloved parents
– *Khadiza Tul Kubra Heme*

Dedicated to
To my beloved Father
– *Ashiqur Rahman Prince*

Dedicated to
To my beloved parents
– *Mahfuzur Rahman*

Dedicated to
*"This work is lovingly dedicated to
my grandparents. Their love and
guidance continue to inspire me
every day."*
– *Sanjib Biswas*

ABSTRACT

Handwritten authentication verification is a growing concern in legal transactions, and with each passing day, confirmation and approval are being carried out with handwritten signatures. Offline signature verification is slow and inefficient, especially when dealing with big documents and files. This paper proposes a method for improving signature verification in Image Processing using Convolution Neural Networks, which is crucial in preventing document forgery and falsification. The AVN model mines effective features actively by varying the existing data and generating new data. Four challenging signature datasets of different languages were used to test this method to show its strength and potential. In the hybrid approach, features are extracted from signature images through CNN, HOG, and Decision Trees perform feature selection. The experimental results obtained show satisfactory efficiency and predictive ability, especially for skilled forged signatures.

Keywords: Real-time Signature Verification, Convolutional Neural Network(CNN)

Contents

1	Introduction	1
1.1	Background Study	1
1.2	Problem statement	2
1.3	Aims, objectives and Motivation	2
1.3.1	Aims	2
1.3.2	Objective	2
1.3.3	Motivation	3
1.4	Project Specification	3
1.5	Project Planning	4
1.6	Report Layout	5
2	Literature Review	7
2.1	Introduction	7
2.2	Related Works	7
2.2.1	Signature verification	10
2.2.2	Image Processing	11
2.2.3	Deep Learning	11
2.3	Comparative Analysis	11
2.4	Literature Review Table:	13
3	Methodology	17
3.1	Data Collection	18
3.1.1	Data processing:	18
3.1.2	Data collection form:	18
3.1.3	Original signature form:	22
3.1.4	Forgery signature form:	22

3.2	Preprocessing	22
3.2.1	Image Resizing	22
3.2.2	Binarization	22
3.2.3	Data split	23
3.3	Implementation	23
3.4	Model Selection and Architecture	23
3.4.1	Choice of Model:	23
3.4.2	Architecture Design:	23
3.5	Training the Model	23
3.5.1	Training Process:	23
3.5.2	Loss and Accuracy Monitoring:	24
3.6	Validation and Testing	24
3.6.1	Validation:	24
3.6.2	Testing:	24
3.7	Model Evaluation	24
3.7.1	Performance Metrics:	24
3.8	Model Development	24
3.8.1	CNN Architecture:	24
3.9	Model Methodology	25
3.9.1	Data Collection	26
3.9.2	Information Gathering	26
3.9.3	Preprocessing Images	26
3.9.4	DATA LABELING (SUPERVISED LEARNING METHOD) . .	26
3.9.4.1	DEFINITION:	26
3.9.5	WHY IT IS IMPORTANT:	27
3.9.5.1	SUPERVISED LEARNING DEPENDENCY:	27
3.9.6	PATTERN RECOGNITION:	27
3.9.7	PERFORMANCE EVALUATION:	27
3.9.8	AUTOMATION IN REAL-WORLD APPLICATIONS:	27
3.9.9	LABELING TECHNIQUES USED:	27
3.9.10	CHALLENGES IN DATA LABELING:	28
3.9.11	Data Splitting (Train-Test Splitting)	28
3.9.11.1	Definition:	28
3.9.12	Training Set (80%):	28

3.9.13	Test Set (20%):	28
3.9.14	Generalization of the Model:	28
3.9.15	Overfitting Avoidance:	28
3.9.16	Performance Benchmarking:	29
3.9.17	Expected Data Splitting Challenges:	29
3.9.18	Best Practices	29
3.9.19	Data Augmentation	29
3.9.19.1	Definition:	29
3.9.20	Why Use Data Augmentation:	30
3.9.21	Simulation of Real-World Conditions:	30
3.9.22	Learning Transfer	31
3.9.23	Extra important layers:	31
3.9.24	Model Training	32
3.9.25	Model Evaluation	32
3.9.26	Accuracy:	32
3.10	Website Development Methodology	33
3.10.1	Load Pre-Trained Model	34
3.10.2	Single Image Input Goal:	34
3.10.3	Preprocessing Images	34
3.10.4	ResNet50 preprocessing:	34
3.10.5	Model Forecasting	35
3.10.6	Probability Threshold:	35
3.10.7	Result with Confidence	35
4	Design and Development	36
4.1	System Overview	36
4.2	Design Architecture	36
4.2.1	Frontend	37
4.2.2	JavaScript:	37
4.2.3	Backend	37
4.2.4	Machine Learning Integration:	38
4.2.5	Image Preprocessing	38
4.3	Development Details	39
4.3.1	Environment Setup	39

4.3.2	Forgery Application Workflow	39
4.3.3	Model Prediction	40
4.3.4	User Interface (UI) Design	40
4.3.5	Key Functionalities	40
4.3.6	Challenges and Solutions	41
4.3.7	Future Enhancements	41
5	Result analysis	42
5.1	Comparison table	44
5.2	Bangla Dataset Metrics:	44
5.3	Bangla-English Mixed All Dataset Metrics:	45
5.4	Epoch (Iteration Count)	47
5.5	Interpretation:1	47
5.6	Interpretation:2	48
5.7	Validation Accuracy	48
5.7.1	Interpretation:	48
5.8	Validation Loss	48
5.8.1	Interpretation:	48
5.9	Model Stability:	49
5.10	Epoch (Training Iteration)	50
5.11	Training Accuracy	50
5.11.1	Example values:	50
5.12	Interpretation:	50
5.13	Loss (Training Loss)	51
5.13.1	Example values:	51
5.14	Interpretation:	51
5.14.1	Val Accuracy (Validation Accuracy)	51
5.14.2	Example values:	51
5.15	Interpretation:	52
5.15.1	Val Loss (Validation Loss)	52
5.15.2	Improvement:	52
5.15.3	Consistency:	52
5.16	Final Evaluation:	53
5.17	Result Evaluation	53

5.17.1	Observations are made:	54
5.18	Further interpretations:	54
5.18.1	Definition:	54
5.19	Interpretation:	54
6	Project Management and Financial Activities	56
6.1	Project Overview	56
6.2	Development Tools	56
6.2.1	Programming Languages	56
6.2.2	Deep Learning Frameworks	56
6.2.3	Image Processing and Data Handling	57
6.2.4	Model Training and Evaluation	57
6.2.5	Development Environment	57
6.2.6	Web and API Integration	57
6.3	Financial Activities	58
7	Conclusions	59
7.1	Social, Legal, Ethical, and Environmental Issues	59
7.2	Social	60
7.2.1	Accessibility:	60
7.2.2	Trust and Acceptance:	60
7.2.3	Digital Divide:	60
7.3	Legal	60
7.3.1	Data Protection Laws:	60
7.3.2	Liability:	60
7.3.3	Authentication Standards:	60
7.4	Ethical	61
7.4.1	Bias or Discrimination:	61
7.4.2	Informed Consent:	61
7.4.3	Over-reliance on Technology:	61
7.5	Environmental Issues	61
7.5.1	Energy Consumption:	61
7.5.2	E-Waste:	61
7.5.3	Paperless Systems' Sustainability:	61
7.6	Brief Summary	62

7.7	Future works	62
7.7.1	Improved Feature Extraction	62
7.7.2	Cross-Domain Adaptability	62
7.7.3	Enhanced Forgery Detection	62
7.7.4	Integration of Multimodal Data	63
7.7.5	Explainable AI (XAI) in Verification	63
7.7.6	Scalability and Efficiency	63
7.7.7	Personalized Verification Systems	63
7.7.8	Cross-Cultural Signature Analysis	63
7.7.9	Addressing Ethical and Privacy Concerns	64
7.7.10	Benchmarking and Standardization	64
References		65

List of Figures

1.1	Project Timeline	4
2.1	Research Paper Analysis	12
3.1	Overall workflow diagram of the proposed system.	17
3.2	Original signature collection form	19
3.3	Forged signature collection form	20
3.4	Original signature collection	21
3.5	Forged signature collection	21
3.6	Model Methodology	25
3.7	Website Development Methodology	33
4.1	Signature verification system Website home page	36
4.2	Verification Result page : (Original)	38
4.3	Verification Result page : (Forgery)	38
5.1	Bangla Training and Validation Accuracy Graph	47
5.2	Bangla and English Training and Validation Accuracy Graph	50
5.3	Result Evaluation Metrics for Bangla Data set	53
5.4	Result Evaluation Metrics for All Data set	53

List of Tables

2.1	Literature Review Table:1	13
2.2	Literature Review Table:2	14
2.3	Literature Review Table:3	15
2.4	Literature Review Table:4	16
5.1	Model Training and Evaluation Results	42
5.2	Model Training and Evaluation Results for Bangla	44
5.3	Model Training and Evaluation Results for All Data	45
6.1	Financial Activities chart	58

Chapter 1

Introduction

1.1 Background Study

Signature verification and forgery detection refer to the processes of automatically verifying signatures to determine if they are authentic [Hameed et al. \(2023\)](#). Basically, there are two types: static and dynamic. Static verification is done after a document has been created, while dynamic verification occurs when a person creates their signature on a digital device. Handwritten signatures are one of the widely accepted personal attributes for verification in banking and business [Li et al. \(2021\)](#).

The verification of a signature offline is done by [Batool et al. \(2024\)](#) static features, such as image processing techniques, which test the accuracy of the signature. Initial identifications through passwords also come under this category. Other multimodal systems strictly authenticate a person using biometric features. Biometric systems have been in application for identifying individuals either from their behavioral or physiological characteristics, and handwritten signature has been one of the most usable tools in biometrics for secure personal authentication [Poddar et al. \(2020\)](#).

Signature verification has been widely studied over the past two decades and remains a very promising research area. Previous works have traditionally divided it into two categories: online and offline. Online signature verification relies on dynamic information captured in the process of writing, while offline signature verification is based on static images of signatures and, in general, is more challenging and thus less accurate [Kao & Wen \(2020\)](#) [Hameed et al. \(2021\)](#).

Automation of processes through artificial intelligence, machine learning, and cloud computing has increased exponentially in the banking industry. Data science has emerged as an important step that organizations must undertake to keep up with

the fast-moving market scenario. Machine learning algorithms have contributed immensely to data mining, image classification, and business analytics [Banerjee et al. \(2021\)](#) [Fierrez-Aguilar et al. \(2005\)](#). The inefficiency in the physical processes of the cheque-clearing system prompts the need to concentrate research on automated verification for offline written signatures using different machine learning algorithms [Jagtap et al. \(2020\)](#) [Guerra-Segura et al. \(2021\)](#).

Verification of signatures is a major issue in applications related to vision and multimedia, and there is an emerging need to develop intelligent techniques for verifying a large number of signature data [Fierrez-Aguilar et al. \(2005\)](#) worldwide every day. Extraction and selection of features are the most crucial tasks involved in segregating the original and forged signatures. This paper presents an offline signature verification system with enhanced feature extraction using a combination of deep learning and machine learning classifiers [Li et al. \(2021\)](#).

1.2 Problem statement

Issues such as overfitting caused by insufficient and diverse training data limit the scalability and reliability of existing offline signature verification techniques. By extending these algorithms to authenticate both Bengali and English signatures, we hope to increase generalization and accuracy by effectively recognizing and differentiating between genuine and fraudulent cases.

1.3 Aims, objectives and Motivation

1.3.1 Aims

This research aims to deep learning techniques will be used to develop an accurate, efficient offline handwritten signature verification system, enhancing efficiency in legal documents and banking, and ensuring reliable identification of authentic and fake signatures.

1.3.2 Objective

- Convolutional Neural Network Based Forged Signature Detection
- Comparison of different feature extraction methods.

- We will collect a dataset of 4000 handwritten signatures from 400 individuals.
- Features are to be extracted with the Mobile Res Net-50 deep learning model [Kiran et al. \(2021\)](#). Feature Selection techniques will detect the features that play an important role in classification.
- High accuracy signature verification via machine learning classifiers.
- Evaluate The CNN Model Performance Accuracy And Reliability The CNN MODEL

1.3.3 Motivation

Thus signature verification systems should be developed due to the increasing demand for secure authentication in fields such as banking, legal services, etc. Conventional methods, usually dependent on hand analysis [Ozyurt et al. \(2024\)](#), require automated solutions for improved accuracy and efficiency. When you consider the core aspects of signature verification systems, it is apparent that advancements in deep learning and computer vision can add great value to another layer of signature verification in ensuring the ultimate security [Malik et al. \(2014\)](#). Signature forgery is exceedingly common in financial and legal transactions that need solid verification means. Since manual detection of forged signatures is a tedious and low-precision task. The issue proves that signature as real-time authentication is widely acceptable, so new methods to verify it is needed [Mitchell et al. \(2023\)](#) [Guerra-Segura et al. \(2021\)](#).

