Bangla Handwritten Character Recognition Using Deep Learning Approaches and its Explainability With AI

By

Fahim Habib
Student ID: 20201169010

Department of Computer Science and Engineering
Faculty of Science & Technology
North Western University, Khulna-9100
Bangladesh
May, 2024

Bangla Handwritten Character Recognition Using Deep Learning Approaches and its Explainability With AI

By

Fahim Habib
Student ID: 20201169010

SUBMITTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF
BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING
AT
NORTH WESTERN UNIVERSITY
KHULNA, BANGLADESH
May 2024

NORTH WESTERN UNIVERSITY, KHULNA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

The undersigned hereby certify that they have read and recommended for acceptance a thesis Entitled "**Bangla Handwritten Character Recognition using Deep Learning Approaches**" by **Fahim Habib, Noushin Atia, Fardin Sanjida** in partial fulfillment of the requirements for the degree of **Bachelor of Science in Computer Science and Engineering**.

1. Research Supervisor

Md. Mahedi Hasan
Senior Lecturer
Department of Computer Science and Engineering
North Western University, Khulna

2. Second Examiner

Md. Inzamam-Ul-Hossain
Assistant Professor
Department of Computer Science and Engineering
North Western University, Khulna

3. Head of the Department

Md. Mahedi Hasan
Senior Lecturer
Department of Computer Science and Engineering
North Western University, Khulna

NORTH WESTERN UNIVERSITY, KHULNA

Date: May 6, 2024

Authors	:	**Fahim Habib.**
Title	:	**Bangla Handwritten Character Recognition using Deep Learning Approaches**
Department	:	**Computer Science and Engineering**
Degree	:	**Bachelor of Science in Computer Science and Engineering**

Permission is herewith granted to North Western University to circulate and to have copied for non-commercial purpose, at its discretion, the above title upon the request of individuals or institutions.

Fahim Habib

Abstract

The realm of Bangla handwritten character recognition (BHCR) has long been overshadowed by the dominance of more mainstream languages, despite Bangla's status as one of the most widely spoken languages globally. Deep learning (DL) techniques have become a prominent solution for Bangla handwritten character recognition (BHCR) due to their ability to extract high-level features from complex data. In our comprehensive study, we meticulously explored the efficacy of twelve DL models on the arduous task of Bangla character recognition, meticulously evaluating their performance on two distinct datasets: a handwritten character dataset and CMATERDB, comprising a formidable collection of 15,000 images. We also have provided a comparative performance analysis of the DL models for BHCR. Among the compared model LSTM, Bi-LSTM, CNN, Inception, VGG, ResNet. We achieved the maximum performance at ResNet152V2. The proposed model consists of gaining above 96%. In this study, the proposed method is shown satisfactory recognition accuracy 98.76% on dataset, which is so far, one of the best accuracies for Bangla character recognition.

Keywords: *Ability, BHCR, Comprising, Character, Deep learning, Prediction*

Acknowledgments

We are blessed by the grace of almighty ALLAH for giving us the ability, intelligence and energy to complete our thesis paper.

At first we are grateful to almighty Allah for giving us patience to perform our Thesis properly. We would like to express our sincere gratitude to our supervisor Md. Mahedi Hasan for giving us the opportunity to pursue research on this rapidly emerging field and move forward along the path of innovation and novelty. His invaluable research ideas and experience kept us motivated all the way towards the progress and completion of this research work.

Our parents also give us support and encouragement to fulfill our graduation. Last but not the least, we are grateful to We are very much thankful to our most kind and helpful supervisor Md. Mahedi Hasan for his extraordinary patience, continuous encouragement, guidance and advice. This research experiment would not have been possible without his support. We have been amazingly fortunate to have an advisor who gave us the freedom to explore on our own. He patiently supported us to overcome many crisis situations and at the same time he guided us to recover when our steps faltered.to assist us in our thesis work/ project work.

Dedication

A special feeling of gratitude to our loving parents whose words of encouragement and push worked as magic on us. And also, our favorite sir Md. Mahedi Hasan has never left our side and always encouragedus in every step. We will always appreciate what they have done for us. We dedicate this work to them with a special thanks and honor.

Table of Contents

List of Tables

List of Figures

Glossary of Terms

TERM		FULL FORM
BHCR	**:**	Bangla handwritten character recognition
DL	**:**	Deep learning
CNNs	**:**	Convolutional Neural Networks
ResNet	**:**	Residual Network
LSTM	**:**	Long Short-Term Memory
Bi-LSTM	**:**	Bidirectional Long Short-Term Memory
VGG	**:**	Visual Geometry Group
Inception	**:**	GoogLeNet

Chapter 1

Introduction

1.1 Background

In recent years, the intersection of deep learning and character recognition has propelled significant breakthroughs across various linguistic domains, fostering advancements in language processing, image analysis, and document digitization. Among these languages, Bengali holds a unique position with its intricate script and widespread usage across South Asia. The complexity of the Bengali script, characterized by its cursive nature and diverse character set, poses significant challenges for automated recognition systems. As such, research in Bangla Handwritten Character Recognition (BHCR) remains a critical endeavor, driven by the need to develop robust and accurate recognition models capable of deciphering handwritten Bengali characters with high precision and efficiency.

In this paper, we present a comprehensive study on BHCR utilizing state-of-the-art deep learning models, aiming to push the boundaries of performance and accuracy in character recognition tasks. Our research builds upon a foundation of meticulously curated datasets, comprising 15,000 handwritten Bengali character images sourced from diverse sources to ensure representativeness and variability in writing styles. The dataset is partitioned into 8400 images for training and 3,000 images for testing and Total Validation images are 3600, with each image annotated and classified into one of 50 distinct character classes.

The primary objective of our study is to evaluate the efficacy of various deep learning architectures in the context of BHCR, thereby contributing to the advancement of character recognition technology in the Bengali language. To achieve this objective, we employ a diverse set of deep learning models, including Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and advanced architectures such as Inception, VGG, and Bi-LSTM. Each model undergoes rigorous training and evaluation over 30 epochs, following a standardized methodology to ensure consistency and reliability in performance assessment.

Our initial experimentation yields promising results, with notable variations in performance across different model architectures. Notably, the ResNet model emerges as the top-performing architecture. Motivated by this success, we delve deeper into the ResNet framework, exploring various iterations and modifications to optimize performance further.

In particular, we investigate variants of the ResNet architecture, including ResNet50, ResNet-50V2, ResNet101, ResNet101V2, ResNet152, and ResNet152V2, aiming to identify the most efficient and accurate model for BHCR. Through iterative refinement and fine-tuning, we strive to leverage the capabilities of ResNet-based architectures to achieve unprecedented levels of accuracy in Bangla handwritten character recognition.

Our findings underscore the significant potential of deep learning methodologies in addressing the