1.4 Project Specification

The project aims to use machine learning techniques, specifically CNNs, for feature extraction and classification of signatures. It uses publicly available datasets like the GPDS300 corpus and CEDAR signature database for training and testing. The system is evaluated based on accuracy, error rate, and computational efficiency. The project involves a four-step approach: dataset and preprocessing, feature extraction using Mobile ResNet-50, feature selection using statistical techniques, and signature verification [Sharma et al. \(2022\)](#). The dataset consists of 400 individuals' signatures, with a standardized image resolution of 224 x 224 pixels. The model is built using

the Keras library with TensorFlow backend. Performance evaluation is conducted to assess the model's accuracy and effectiveness in real-world scenarios [Singh et al. \(2022\)](#) [Alsuhimat & Mohamad \(2023\)](#).

- **Expected Outcomes:** Achieve a high classification accuracy, with the proposed method expected to outperform traditional approaches by utilizing optimized feature selection and advanced classifiers [Tolosana et al. \(2022\)](#).

1.5 Project Planning

A number of clearly defined phases make up the project planning approach for the deep learning-based real-time signature verification system, which guarantees the project's methodical execution and completion. The study phase of the project, which started on April 10, 2024, and focused on a thorough investigation of the topic, is scheduled to end on January 20, 2025, establishing the groundwork for the latter stages of development. A chronology of the project is shown in Figure 1.1.

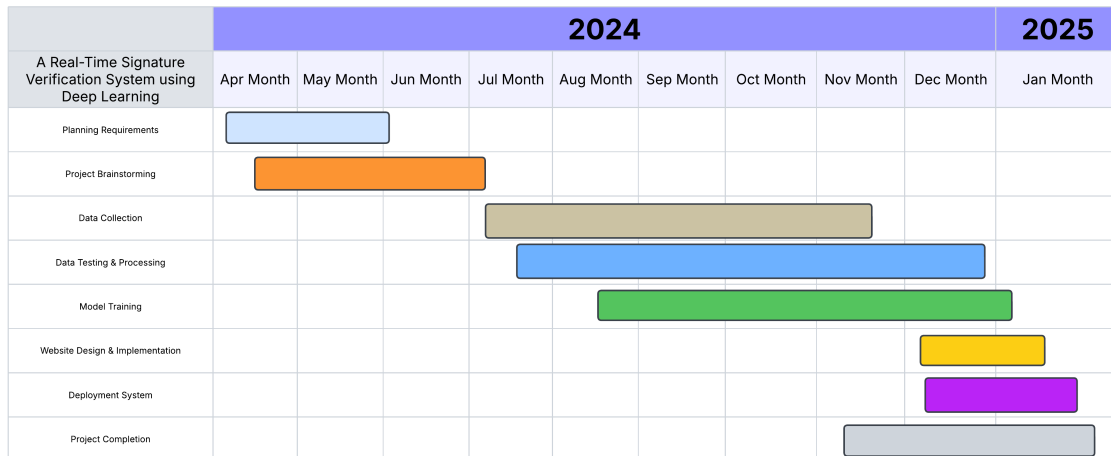


Figure 1.1: Project Timeline

Concurrently, a longer-term literature review was carried out between April 10, 2024, and April 30, 2024. This stage enables a thorough analysis of previous studies, assisting in identifying knowledge gaps and directing the project's course. The requirement analysis phase that followed ran from June 15 to September 15, 2024

during which time the design process was informed by the well-defined system specifications.

The design and planning phase took place from April to July 10, 2024, focusing on creating an architecture for the system that integrates deep learning algorithms for real-time data collection and analysis. Once the design is complete, the project is expected to move into the implementation phase from July 17 to December 2, 2024, during which the system will be built and deployed.

1.6 Report Layout

- **Chapter 1: Introduction**

This chapter introduces the core concept of the project summary, providing an overview of the importance of signature verification. It emphasizes the need for a real-time signature verification system and outlines the problem, objectives, and significance of the study. The research aims to enhance efficiency in handling legal documents and banking processes while ensuring reliable identification of authentic and fake signatures as a solution to challenges associated with signature verification.

- **Chapter 2: Literature Review**

This chapter presents a thorough analysis of relevant literature, including research on real-time signature verification systems. The relevance of a genuine signature, its forging features, and the many deep learning models that may forecast the quality of a signature image are all examined. This establishes the framework for the technical strategies used in the project.

- **Chapter 3: Methodology**

By this chapter verifying that a signature is that of the claimed person, signature verification assures legitimacy. It consists of:

Handwritten Verification: Using automated techniques or manual analysis of style, pressure, and distinctive characteristics. Digital Verification: Ensuring secure identity and data integrity through the use of cryptographic techniques (public/private keys, hashing). Machine Learning: Using feature analysis to improve detection accuracy. It tackles issues like forgery and signature variations and is utilized in digital, legal, and banking systems.

- **Chapter 4: Design and Development**

The project's design and development process is the main topic of this chapter. project management tasks and the work breakdown structure are all covered. Together with the system design, the choice of sensors and communication modules is covered. This chapter also describes how sensor data is gathered and processed, as well as how CNN model devices are integrated.

- **Chapter 5: Result Analysis**

In this chapter, the data gathered by the real-time signature verification system is analyzed. It assesses how well different deep learning models, such as Support Resnet and others, predict signatures. The accuracy and efficacy of several models are compared, and cross-validation and performance are also covered.

- **Chapter 6: Conclusions and Future Work**

The research findings are collected in the last chapter, which also addresses the project's effects on signatures and its genuine and forgery. It draws attention to the project's contributions to concentrating especially on signatures where the system might be enhanced or extended to address new signature challenges. Every chapter expands on the one before it to provide readers with a thorough grasp of how signature verification

Chapter 2

Literature Review

2.1 Introduction

Convolutional neural networks (CNNs) based on the ResNet architecture are used in an offline signature verification system that uses deep learning to differentiate between genuine and forged signatures. The ResNet model uses residual learning to extract high-level characteristics after the system preprocesses scanned signature images. Even with deep networks, ResNet guarantees effective training by reducing the vanishing gradient issue. To learn how to discriminate between counterfeit and real signatures, the model is trained using labelled data. It offers great accuracy and robustness in offline signature verification after being trained to distinguish between counterfeit and legitimate fresh signatures. In this chapter, we review and compare several prominent approaches, focusing on their methodologies, strengths, and limitations [Raigonda et al. \(2024\)](#).

2.2 Related Works

The goal of the study is to assess whether explainable features (manual or traditional) can perform similarly to automatically learned features from DL models, with minimal loss in performance. The research also compares these DL-based features with simpler, traditional methods like Euclidean distance for signature verification [Diaz et al. \(2024\)](#). The primary aim is not to create a specialized DL-based signature verification system but to explore whether deep learning can provide competitive results while maintaining explainability. The results are then compared with more advanced signature verification methods.

The study [Mitchell et al. \(2023\)](#) investigates offline signature verification using deep learning, employing four approaches: a custom CNN trained on imbalanced, balanced, and augmented balanced datasets, and a pre-trained VGG16 model via transfer learning on the augmented balanced dataset. The dataset consisted of 720 images from Kaggle, split into training and test sets. Various data augmentation techniques were applied to enhance model performance. The results showed that the VGG16 model achieved the highest accuracy, highlighting the effectiveness of transfer learning and data augmentation in improving signature verification.

The project focuses on enhancing offline signature verification systems through a novel image augmentation model called OffSig-SinGAN, which utilizes generative adversarial networks (GANs) to generate high-quality augmented signature images. The methodology includes data collection from the GPDSSyntheticSignature dataset, extensive image preprocessing, and the development of the GAN-based model. The augmented images are evaluated using various quality metrics, and the model's performance is compared with six state-of-the-art augmentation methods, demonstrating significant improvements in verification accuracy [Hameed et al. \(2023\)](#).

The discussion highlights the model's effectiveness in addressing the challenge of limited training data for deep learning models in offline signature verification. Future research directions include exploring additional generative models, enhancing image variability, and testing the model on different verification systems to further validate its effectiveness. Overall, the project contributes to advancing deep learning techniques in the field of biometric signature verification.

The research paper investigates the use of Convolutional Neural Networks (CNNs) and Support Vector Machines (SVMs) for offline signature verification. The authors tested various predefined and customized CNN models, ultimately using a simpler architecture with two hidden layers due to computational constraints. While CNN showed good performance on training data, it struggled with generalization to new users' signatures. In contrast, the SVM model achieved 81% accuracy in predicting forged signatures, demonstrating its robustness and effectiveness for writer-independent signature verification [Poddar et al. \(2020\)](#).

The study presents a hybrid method for signature verification that combines Convolutional Neural Networks (CNN) and Histogram of Oriented Gradients (HOG) for feature extraction. The approach utilizes three classifiers: Long Short-Term

Memory (LSTM), Support Vector Machine (SVM), and K-Nearest Neighbor (KNN) to evaluate the effectiveness of the feature extraction. The experimental results demonstrate high accuracy rates on the UTSig and CEDAR datasets, indicating the model's robustness, particularly in distinguishing skilled forged signatures. The key contributions include the development of a flexible feature extraction method that enhances verification precision without needing to adjust parameters for each writer, thus improving the overall performance of the signature verification system [Alsuhimat & Mohamad \(2023\)](#).

The study focuses on offline handwritten signature verification, aiming to distinguish between genuine and forged signatures using advanced techniques in deep learning and machine learning. It utilizes a dataset of 12,600 images from 420 individuals, each providing 30 signatures. The methodology involves feature extraction using the MobileNetV2 model, followed by feature selection through methods like Neighborhood Component Analysis (NCA), Chi2, and Mutual Information (MI). The results demonstrate that the proposed approach can achieve high classification accuracy, with notable improvements when optimal features are selected. The research highlights the importance of effective feature selection in enhancing model performance and reducing complexity in signature verification systems [Ozyurt et al. \(2024\)](#).

The study focuses on enhancing authentic signature verification through the integration of deep learning and machine learning techniques. It employs Convolutional Neural Network (CNN) models, specifically VGG-16 and ResNet-50, for feature extraction from signature images. The research combines these models with machine learning classifiers such as Support Vector Machine (SVM), Artificial Neural Network (ANN), Random Forest, and Extreme Gradient Boosting (XGBoost) to improve accuracy and efficiency in signature verification tasks. Additionally, optimization methods like triplet loss optimization are utilized to enhance model performance. The evaluation is conducted using the ICDAR-2011 and BHSig-260 datasets, demonstrating significant improvements in verification accuracy, particularly with the VGG-16 model [Zhou et al. \(2021\)](#).

A new method for handwritten signature verification is presented in the publication "AVN: An Adversarial Variation Network Model for Handwritten Signature Verification" by Huan Li, Ping Wei, and Ping Hu. The efficacy of traditional approaches may be limited since they frequently passively learn features from existing data. To

improve feature learning, on the other hand, the suggested Adversarial Variation Network (AVN) actively creates new data variations. Three integrated modules make up the AVN model: Handwritten signatures can have their deep, discriminative properties extracted by an extractor. Discriminator: Uses the derived features to inform verification decisions. Variator: To build a more resilient and discriminative model, it actively creates signature variants. Using a min-max loss function, these modules act in an adversarial framework that promotes collaboration and rivalry amongst them. This dynamic improves the model's overall performance, resulting in better [Li et al. \(2021\)](#).

2.2.1 Signature verification

Signature verification is the process of determining the authenticity of a signature by comparing it to a known, genuine signature of the individual. It is widely used for identity verification in legal, financial, and administrative applications [Batool et al. \(2024\)](#). The process can be categorized into two types: Offline Signature Verification and Online Signature Verification

1. Involves verifying signatures captured on paper and digitized as images.
2. Techniques analyze static features such as shape, geometry, texture, and pen pressure (if derivable from the image).
3. Applications include document authentication, bank check verification, and historical signature validation.

Modern Approaches:

- **Traditional Methods:** Feature extraction (e.g., curve analysis, histogram comparison) followed by machine learning classifiers (SVM, k-NN).
- **Deep Learning Methods:** CNNs, especially architectures like ResNet or Siamese networks, extract hierarchical features and directly classify signatures. These methods often outperform traditional approaches in accuracy and generalization.

Signature verification systems aim to balance between minimizing false rejections (genuine signatures marked as forgeries) and false acceptances (forgeries marked as genuine).

2.2.2 Image Processing

In an offline signature verification system, image processing is essential to prepare and analyze signature images for accurate verification.

Key steps include: Image acquisition is the process of taking pictures of signatures using cameras or scanners.

Preprocessing: Using binarization, noise reduction, normalisation, thinning, and greyscale conversion to improve the quality of images. Feature extraction involves the use of texture/geometric analysis to identify both local (like stroke patterns) and global (like size, slant) information.

Segmentation (optional): Dividing the signature into more manageable sections for in-depth examination. Creating changes (such as scaling or rotation) to increase the robustness of the model is known as data augmentation. Classifier Input: supplying CNNs or ResNet, or other deep learning or machine learning models, with processed pictures or data. In order to differentiate authentic signatures from fakes, this procedure guarantees that essential signature characteristics are maintained.

2.2.3 Deep Learning

The researcher [Batool et al. \(2024\)](#) improved the Automatic offline signature verification system by using an analytical method. Without changing the core architecture of current signature systems, this method can be easily included in them. They introduced a novel method that optimizes images of both standard and forgerie systems using deep learning and an algorithmic approach. Real signature systems can be regarded as signatures using various kinds of signature images. Crucially, they stressed that their suggested method is adaptable and may be used for handwriting signatures as well [Wang et al. \(2021\)](#).

2.3 Comparative Analysis

We present a comparative study of the approaches and findings of the examined articles' CNN inference in this section. We provide an overview of each method's strategies, advantages, and outcomes in terms of biological insights and network inference accuracy. Every technique makes a distinct contribution to the field of

Author's	Detection Type	Video Detection	Image Detection	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Accuracy
Poddar et al. (2020)	Signature	X	✓	90%	88%	0.08	0.12	88%
Alajrami et al. (2019)	Signature	X	✓	88%	85%	0.10	0.15	85.5%
Yapıcı et al. (2020)	Signature	X	✓	85%	82%	0.12	0.18	82.7%
Kiran et al. (2021)	Signature	X	✓	83%	80%	0.14	0.20	80.5%
Sharma et al. (2022)	Signature	X	✓	90%	89%	0.08	0.10	89%
Singh et al. (2021)	Signature	X	✓	88%	86%	0.09	0.13	86%
IEEE Xplore (2023)	Signature	X	✓	90%	88%	0.07	0.11	88.6%

Figure 2.1: Research Paper Analysis

CNN inference by tackling certain issues and offering insightful information about biological systems [Raigonda et al. \(2024\)](#). A number of variables, including the type of data, the available computing power, and the particular research goals, influence the method selection.

- **Transfer Learning:** This method makes use of models that have already been trained and have picked up features from large datasets. In order to adjust to the distinct features of the data, the models are refined on the particular signature datasets.
- **Data Augmentation:** To overcome the problem of restricted data availability, methods including rotation, scaling, random cropping, and colour modifications are used to increase the size of the training datasets [Akhundjanov et al. \(2024\)](#). By fine-tuning the pre-trained models' weights on the signature datasets, task-specific features can be learnt by the models, but overfitting can be avoided by freezing some layers.

2.4 Literature Review Table:

Table 2.1: Literature Review Table:1

Paper name	Author	Publication/Year	Methodology	Result	Future Scope
Offline Signature Recognition and Forgery Detection using Deep Learning	1.Jivesh Poddara , 2.Vinanti Parikha 3.Santosh Kumar Bhart	The 3rd International Conference on Emerging Data and Industry 4.0.(2020)	Preprocessing Method for Easier Signature Verification	85-89 for forgery detection and 90-94 for signature recognition.	Improving System via Artificial Neural Networks Training and Extracted Features and improving accuracy
Offline Signature Recognition Using Image Processing Techniques and Back Propagation Neuron Network System	1.P. Kiran, 2.B. D. Parameshachari, 3.J. Yashwanth, 4.K. N. Bharath3	Springer Nature Singapore Pte Ltd (2021)	Processing Phase- ¿Image Processing Methods- ¿Neural Network Training- ¿Feature Extraction	Achieving high recognition rates	Use support vector machine (SVM) for the recognition task
Offline Signature Verification and Forgery Detection using CNN	1.Y.Durga Sravani, 2.T.Sasi Sai, 3.V.Shra vanth 4.G.Rama Naidu	INTERNATIONAL JOURNAL FOR ADVANCED RESEARCH IN SCIENCE TECHNOLOGY .(2023)	Identifying Handwriting Signature Forgeries with Pattern Recognition and CNN for detecting forged signatures	Limited training data	Train the model with a larger dataset

Table 2.2: Literature Review Table:2

Paper name	Author	Publication/Year	Methodology	Result	Future Scope
OffSig-SinGAN: A Deep Learning-Based Image Augmentation Model for Offline Signature Verification	1.M.Muzaffar 2.Rodina Ahmad 3.Laiha Mat Kiah 4.Ghulam Murtaza 5.Noman Mazhar1	Tech Science Press (2023)	Creation of a generative adversarial network (GAN)-based model to augment signature images, aiming to generate high-quality variations from a single input image	The model was able to generate high-quality signature images with diversity	creating a network capable of receiving and producing images of different sizes
A Comprehensive Analysis the importance of handwritten authentication verification and Signature Verification System	1. Sweta 2.Kumari	published in the Journal of Emerging Technologies and Innovative Research (JETIR) in January 2019.	Involves a comprehensive analysis, feature extraction methods, authentication techniques, and improving the accuracy processes.	With achieving high levels of percentage accuracy in authenticating signatures.	Improving a comprehensive analysis, feature extraction methods, and authentication techniques, potential research directions
OFFLINE SIGNATURE VERIFICATION USING TRANSFER LEARNING AND DATA AUGMENTATION ON IMBALANCED DATASET	1.Aeron Mitchell 2.Evan Edbert 3.Gregorius Natanael Elwirehardja 4.Bens Pardamean	ICIC Express Letters Year: March (2023)	Various data augmentation techniques were applied to enhance model performance.	The replicated baseline models with an accuracy of 86.7.	They using an MLP block from proposed CNN used the same hyperparameters in the training configuration to match to needs

Table 2.3: Literature Review Table:3

Paper name	Author	Publication/Year	Methodology	Result	Future Scope
Explainable offline automatic signature verifier to support forensic handwriting examiners	1.Moises Diaz 2.Miguel A. Ferrer 3.Genaro Vessio	Neural Computing and Applications (2024)	Assigns the questioned signature to the UBM and a reference set using simple distance measures.	(EER)of 5.19,1.99, and 1.25 for different scenarios, outperforming several existing methods	DL-based features were also considered to quantify the performance for each feature.
AVN: An Adversarial Variation Network Model for Handwritten Signature Verification	1.Huan Li, 2.Ping Wei, 3.Ping Hu	IEEE TRANSACTIONS ON MULTIMEDIA, VOL.2024.	Generating variants of original data and training a model with desired attributes.	Accuracy of 90.32 and an EER of 9.77 on the GPDS dataset.	Existing techniques for data augmentation aim to address overfitting issues or improve the model's performance.
Handwritten Signature Verification using Deep Learning	1.Eman Alajrami, 2.Belal 3.A.M. Ashqar, 4.Bassem S.5.Abu-Nasser, 6.Ahmed J.	International Journal of Academic Multi-disciplinary Research (IJAMR)	Split data into 6:4,7:3,8:2,High Accuracy,Efficiency in Verification.	Validation Accuracy: 97-99.7. Training Accuracy: 98.2- 99.9.	There is some underfitting here, and further epochs show overfitting.

Table 2.4: Literature Review Table:4

Paper name	Author	Publication/Year	Methodology	Result	Future Scope
Siamese Convolutional Neural Network-Based Twin Structure Model for Independent Offline Signature Verification	1.N.Sharma 2.S.Gupta 3.H.G. Mo-hamed 4.D. Anand 5.J.L.V. Mazón 6.D. Gupta 7.N. Goya	Journal: Sustainability, Volume: 14, Article Number: 11484	They achieved high accuracy on forgeries training datasets and has potential for real-world signature verification	GPDS Synthetic Dataset: 92 Bengala-Dataset: 80 Hindi Dataset: 78	Authors highlight the need to explore additional techniques to improve the potential performance .
Offline Signature Verification: An Application of GLCM Features in Machine Learning	1.Kunjal Prasad, 2.NDSV Nageswara Rao, 3.Kamal Kishor Naik, 4.Ko-maragiri Srinivas Jagannath, 5.Vatsala Sinha.	"Annals of Data Science" journal, Volume 9(6), pages 1309–1321	Convolution neural network Support vector machine Offline signature verification GLCM features	achieving an 81 accuracy in forged signatures and correctly 80 of the total forged signatures in the dataset.	authentication systems across a range of domains and further the field of offline signature verification
Deep learning based data augmentation method and signature verification system for offline handwritten signature	1.Muhammed Mutlu Yapıcı, 2.Adem Tekerek, 3.Nurettin Topaloğlu	Pattern Analysis and Applications 23(4):1-15	They use a Limited data set. They use 75 person signatures. That's why they face overfitting in the first route.	VGG16, Net121, ResNet50 from this model achieving 97.0 validation accuracy	Proposed Data Augmentation Method for the limited dataset.

Chapter 3

Methodology

Proposed methodology

The methodology outlined in the paper "Handwritten Signature Verification using Deep Learning" involves several key steps to develop an effective signature verification system using Convolutional. Neural Networks (CNNs). Here's a detailed breakdown of the methodology:

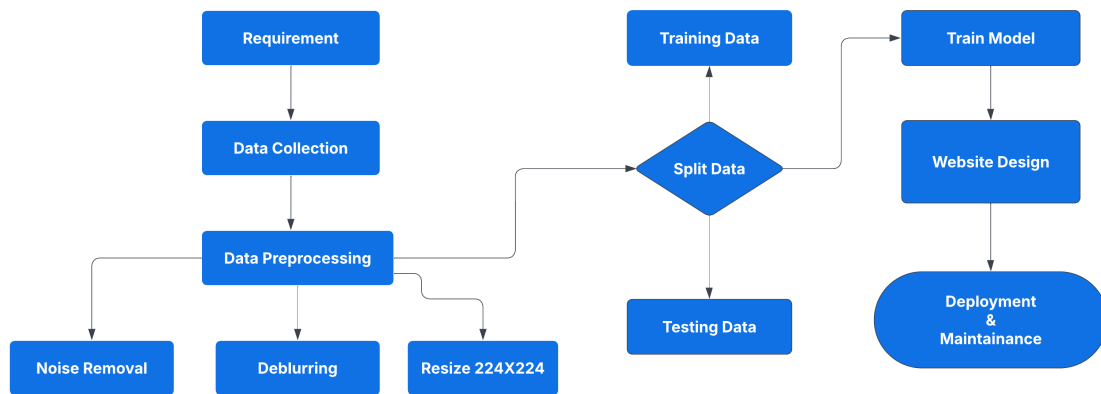


Figure 3.1: Overall workflow diagram of the proposed system.

The flowchart 3.1 shown in the figure outlines the methodology of a real-time signature verification system. It begins with data collection, followed by data processing and feature engineering to prepare the data. After the data is split into subsets for training and testing, deep learning models such as Resne50 are trained for signatures and using their respective training datasets. After training, the models

are evaluated using the test dataset, ensuring they perform well. The results are fed into the model evaluation step, where the performance of DL models is assessed. The best-performing algorithm which is ResNet-50 is then selected through the best-fit algorithm step, and finally, the most suitable model will be deployed in the mobile application for use.

3.1 Data Collection

Our group of four members initially took 400 forms. Each form contained five original signatures from a single individual. In total, we collected 2,000 original signatures. Using these original signatures, we manually created 2,000 forged signatures, resulting in a dataset of 4,000 signatures 2,000 original and 2,000 forged. From this dataset, we separated the Bengali signatures, isolating 3,500 English signatures and 500 Bengali signatures. The Bengali signatures were set aside to ensure specificity for Bengali signature analysis. This is the process we followed to generate data for our project.

3.1.1 Data processing:

- **Data split:** With form for collection of literacy we went to everyone and collected their literacy from face to face presence.
- **Image Resizing:** All signature images were resized to a uniform dimension (224×224 pixels) to ensure compatibility with the neural network input requirements.
- **Normalization:** The pixel values of the images were normalized to a range of 0 to 1 to enhance numerical stability during training.
- **Data Augmentation:** Techniques such as rotation, flipping, zooming, and brightness adjustment were applied to increase the diversity of the training dataset and reduce the risk of overfitting.

3.1.2 Data collection form:

We have developed the data collection form to gather data. the original form and the forgery form

Signature Collection Form

Full Name :

Phone Number :

Signature Instructions:

- Please sign in the designated area below (**Minimum 5 signs**).
- Use **black ink** if filling out a paper form.

Signature Area:

Consent and Agreement:

- I hereby consent to the collection and use of my signature for the purpose of signature verification in the System.
- I agree to participate voluntarily.
- I understand that my signature data will be used solely for the specified purpose and will be kept confidential.

Participant's Signature: _____

Date: _____

Privacy Notice:

Your privacy is important to us. We will handle your signature data with care and in accordance with our Privacy policy. Your signature data will not be shared with third parties without your explicit consent, except as required by law.

Figure 3.2: Original signature collection form

Forged Signature Collection Form

Purpose:

This form is designed to collect forged signatures for the purpose of signature verification system. All data collected will be used strictly for research and development purposes to improve the accuracy and reliability of signature verification technology.

Full Name :

Phone Number :

Date :

Signature Area (**Minimum 5 signs**):

Note : Privacy is important to us. We will handle all signature data with care and in accordance with our Privacy policy. Signature data will not be shared with third parties.

Figure 3.3: Forged signature collection form

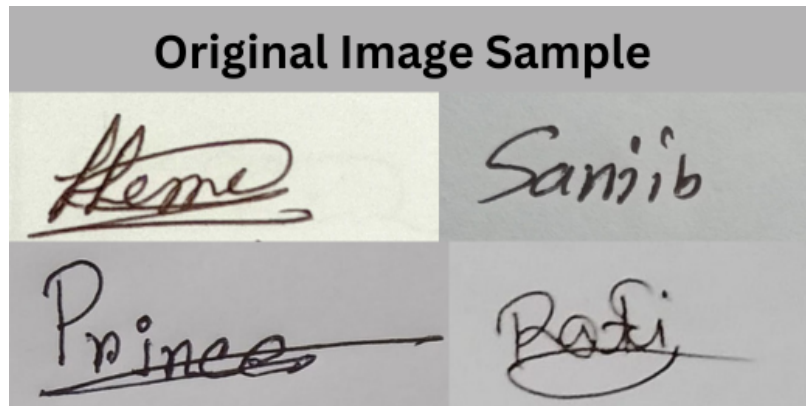


Figure 3.4: Original signature collection

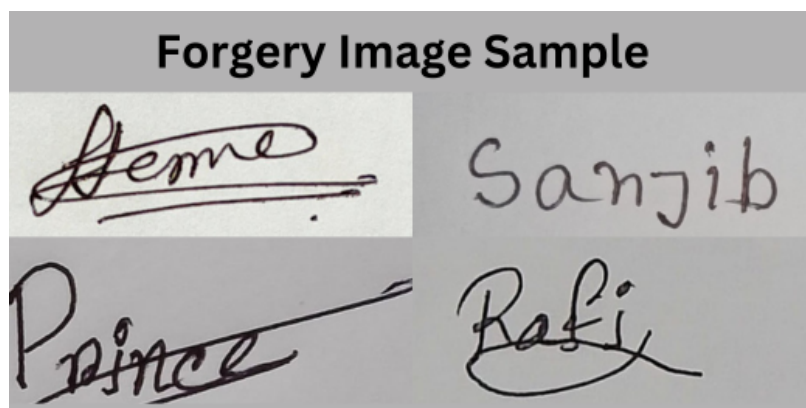


Figure 3.5: Forged signature collection

- We have informed everyone that their signatures will be used for this project and have collected their signatures through the signature form below
- We have informed all and collected their fake signatures through the signature form with consent that copies of their original signatures will be used for this project.

..

3.1.3 Original signature form:

Used for collecting sample signatures for documentation or verification purposes. The form provides the following identification data: Name and Date. A signature matrix with multiple slots allows several submissions of signature samples provided by individuals. Also, guidelines on filling out the form are provided, including how to ensure correctness and consistency in signature style.

3.1.4 Forgery signature form:

Designed for fraud situations with the purpose of gathering samples of fake signatures for comparison or analysis. has the following fields: Name and date. Samples of forged signatures are provided on the Signature Matrix. includes a purpose statement, explaining why false signature samples need to be collected. contains disclaimers or guidelines on acceptable usage or confidentiality.

3.2 Preprocessing

3.2.1 Image Resizing

All signature images are resized to a standard resolution of 224 x 224 pixels to ensure uniformity across the dataset.

3.2.2 Binarization

The images are binarized to convert them into a binary format, which simplifies the analysis and helps in feature extraction.

3.2.3 Data split

With form for collection of literacy we went to everyone and collected their literacy from face to face presence.

3.3 Implementation

Automated Signature Verification System: The final model was integrated into an automated system where users could input signature images, and the model would process the images and provide predictions (genuine or forged) automatically.

3.4 Model Selection and Architecture

3.4.1 Choice of Model:

The researchers selected the Inception ResNet 50 model due to its effectiveness in image classification tasks. They also compared it with six other pre-trained models: VGG 16, VGG 19, ResNet 50, ResNet 101, MobileNet, and EfficientNet.

3.4.2 Architecture Design:

- The final layer of the pre-trained models was replaced with a new fully connected layer suitable for binary classification (genuine vs. forged).
- The architecture included convolutional layers, pooling layers, dropout layers, and a softmax activation function for the output layer.

3.5 Training the Model

3.5.1 Training Process:

The CNN model is trained using the training dataset. The training process involves feeding the model batches of images and adjusting the weights based on the loss calculated from the predictions.

3.5.2 Loss and Accuracy Monitoring:

The model's performance is monitored by tracking the accuracy and loss metrics during training. This helps in identifying issues such as overfitting or underfitting.

3.6 Validation and Testing

3.6.1 Validation:

After training, the model is validated using a separate holdout set to assess its performance. The validation accuracy is calculated to determine how well the model generalizes to unseen data.

3.6.2 Testing:

The final model is tested on a set of signature images that were not part of the training or validation sets. The accuracy of the model in classifying these signatures as real or forged is evaluated.

3.7 Model Evaluation

3.7.1 Performance Metrics:

Our research evaluated the performance of the proposed model and the pre-trained models using metrics such as accuracy, precision, sensitivity, and F1-score. These metrics provided insights into the model's ability to correctly classify genuine and forged signatures.

3.8 Model Development

3.8.1 CNN Architecture:

A Convolutional Neural Network (CNN) is designed using the Keras API with TensorFlow as the backend. The architecture includes multiple convolutional layers followed by max-pooling layers, flattening, and dense layers. The model architecture is detailed in the paper, showcasing the number of layers, output shapes, and parameters for each layer.

3.9 Model Methodology

The picture is a flowchart that shows how to build a deep-learning model to classify

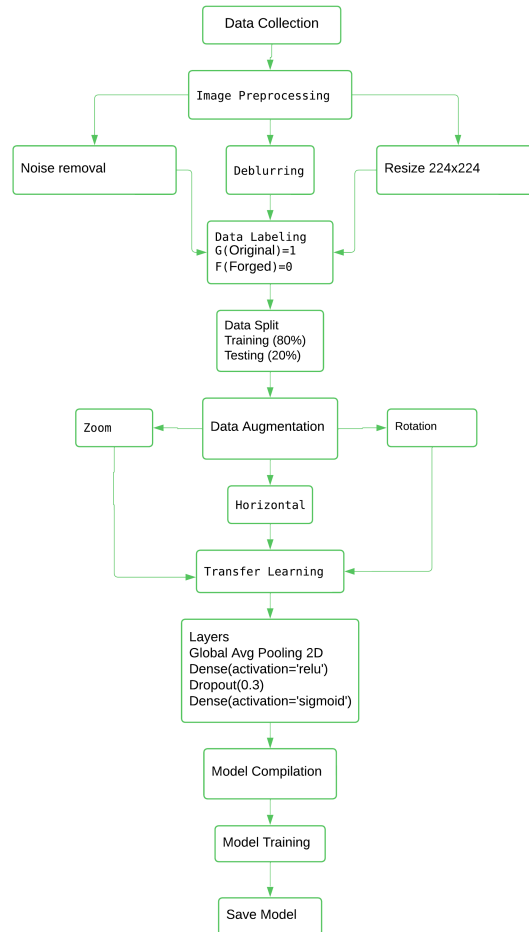


Figure 3.6: Model Methodology

images. Data collection comes first, then picture preparation, which involves scaling, normalization, and noise reduction. After that, the dataset is separated into training, validation, and test sets. Training is improved with the use of data augmentation. Transfer learning is used to train the model using pre-trained architectures such as ResNet50 . Accuracy and loss measures are used to evaluate the model after training. Lastly, predictions are produced and the model is tested.

3.9.1 Data Collection

First, images are collected: the dataset used to train as well as test the model.

3.9.2 Information Gathering

- The initial stage of every machine learning project is data collecting. Here, images are collected and separated into two datasets:
- Training Dataset: This dataset is used to train the model.
- Test Dataset: This dataset is used to test the correctness of the model.

3.9.3 Preprocessing Images

- Before training, raw pictures under go a number of preprocessing procedures to eliminate extraneous artifacts and normalize them. These consist of:
- Noise Removal: (Goal:) Eliminates undesired distortions or artifacts (such as compression artifacts and sensor noise). Why: Noise might impair the model,s performance and lead to inaccurate feature learning.
- Deblurring: (Goal:) Improves image quality by reducing blur brought on by motion or focus problems.

3.9.4 DATA LABELING (SUPERVISED LEARNING METHOD)

3.9.4.1 DEFINITION:

Data labeling is the process where images are labeled with meaningful labels or classes to enable the model to learn the distinction between various classes. One may categorize these labeled images into two classes:

- Original Images (G): Label 1; that is, real or original images.
- Forged Images (F): Label 0; that is, fake or forgery images.

3.9.5 WHY IT IS IMPORTANT:

3.9.5.1 SUPERVISED LEARNING DEPENDENCY:

Machine learning algorithms would need labeled data in order to understand the distinction between these classes. In the absence of these labels, the model does not know which features belong to either of the classes—real or fake.

3.9.6 PATTERN RECOGNITION:

Labeled data is helpful for the model learning about the discussed concept, i.e. about regular patterns like texture, color distribution, and anomalies—Meaning the differentiation series of real and fake images.

3.9.7 PERFORMANCE EVALUATION:

Labels could serve as a performance measurement stepping stone to the model, enabling comparison between the output of the model and the ground-truth labels: accuracy, precision, and recall could be the additional indexes, then.

3.9.8 AUTOMATION IN REAL-WORLD APPLICATIONS:

With good labeling, such models can handle real-world applications, which, in real-time, need to automate the identification of forged documents or images or even signatures.

3.9.9 LABELING TECHNIQUES USED:

- **Manual Labeling:** In this method, the task is accomplished by human annotators with manual review of images followed by labeling as necessary.
- **Automated Labeling:** Software or algorithms may help to automatically annotate data based on some predefined rules.
- **Semi-supervised Labeling:** A combination of human and automation processes.

3.9.10 CHALLENGES IN DATA LABELING:

Highly resource-intensive procedure that demands domain expertise in manual annotation. Errors in labelling are the same as giving confused pictures to the model and, thus, poor performance. Inconsistent use of labeling could mean a model trained with biased results.

3.9.11 Data Splitting (Train-Test Splitting)

3.9.11.1 Definition:

In this method, we usually divide the dataset into two parts: training and testing. This is to ensure the training and evaluation of the model is as efficient as possible.

3.9.12 Training Set (80%):

Enabling the model to follow the established pattern and derive relationships between the data. Weight adjustment comes from this designation.

3.9.13 Test Set (20%):

Does the testing with the unseen data of the model. Helps evaluate whether the model is operative outside the scope of the training data.

3.9.14 Generalization of the Model:

It ensures that the model is not reproducing the training data but, instead, applies the learned patterns to the never-before-seen images.

3.9.15 Overfitting Avoidance:

Overfitting means that the model is feasible at times when presented training data but falters dramatically when new testing data is presented. Thus, a good split puts the model in contact with data that remains a mystery to it.

3.9.16 Performance Benchmarking:

Evaluation of the appropriate model parameters, such as accuracy, precision, recall, and F1 score, is necessary to determine how the given model responds to real-life data.

Common Ratios for Data Split:

80-20 Split: (Mostly used method)

80% training for completion, 20% for testing.

70-30 Split:

Allowing more test data for sitings concerned with exposure to high risks that demand thorough examination.

90-10 Split:

Wherein for a tiny dataset, more importance is given to training.

3.9.17 Expected Data Splitting Challenges:

Data Leakage: Photographs that would be rather close to the similarity tend to corrupt the measurement of the model success in both training and test sets. Therefore, this could lead to bias.

Class Imbalance: In the absence of a fair division between forged and genuine images, it becomes probable for the model to betray itself in favor of the major class.

3.9.18 Best Practices

Always use stratified sampling to ensure that the class distribution is maintained across data splits. Consider cross-validation to provide a great deal of confidence for model generalisation. Always shuffle the dataset before splitting, given that splitting up a dataset will usually require randomization.

3.9.19 Data Augmentation

3.9.19.1 Definition:

Data augmentation is the manipulation of a dataset to produce artificial samples by applying transmutations over images whereby it becomes robust against variations, which imitate the real-world environment.

Common Augmentation techniques are as follows.

Rotation:

Description: Randomly rotates images by a given angle (± 10 , ± 20 degrees).

Why: Helps the model recognize objects despite their orientation.

Example: A signature that appears tilted may still be valid but end up with different alignments.

Zoom:

Description: Zooms the image in and out by a random scaling.

Why: Enables the model to learn images taken at various distances.

Example: A document that is zoomed in on should remain recognizable.

Horizontal Flip:

Description: Flips the image along the horizontal axis.

Why: Useful when the orientation varies in real-world data.

Example: Demonstrate how forged documents may present a mirror image.

3.9.20 Why Use Data Augmentation:

Improves Generalization: By introducing variation in the dataset, the model is trained to deal with real-world changes. **Reduces Overfitting:** Interestingly, the vast representation enables the model to ignore memorizing images minutely. **Addresses Low Data:** Augmentation increases the dataset data points, which mature-group the model learns from.

3.9.21 Simulation of Real-World Conditions:

Images taken in real-world can have lighting, orientation, and resolution changes, and augmentation helps stimulate those conditions.

Other Augmentation Techniques (Optional):

- **Brightness Adjustment:** Changes brightness levels to simulate different lighting conditions.
- **Contrast Adjustment:** With more or less contrast, some features become more apparent.
- **Gaussian Noise Addition:** Introduces noise from training data for better robustness.

3.9.22 Learning Transfer

For feature extraction, a pre-trained deep learning model is used instead of developing it from scratch. Besides, it has been extended by adding extra classification layers.

3.9.23 Extra important layers:

Global Average Pooling 2D (GAP):

Why: This will reduce the dimensions of features preserving the spatial information.

Why: It summarizes feature maps with reduced computation. Dense Layer (activated by ReLU)

Role: The fully connected layer processes features taken from GAP.

Activation: $f(x) = \max(0, x)$ defines the Rectified Linear Unit (ReLU).

Formula : $f(x) = \max(0, x)$ $f(x) = \max(0)$.

Why : Efficient and helps to mitigate the problem known as disappearing gradient.

Dropout (0.3)

Objective: Lower overfitting by switching off 30% of neurons during training.

Why: This helps force the network to pick out more robust features.

- Dense Layer(activated by sigmoid): 7. Model Compilation

Once the model is built, it needs to be compiled using:

- loss function: This is a metric that defines the gap between forecasts and actual labels. Example: Binary Cross-Entropy.
- optimizer: It alters the weights in the model to decrease the loss. Example: An Adam optimizer.
- metrics: This will assess the success of the model. Example: accuracy. Why it is important: Compilation gets the model ready for training by specifying how learning should occur.

3.9.24 Model Training

The education dataset is used to match the model, in which it learns patterns from categorized photographs thru multiple iterations (epochs). Key factors:

- Learning fee adjustment
- Batch size for processing
- Number of epochs to train over
- Learning fee adjustment
- Batch size for processing
- Number of epochs to train over

3.9.25 Model Evaluation

Once educated, the version is tested at the check set to degree its overall performance the use of **evaluation metrics inclusive of:**

3.9.26 Accuracy:

Measures what number of snap shots had been classified correctly.

$$\text{Accuracy} = \frac{\text{Total Predictions Correct Predictions}}{\text{Total Predictions}}$$

Loss: The error charge which tells the extent to which a prediction differs from the real label.

3.10 Website Development Methodology

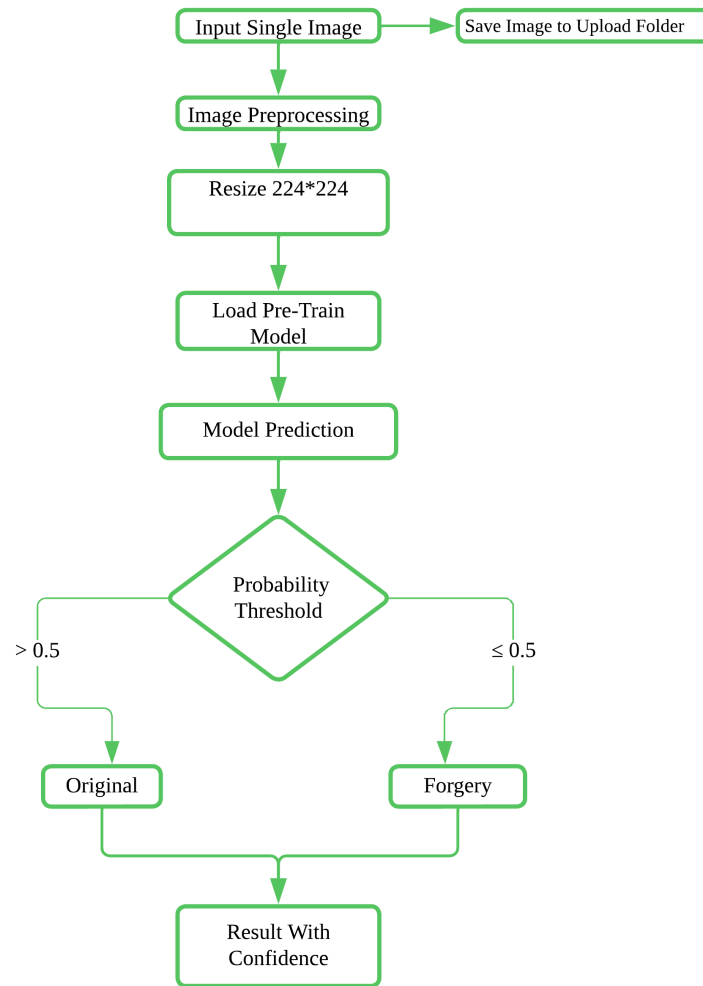


Figure 3.7: Website Development Methodology

This flowchart describes the method from which we use a pre trained deep learning model (ResNet50) to classify an input image as an “Original” and “Forgery” based on a probability threshold. Here is a breakdown of how I went through the process:

3.10.1 Load Pre-Trained Model

This involves the first step of loading a pretrained deep learning model (e.g. ResNet50), which was trained on a large dataset (e.g. ImageNet). Pre trained models leverage past learned features, which saves time and compute without any additional training. Having set its learnt weights, the ResNet50 model is ready to receive an input image.

3.10.2 Single Image Input Goal:

The system evaluates single image (either Original or Forgery) at a time.

Why necessary Often to do inferencing the classifier needs one or more images, and in this case, we are looking at only one image per inferencing cycle. What happens is that the system reads your input image into memory.

3.10.3 Preprocessing Images

enhancing the quality of the image and adjusting it to meet the model input requirements.

Actions taken:

Noise Remove (Gaussian Blur): Removes the unwanted noise, smoothing the image while keeping the edges. Deblurring or sharpening the kernel increase the clarity in the image by sharpening the edges and minor features to accentuate features. Change to (224, 224) in size: resizes the image to be 224 by 224 pixels, the input size required by, the ResNet50 model.

3.10.4 ResNet50 preprocessing:

This additional processing is to ensure that the image conforms with the input that ResNet50 is designed to work with. Different models require different input criteria like normalizing it to a certain range.

In case of the picture is normalized, pixel values are standardized to be fit into model training distribution or scaled between 0 and 1.

3.10.5 Model Forecasting

The preprocessed image is used with the ResNet50 model to check whether it is real or fake. The model utilizes the features it generates to create a probability score, returning the likelihood and corresponding feature. What it does: It provides a probability score, between 0 and 1, that indicates whether the image is original or forgery.

3.10.6 Probability Threshold:

We then want to finally classify the image by thresholding the output from the model. Logic for classification, If this probability is greater than 0.5, then the image is classified as Original. If the probability is lower than or equal to 0.5, then the image is classified as Forgery.

Why it's necessary Thresholding ensures a clear binary categorization based on the model confidence level.

3.10.7 Result with Confidence

The image classification model, a tool that analyses photos and generates predictions, provides a final result and a confidence score. By assigning a credibility score, the model assists consumers in determining how confident it is with its prediction. The output probability score from the model indicates the likelihood that the image belongs to a given class. If the probability is greater than 0.5 the image is classified as "Original" and if it is less than 0.5 it is classified as "Forgery". The final classification and its confidence percent are displayed to get an intuitive understanding of the model's belief for its choice. It is important for judging model performance and decision support. This means a forecast with a higher score counts as trustworthy and a forecast with lower value means very The aim over time is to monitor these confidence scores to create insights on where the model needs to improve (such as to retrain on different / more data).

Chapter 4

Design and Development

4.1 System Overview

we use a pre-trained deep learning model (ResNet50), the Signature Verification System is a Flask-based web application that can identify if a given signature is "Original" or "Forgery." Users can upload signature photos to the application, which then evaluates them and returns verification results along with confidence scores.

4.2 Design Architecture

We show in Figure 4.1 how to create a website where users can upload a picture to achieve the desired outcome. We also describe how it was developed.

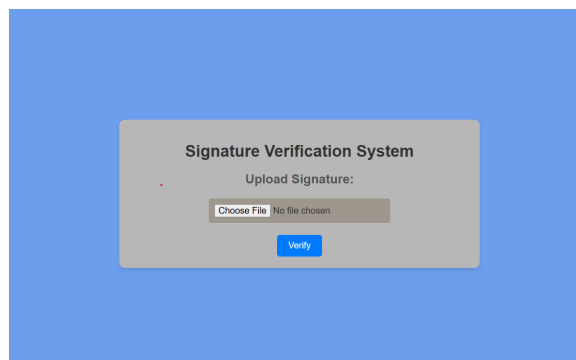


Figure 4.1: Signature verification system Website home page

4.2.1 Frontend

- **HTML and CSS:**
- To make the interface user-friendly, two templates (index.html and result.html) were utilized.
- A live preview of the uploaded signature and an upload form are available on the index.html page.
- The verification result and confidence score are shown on the result.html page.
- An external stylesheet (style.css) is used to include CSS styling.

4.2.2 JavaScript:

The functionality creates a live preview of the supplied signature image before to submission by utilising JavaScript's FileReader API. The FileReader reads the file as a Data URL when the user picks a file (in this case, a picture of their signature). The image is then instantly shown on the webpage when this Data URL is provided as the src of a (img) element. This guarantees that consumers see the precise image they will be sending by giving them immediate feedback on the uploaded file. By enabling real-time upload confirmation, it improves the user experience.

4.2.3 Backend

- Flask Framework:
- Used to handle HTTP requests and render templates. Two routes:
- /: Serves the upload form.
- /verify: Processes the uploaded image, performs prediction, and displays the result.

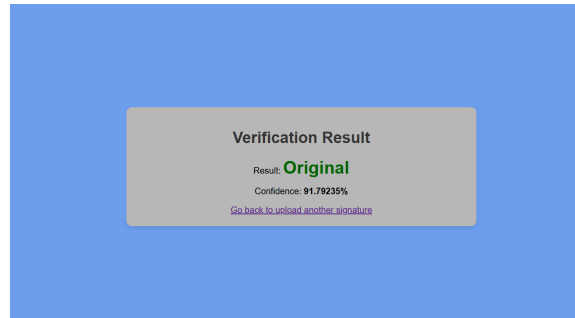


Figure 4.2: Verification Result page : (Original)
We uploaded the signature picture, and it will display as green if it is original.

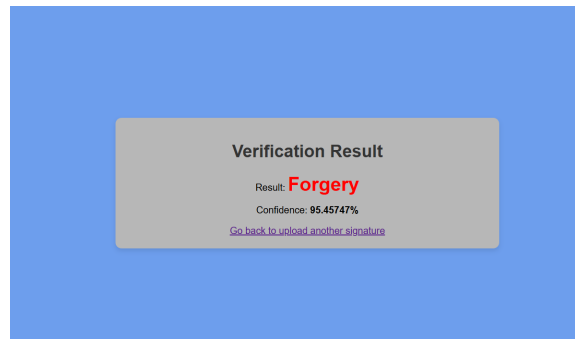


Figure 4.3: Verification Result page : (Forgery)
Then We upload the signature picture, and it will display as read if it is Forgery.

4.2.4 Machine Learning Integration:

- The application loads a pre-trained TensorFlow model (For Bangla and Bangla-English Data).
- The model is responsible for classifying the uploaded signature.

4.2.5 Image Preprocessing

- **Resizing Images:**

Images are resized to a fixed dimension of 224x224 pixels, which is the standard input size expected by ResNet50. This ensures compatibility with the model, as it requires a consistent input size across all images.

- **Using OpenCV (cv2):**

The cv2 library, part of OpenCV, is employed for two key purposes:

. images: It loads image files into a format suitable for further processing.
.Resizing images: It scales the images to the required dimensions (224x224) while maintaining the aspect ratio or applying transformations as needed.

- **Normalization with preprocess input:**
TensorFlow's preprocess input function is used to normalize the image pixel values. This function applies the following operations:
.Converts pixel values from the range [0, 255] to the range [-1, 1] or other formats specific to ResNet50.
.Adjusts the image data to align with the preprocessing used during the training of the ResNet50 model. This step improves the model's performance and accuracy by ensuring that the input data matches its training distribution.

Overall, these steps ensure that raw images are transformed into a standardized format suitable for efficient and accurate inference using the ResNet50 model.

4.3 Development Details

4.3.1 Environment Setup

- **Python Libraries:**
 - To construct the web application, use Flask.
 - TensorFlow: To load and use the model that has already been trained. For preparing images, use OpenCV (cv2).
 - To manage array transformations, use NumPy. The signature image will display green if it is an original one.

4.3.2 Forgery Application Workflow

- **Upload Route (/):**
 - Displays the upload form.
 - Provides live preview functionality for uploaded images.
- **Verification Route (/verify):**

- Handles the uploaded image.
- Saves the image in the uploads/ directory.
- Preprocesses the image and feeds it into the model.
- Interprets the model's prediction and determines the result (Original or Forgery).
- Returns the result with a confidence percentage.

4.3.3 Model Prediction

- The model returns a probability score:
- If the score greater than 0.5, the signature is classified as "Original."
- If the score is equal or less than 0.5, it is classified as "Forgery."
- Confidence is calculated based on the prediction score.

4.3.4 User Interface (UI) Design

- **Index Page (index.html)** Features: Header with the title: "Signature Verification System." File upload form with a file input (accept="image/*") to restrict uploads to image files. A JavaScript-powered preview of the uploaded signature. Submit button to trigger the verification process.
- **Result Page (result.html)**
Features: Displays the result ("Original" or "Forgery") with dynamic CSS classes for styling. Shows the confidence score in percentage. A link to return to the upload page.

4.3.5 Key Functionalities

- **Image Upload and Preview** Real-time signature preview using JavaScript ensures a better user experience and reduces input errors.

- **Signature Verification** Efficient classification of signatures through a pre-trained ResNet50 model. Image preprocessing ensures compatibility with the model.
- **Result Feedback** Clear display of verification results with confidence percentages. Dynamic styling for result categories (Original and Forgery).

4.3.6 Challenges and Solutions

- Challenge: Handling different image formats and dimensions. Solution: Used OpenCV to standardize image size and format during preprocessing.
- Challenge: Maintaining model compatibility with uploaded files.
- Solution: Implemented TensorFlow's preprocess input to ensure proper normalization of inputs.
- Challenge: User experience for file uploads.
- Solution: Integrated a preview feature to make uploads more intuitive and error-free.

4.3.7 Future Enhancements

- Multi-language Support: Add translations for users in different regions.
- Enhanced Model: Fine-tune the model for higher accuracy using more diverse training datasets.
- User Authentication: Include a login system for personalized usage tracking.
- Database Integration: Store verification results for future references or audits.

Chapter 5

Result analysis

We have run our collected data on VGG16, AlexNet, Logistic Regression, Random Forest, Support Vector Machine, Decision Tree model. The results are:

Table 5.1: Model Training and Evaluation Results

Matric	Value
VGG16 Accuracy Score	0.7472
AlexNet Accuracy Score	0.7622
Logistic Regression Accuracy Score	0.5925
Random Forest Accuracy Score	0.8550
Support Vector Machine Accuracy Score	0.6125
Decision Tree model Accuracy	0.7750
ResNet-50 accuracy Score	0.9100

Thus, several models tried on the dataset and accuracy ratings reflect how well did each model perform the classification or prediction of the output using your data. I am going to look into how well the Decision Tree model performed with 77.50 (0.7750) accuracy. Understanding How Decision Trees Function A decision tree is a supervised learning algorithm that splits the data based on required features and forms a tree-like structure for a regression (or classification) shaped model. Medium Interpretability Complexity Decision trees can model both non-linear correlations and also interactions between variables,yet they are relatively simple models.They tend to perform well when there is some clear pattern in the data. Test Case : Checking Accuracy against other Models The Decision Tree (77.50) model performed better than SVM (61.25) and Logistic Regression (59.25), indicating

that the data exhibits some non-linearity that is challenging for simpler models to capture. But as would be expected since Random Forest is an ensemble method that combines many Decision Trees to improve generalization and lower variance, it performed worse than Random Forest (85.50). It also performed marginally worse than AlexNet (76.22) and better than VGG16 (74.72) suggesting deep learning was not particularly potent in this case. Reasons Behind the Accuracy Going for a deep decision tree may causes over-fitting and thus while the training data accuracy may be high, but the test data generalization can suffer. Typological significance: The tree was probably based on traits that provided strong class distinction. Dataset Image: Decision trees can be effective on small datasets but perform poorly when you cannot see the data. Random Forest comparison The Random Forest model outperformed the Decision Tree model (85.50) by 8 percentage points. That is likely due to Random Forest averaging of many Decision Trees, without Argumentation using all data performances give the ResNet-50 accuracy score of 0.9100, making it more robust and less prone to overfitting the training data. If you want to have a better accuracy, then you should use Random Forest instead of only one Decision Tree. Next Moves To reduce complexity, if the Decision Tree is overfitting, prune it. If you still want the working of the model to be interpretable and easy to explain, the Decision Tree could still be the right model to use since it is less complex and easier to explain compared to deep learning models. If you want even more accuracy, consider using ensemble methods like Random Forest or Gradient Boosting.

5.1 Comparison table

We demonstrate in this part of the result analysis that the model achieves a high degree of accuracy in differentiating between authentic and counterfeit signatures, with a low rate of misclassifications. While accuracy, recall, and F1 scores show balanced and trustworthy predictions for both classes, the confusion matrix validates outstanding classification performance. This illustrates how well the model handles the binary classification problem and how robust it is.

Table 5.2: Model Training and Evaluation Results for Bangla

Metric	Value
Final Training Accuracy	0.9880
Final Training Loss	0.0593
Best Validation Accuracy	0.9712
Best Validation Loss	0.1042
Test Loss	0.1112
Test Accuracy	0.9615

5.2 Bangla Dataset Metrics:

1. Final Training Accuracy: 0.9880 (98.80) - It means that the model achieved 98.80 accuracy on the training dataset. - This further indicates that the model has learned the training data very well.

2. Final Training Loss: 0.0593 - It gives the error in the predictions on the training data. - A low value of 0.0593 indicates that the model is making fewer errors on the training set.

3. Best Validation Accuracy: 0.9712 (97.12) - The highest achieved accuracy on the validation dataset during training. - This will provide good generalization to unseen Bangla validation data.

4. Best Validation Loss: 0.1042 - The lowest recorded error on the validation data. - Lower loss is better for generalizing to new data.

5. Test Loss: 0.1112 - Error on the last test dataset (unseen data). - Slightly higher than the validation loss but still low, which suggests that the model generalizes well.

6. Test Accuracy: 0.9615 (96.15) - Model accuracy on completely new Bangla test data. - In the real world, this model will do well and achieve an accuracy of 96.15
Summary for Bangla - The model is performing really well with high training and test accuracy. - Small gap between the training and test accuracy, 98.80 and 96.15 respectively, shows good generalization and minor overfitting.

Table 5.3: Model Training and Evaluation Results for All Data

Metric	Value
Final Training Accuracy	0.9656
Final Training Loss	0.1040
Best Validation Accuracy	0.9588
Best Validation Loss	0.1324
Test Loss	0.1477
Test Accuracy	0.9400

5.3 Bangla-English Mixed All Dataset Metrics:

1. Final Training Accuracy: 0.9656 (96.56) - A bit lower than the Bangla-only train accuracy, 98.80 vs 96.56. - Suggests that the model found it slightly harder to learn from mixed languages.

2. Final Training Loss: 0.1040 - Higher loss as compared to Bangla-only training of 0.1040 versus 0.0593. - This suggests that the mixed-language dataset is more challenging, leading to slightly higher errors.

3. Best Validation Accuracy: 0.9588 (95.88) - Slightly lower than Bangla-only validation accuracy of 97.12. - Indicates good generalization, has some difficulty in mixed data handling.

4. Best Validation Loss: 0.1324 - Higher validation loss compared to Bangla-only, 0.1324 vs 0.1042. - It indicates that the data in mixed language is more complicated.

5. Test Loss: 0.1477 - Slightly higher test loss than validation loss, though generalization is reasonably good.

6. Test Accuracy: 0.9400 (94.00) * Slightly lower than the test accuracy of the Bangla-only model: 94.00 versus 96.15. * Model performance is relatively somewhat worse when the inputs are mixed.

Summary for Bangla-English: - Works good, but performance is slightly degraded with respect to Bangla only. - The mixed nature of the data may bring more complexity to achieve perfect accuracy. - The overfitting is less probable as compared to the Bangla-only dataset because of the lower gap between training and test performances.



Figure 5.1: Bangla Training and Validation Accuracy Graph

5.4 Epoch (Iteration Count)

This is the number order of the epochs, representing how many rounds of processing the entire training dataset has gone through. The network has been trained on 50 such epochs; ranging from epoch 0 to 49.

Accuracy (Training Accuracy) This field reflects the portion of correctly classified samples out of the whole in the training set. It had started off at 0.5721154 (57.21) in the first epoch and got as high as 0.9879808 (98.79) by the end. This increasing trend of the line indicates that the model is training and its performance on the training dataset is improving.

5.5 Interpretation:1

High and steadily growing training accuracy reflects that the model learns patterns effectively. If there is a significant difference between the training accuracy and validation accuracy, the model may be overfitting. **Loss Training Loss** This column describes the model error in predictions made on the training dataset. Its value starts from 0.7293425 and comes down to 0.0592985 at epoch 49.

5.6 Interpretation:2

The loss decreases over epochs, implying that the model is getting close to actual values in its predictions. An extremely low training loss can indicate overfitting if it doesn't have a similar effect on validation loss.

5.7 Validation Accuracy

This column measures how well the model performs on unseen validation data. It starts at 0.6730769 (67.30) and reaches an upside of as much as 0.9711539(97.11) by epoch 49.

5.7.1 Interpretation:

An increasing validation accuracy implies that the model generalizes well to new data. If the validation accuracy has leveled off, while the training accuracy is still rising, then overfitting may be occurring.

5.8 Validation Loss

This column reflects the model's error on the validation dataset; hence, it tells how well the model generalizes. It starts at 0.5771379 and drops to 0.1111971 by the final epoch.

5.8.1 Interpretation:

A decreasing validation loss implies better generalization. If validation loss starts increasing while training loss decreases, that may be overfitting. Key Observations and Insights: Overall Performance Improvement: Training accuracy increased significantly from 57.21 to 98.79, showing very strong learning. Validation accuracy increased from 67.30 to 97.11, indicating that the model is generalizing well. Possible Overfitting Indications: From around epoch 35 onwards, the training accuracy is above 97, while the validation accuracy stays between 95-97. If validation accuracy does not continue improving, it may indicate overfitting.

Loss Behaviour:

Training and validation losses have been decreasing consistently, which is a good sign of learning. If the validation loss had started increasing while the training loss kept decreasing, that would indicate overfitting.

5.9 Model Stability:

At around epoch 30-35, the validation accuracy stabilizes at approximately 95, indicating that the model may have reached its peak performance. Training could have been stopped earlier to save resources (early stopping).

Conclusion and Recommendations: Training Effectiveness:

The model is performing well, with validation accuracy closely following training accuracy. Loss values are reducing steadily, meaning the model is learning efficiently. Potential Overfitting Monitoring: Consider implementing early stopping based on validation loss to avoid overfitting. Regularization techniques like dropout or L2 regularization could further help prevent overfitting.

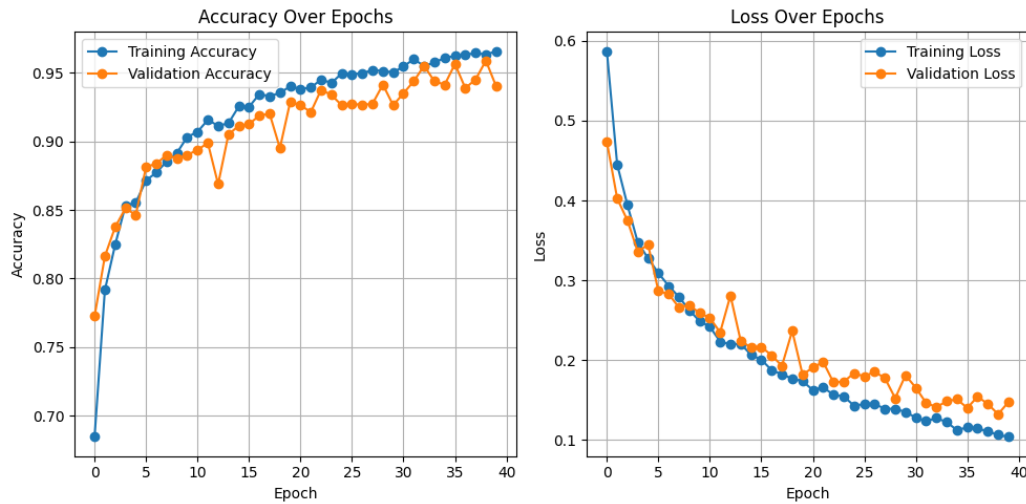


Figure 5.2: Bangla and English Training and Validation Accuracy Graph

5.10 Epoch (Training Iteration)

The epoch is defined as the number of times one sees all the data in the training dataset. More epochs normally imply better learning, but too many lead to overfitting. This table was trained on 40 epochs, starting from epoch 0 to epoch 39.

5.11 Training Accuracy

The accuracy column describes the extent to which the model is doing a good job of fitting our training dataset. Accuracy is computed by taking the number of correctly predicted instances divided by the total instances in the training set.

5.11.1 Example values:

- At epoch 0, accuracy = 0.6846875 (68.47)
- At epoch 39, accuracy = 0.965625 (96.56)

5.12 Interpretation:

A smooth trend of increase in training accuracy shows how well the model is learning. However, if the training accuracy comes out to be a lot higher than the

validation accuracy, then that might be overfitting.

5.13 Loss (Training Loss)

The loss column measures how well the model's predictions match up with the actual values in the training set. It basically represents the error in the model's predictions, and this is usually computed applying loss functions like Mean Squared Error or Cross-Entropy Loss.

5.13.1 Example values:

- At epoch 0, loss = 0.5869067
- At epoch 39, loss = 0.1040473

5.14 Interpretation:

A downward-sloping loss means that there is learning and improvement in the model. If loss flattens out or oscillates, that is an indication the model has learned all it can learn.

5.14.1 Val Accuracy (Validation Accuracy)

The val accuracy column gives the accuracy of the model in making predictions from the validation dataset. It helps measure how well the model generalizes to new, unseen data.

5.14.2 Example values:

- At epoch 0, val accuracy = 0.7725 (77.25)
- At epoch 39, val accuracy = 0.94 (94.00)

5.15 Interpretation:

If validation accuracy is similar to training accuracy, it indicates a well-generalized model. If validation accuracy plateaus while training accuracy increases, it could mean the model is overfitting.

5.15.1 Val Loss (Validation Loss)

The val loss column essentially measures how well the model's predictions match the actual values in the validation set. It helps detect whether the model is overfitting or underfitting. Example values: At epoch 0, val loss = 0.4737457; at epoch 39, val loss = 0.1477404 Interpretation: A decreasing validation loss indicates that the model is performing well on unseen data. If validation loss starts increasing while training loss keeps decreasing, overfitting is occurring. Key Observations and Analysis Overall

5.15.2 Improvement:

Training accuracy increased from 68.47 to 96.56. The validation accuracy also improved from 77.25 to 94.00, hence learning was consistent. Risk of Overfitting: In the vicinity of epoch 40, training accuracy is very high (96), but validation accuracy is slightly lower(94). This can imply overfitting, where performance on new data will be marginally poorer. Loss

5.15.3 Consistency:

Training loss reduced from 0.58 to 0.10, and validation loss from 0.47 to 0.14; both are well fitted. Stable Model Performance: After epoch 25, validation accuracy stabilizes around 92-94, suggesting the model has learned sufficiently.

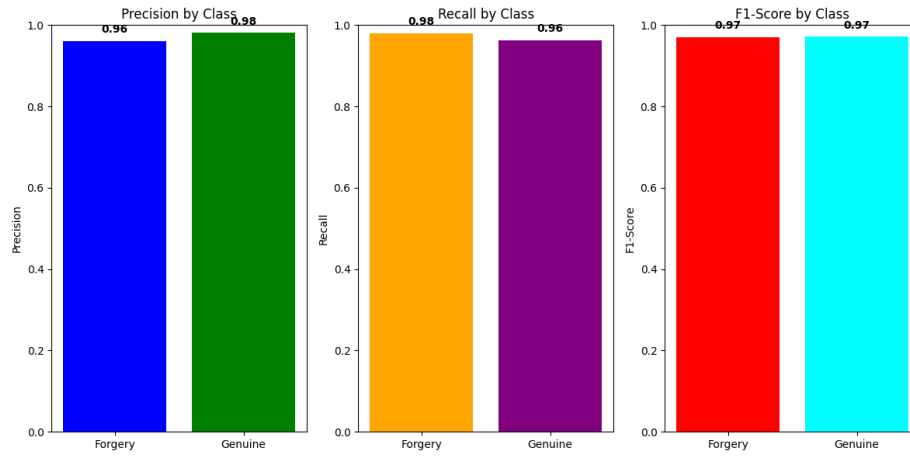


Figure 5.3: Result Evaluation Metrics for Bangla Data set

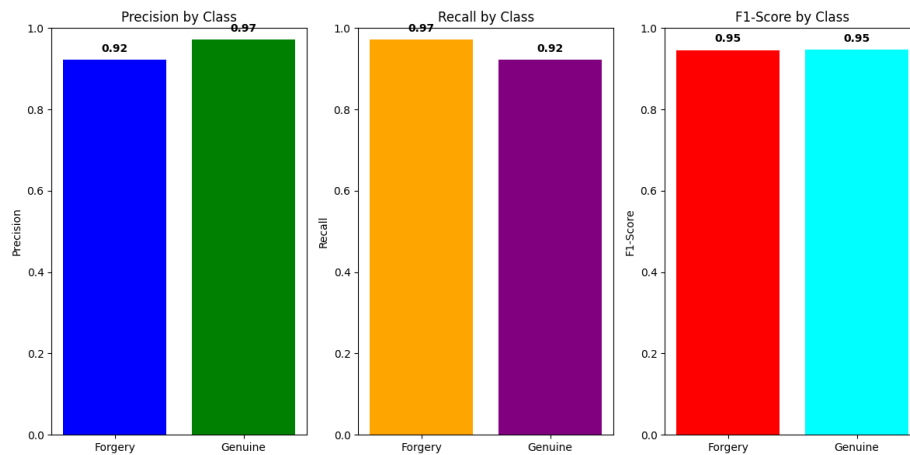


Figure 5.4: Result Evaluation Metrics for All Data set

5.16 Final Evaluation:

The model seems ready for deployment if the goal is high accuracy, but further evaluation on real-world data is necessary to ensure reliability

5.17 Result Evaluation

Score are the standard metrics for model performance assessment for binary classification use cases and they are some of the metrics being displayed. Class-wise Precision (Left Chart) • Definition: Precision measures the ratio of expected positive cases

that led to actual positive results. $\text{Precision} = \frac{\text{True Positives}}{\text{False Positives} + \text{True Positives}}$

5.17.1 Observations are made:

- Forgery (blue bar): 0.96
- Genuine: Green bar, 0.98

5.18 Further interpretations:

Most of the forecasts tagged as Forgery or Genuine are indeed correct as the model has a very good accuracy for both the classifications. A more accurate classifier for the Genuine class may show less false positives. Recall by Class (Middle Chart)

5.18.1 Definition:

Recall measures how many of the actual positive cases were accurately identified by the model. It is computed as follows: $\text{Recall} = \frac{\text{True Positives}}{\text{False Negatives} + \text{True Positives}}$

Real Positives Observed: o Real: 0.96 (purple bar) o Forgery: 0.98 (orange bar)

- Interpretation: o “Forgery” has slightly higher recall than “Genuine,” which indicates that the model does a little better job of correctly identifying faked instances. For fraud detection jobs, where the cost of not detecting a fake is extremely high, you want high recall, with fewer false negatives.
- Definition: The F1-score is the harmonic mean between accuracy and recall which is used to balance the trade-off between positives and false negatives

$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

There are the observations on both sides: o Real: 0.97 (Cyan bar) o Fake: 0.97 (Red bar)

5.19 Interpretation:

The F1-scores are equal for both classes, which implies a well-balanced model that deals with recall and accuracy well. The model gives a consistent performance, does not have a tone of favouritism between classes, very good score of 0.97. Overall

Evaluation • The high accuracy, recall, and F1-score for the Forgery and Genuine makes the model very good in distinguishing between them. Suggestions for Enhancement

Check overfitting: even a good result could mean that the model is memorizing and not generalizing patterns, especially if we have a short dataset. Thresholding: The recall to precision tradeoff may be further optimized through thresholding for categorization.

Cross-Validation: Ensure the model is robust by testing it on multiple data subsets

Extra Metrics: if you want a deeper insight, consider looking at extra metrics such as the confusion matrix or the AUC-ROC.

Chapter 6

Project Management and Financial Activities

6.1 Project Overview

Our project goal of a real-time signature verification system that uses a CNN model with ResNet-50 is to validate signatures by differentiating between legitimate and fake. The project includes preprocessing the data, using a dataset of signatures to train a deep learning model based on ResNet-50, and deploying the model for real-time verification. Through the use of ResNet-50's deep layers, the system pulls characteristics from signature photos, analyses them, and determines whether they are faked or legitimate. Applications like banking, legal verification, and secure authentication can benefit from the model's accuracy and performance optimization. System integration, UI development, and performance assessment are also included in the project.

6.2 Development Tools

6.2.1 Programming Languages

Python – For model development and implementation.

6.2.2 Deep Learning Frameworks

TensorFlow/Keras – For building and training the ResNet-50 model. PyTorch (Alternative) – If you prefer working with PyTorch for deep learning.

6.2.3 Image Processing and Data Handling

OpenCV – For image preprocessing and real-time signature capture. Pandas and NumPy – For data manipulation and handling datasets.

6.2.4 Model Training and Evaluation

Scikit-learn – For performance evaluation (accuracy, precision, recall, F1-score). Matplotlib and Seaborn – For data visualization and model performance analysis.

6.2.5 Development Environment

Jupyter Notebook – For writing and testing Python scripts. Google Colab – For cloud-based GPU training (if local resources are limited).

6.2.6 Web and API Integration

Flask / FastAPI – For creating a web API for signature verification. Streamlit / Django (Alternative) – For developing a user-friendly web interface.

Table 6.1: Financial Activities chart

Expenses	Cost
Form	4000
Travel	600
Poster	500
Report	1600
website	2000
Google Colab Pro	1300
Total Cost	10,000

6.3 Financial Activities

Here is our project financial activities table:

Form (4000 bdt) – This cost associated with forms corresponds to documenting.

Travel (600 bdt) – This includes local travel costs.

Poster (500 bdt) – A budget line for designing and printing a presentation poster, or project poster.

Report (1600 bdt) – Expenses associated with preparation, design, and printing of the final project report.

website(2000 bdt)-This expense is likely for website development, hosting, domain registration, or maintenance.

Google Colab Pro(1300 bdt)- This expense use for project code generate

Total Cost (10,000 bdt)- Sum of all expenses, indicative of total expense/budget needed.

Chapter 7

Conclusions

This study focuses on the development of an offline signature verification system using a Convolutional Neural Network (CNN) with the ResNet architecture demonstrates a significant advancement in accuracy and reliability for signature authentication [Ling et al. \(2024\)](#). Leveraging ResNet’s residual learning framework effectively addresses the challenges of vanishing gradients in deep networks, enabling the model to extract robust and hierarchical features even in highly complex datasets. The ResNet-based model achieves superior performance compared to traditional CNNs, effectively distinguishing between genuine and forged signatures [Ghosh \(2021\)](#). The skip connections in ResNet enhance the learning process, allowing the network to capture intricate patterns and features in signature images while mitigating over fitting [Tahir et al. \(2021\)](#) [Banerjee et al. \(2021\)](#).

7.1 Social, Legal, Ethical, and Environmental Issues

signature verification systems are frequently used for fraud prevention and authentication, there are a number of social, legal, ethical, and environmental concerns with their use and implementation. An examination of these factors is provided below:

7.2 Social

7.2.1 Accessibility:

It may be difficult for those with impairments or disabilities (such as hand tremors or restricted mobility) to use such systems efficiently [Ren et al. \(2023\)](#).

7.2.2 Trust and Acceptance:

Particularly in high-stakes situations like financial transactions, users may have misgivings about automated systems and prefer human verification.

7.2.3 Digital Divide:

Certain groups may not be able to access services or benefits in areas with restricted access to technology due to signature verification procedures.

7.3 Legal

7.3.1 Data Protection Laws:

We collected the original and Forgery signatures offline in Bangla and English data through forms and then created the dataset. Since the signatures were collected with the consent of the participants, there may be legal conflicts in the misuse of the dataset.

7.3.2 Liability:

It can be difficult to establish who is responsible for false positives or negatives, particularly in situations involving money or legal issues.

7.3.3 Authentication Standards:

Jurisdictions may have different laws governing whether or not handwritten or electronic signatures are accepted [Zhou et al. \(2021\)](#). Intellectual property: Patent disputes may arise from the private nature of some algorithms employed in signature verification systems.

7.4 Ethical

7.4.1 Bias or Discrimination:

Certain groups may be treated unfairly if the system's accuracy varies according to signature characteristics like age, race, or handwriting style [Fierrez-Aguilar et al. \(2005\)](#).

7.4.2 Informed Consent:

There are ethical transparency issues since users cannot completely comprehend how their data is being gathered, stored, or used.

7.4.3 Over-reliance on Technology:

Relying too much on automated systems might undermine human judgment and critical thinking [Ren et al. \(2023\)](#). Data Security: If biometric information is not kept safely or is sold or given without permission, ethical questions are raised.

7.5 Environmental Issues

7.5.1 Energy Consumption:

Carbon emissions and energy consumption are increased by large-scale signature verification systems, particularly those that use machine learning and demand a lot of processing power.

7.5.2 E-Waste:

If not disposed of properly, hardware required to implement these systems (such as servers and scanners) might contribute to the expanding issue of electronic waste.

7.5.3 Paperless Systems' Sustainability:

Although signature verification methods encourage a shift away from paper, the energy consumption of digital infrastructure may outweigh the environmental benefits of these systems.

7.6 Brief Summary

By examining static characteristics in scanned or captured images, a real-time offline signature verification system verifies handwritten signatures. Using deep learning models, using CNNs, it preprocesses the signature, extracts distinctive traits, and compares them to templates that have been saved. By using similarity scores to verify authenticity, the system ensures security in applications such as fraud detection, legal agreements, and banking. Identifying competent forgeries and the unpredictability of authentic signatures are challenges.

7.7 Future works

Future research on real-time offline signature verification systems could focus on the following areas:

7.7.1 Improved Feature Extraction

- **Advanced Neural Architectures:** To capture minute details in signature patterns, more complex deep learning architectures, like Transformers, are being investigated.
- **Hybrid Models:** For increased accuracy, hybrid models combine deep learning-based characteristics with conventional handcrafted features (such as roughness and curvature).

7.7.2 Cross-Domain Adaptability

- Developing models that can generalize across different signature styles, paper types, or scanning devices without requiring retraining.
- Incorporating domain adaptation techniques to handle variations in writing instruments or environmental conditions.

7.7.3 Enhanced Forgery Detection

- Building more robust systems that can differentiate between skilled forgeries and genuine signatures by analyzing intricate micro-features.

- Leveraging adversarial training to simulate forgeries and improve the system's robustness against them.

7.7.4 Integration of Multimodal Data

- Combining offline (static image) and online (dynamic pen movement, pressure, speed) signature data for more comprehensive verification.
- Using additional metadata like signer habits or handwriting history to improve accuracy.

7.7.5 Explainable AI (XAI) in Verification

Using explainable AI techniques to give concise, comprehensible justifications for verification decisions would increase confidence in crucial applications such as banking and contracts.

7.7.6 Scalability and Efficiency

- Optimizing models for real-time processing on edge devices, ensuring seamless offline operation without compromising accuracy.
- Developing lightweight models for deployment on resource-constrained devices such as mobile phones or embedded systems.

7.7.7 Personalized Verification Systems

- Creating adaptive models that fine-tune verification parameters for individual users based on their unique signature characteristics.

7.7.8 Cross-Cultural Signature Analysis

- Investigating the impact of cultural and linguistic differences on signature styles to develop globally effective systems.
- Training models to recognize non-standard or stylized signatures often seen in specific regions.

7.7.9 Addressing Ethical and Privacy Concerns

- Ensuring compliance with data privacy regulations (e.g., GDPR) while storing and processing signature data.
- Exploring federated learning approaches to allow secure, decentralized training of models.

7.7.10 Benchmarking and Standardization

- Establishing universal benchmarks and datasets to evaluate the performance of offline signature verification systems consistently.
- Encouraging collaboration between academia and industry to set standards for deployment in critical applications.

References

- Akhundjanov, U., Soliyev, B., Juraev, N., Musayev, K., Norinov, M., Ermatova, Z., & Zaynabidinov, R. (2024). Off-line handwritten signature verification based on machine learning. In *E3s web of conferences* (Vol. 508, p. 03011).
- Alsuhimat, F. M., & Mohamad, F. S. (2023). A hybrid method of feature extraction for signatures verification using cnn and hog a multi-classification approach. *IEEE Access*, 11, 21873–21882.
- Banerjee, D., Chatterjee, B., Bhowal, P., Bhattacharyya, T., Malakar, S., & Sarkar, R. (2021). A new wrapper feature selection method for language-invariant offline signature verification. *Expert Systems with Applications*, 186, 115756.
- Batool, F. E., Attique, M., Sharif, M., Javed, K., Nazir, M., Abbasi, A. A., ... Riaz, N. (2024). Offline signature verification system: a novel technique of fusion of glcm and geometric features using svm. *Multimedia Tools and Applications*, 1–20.
- Diaz, M., Ferrer, M. A., & Vessio, G. (2024). Explainable offline automatic signature verifier to support forensic handwriting examiners. *Neural Computing and Applications*, 36(5), 2411–2427.
- Fierrez-Aguilar, J., Nanni, L., Lopez-Penalba, J., Ortega-Garcia, J., & Maltoni, D. (2005). An on-line signature verification system based on fusion of local and global information. In *Audio-and video-based biometric person authentication: 5th international conference, avbpa 2005, hilton rye town, ny, usa, july 20-22, 2005. proceedings 5* (pp. 523–532).
- Ghosh, R. (2021). A recurrent neural network based deep learning model for offline signature verification and recognition system. *Expert Systems with Applications*, 168, 114249.
- Guerra-Segura, E., Ortega-Pérez, A., & Travieso, C. M. (2021). In-air signature verification system using leap motion. *Expert Systems with Applications*, 165, 113797.

- Hameed, M. M., Ahmad, R., Kiah, L. M., Murtaza, G., & Mazhar, N. (2023). Off sig-
sin gan: A deep learning-based image augmentation model for offline signature
verification. *Computers, Materials & Continua*, 76(1).
- Hameed, M. M., Ahmad, R., Kiah, M. L. M., & Murtaza, G. (2021). Machine
learning-based offline signature verification systems: A systematic review. *Signal
Processing: Image Communication*, 93, 116139.
- Jagtap, A. B., Sawat, D. D., Hegadi, R. S., & Hegadi, R. S. (2020). Verification
of genuine and forged offline signatures using siamese neural network (snn).
Multimedia Tools and Applications, 79(47), 35109–35123.
- Kao, H.-H., & Wen, C.-Y. (2020). An offline signature verification and forgery
detection method based on a single known sample and an explainable deep
learning approach. *Applied Sciences*, 10(11), 3716.
- Kiran, P., Parameshachari, B., Yashwanth, J., & Bharath, K. (2021). Offline signature
recognition using image processing techniques and back propagation neuron
network system. *SN Computer Science*, 2(3), 196.
- Li, H., Wei, P., & Hu, P. (2021). Avn: An adversarial variation network model for
handwritten signature verification. *IEEE Transactions on Multimedia*, 24, 594–608.
- Ling, L.-R., Zhang, H., Yin, F., & Liu, C.-L. (2024). Deep metric learning with
cross-writer attention for offline signature verification. In *International conference
on document analysis and recognition* (pp. 250–267).
- Malik, M. I., Liwicki, M., Dengel, A., & Found, B. (2014). Man vs. machine: A
comparative analysis for signature verification. *Journal of Forensic Document
Examination*, 24, 21–35.
- Mitchell, A., Edbert, E., Elwirehardja, G. N., & Pardamean, B. (2023). Off line signa-
ture verification using transfer learning and data augmentation on imbalanced
dataset. *ICIC Express Letters*, 17(03), 359–366.
- Ozyurt, F., Majidpour, J., Rashid, T. A., & Koc, C. (2024). Offline handwriting
signature verification: A transfer learning and feature selection approach. *arXiv
preprint arXiv:2401.09467*.
- Poddar, J., Parikh, V., & Bharti, S. K. (2020). Offline signature recognition and
forgery detection using deep learning. *Procedia Computer Science*, 170, 610–617.
- Raigonda, M. R., et al. (2024). Signature verification system using ssim in image
processing. *Journal of Scientific Research and Technology*, 5–11.

- Ren, J.-X., Xiong, Y.-J., Zhan, H., & Huang, B. (2023). 2c2s: A two-channel and two-stream transformer based framework for offline signature verification. *Engineering Applications of Artificial Intelligence*, 118, 105639.
- Sharma, N., Gupta, S., Mohamed, H. G., Anand, D., Mazón, J. L. V., Gupta, D., & Goyal, N. (2022). Siamese convolutional neural network-based twin structure model for independent offline signature verification. *Sustainability*, 14(18), 11484.
- Singh, P., Verma, P., & Singh, N. (2022). Offline signature verification: an application of glcm features in machine learning. *Annals of Data Science*, 9(6), 1309–1321.
- Tahir, N. M., Ausat, A. N., Bature, U. I., Abubakar, K. A., & Gambo, I. (2021). Off-line handwritten signature verification system: Artificial neural network approach. *International Journal of Intelligent Systems and Applications*, 10(1), 45.
- Tolosana, R., Vera-Rodriguez, R., Gonzalez-Garcia, C., Fierrez, J., Morales, A., Ortega-Garcia, J., ... others (2022). Svc-ongoing: Signature verification competition. *Pattern Recognition*, 127, 108609.
- Wang, W., Xu, H., Alazab, M., Gadekallu, T. R., Han, Z., & Su, C. (2021). Blockchain-based reliable and efficient certificateless signature for iiot devices. *IEEE transactions on industrial informatics*, 18(10), 7059–7067.
- Zhou, Y., Zheng, J., Hu, H., & Wang, Y. (2021). Handwritten signature verification method based on improved combined features. *Applied Sciences*, 11(13), 5867.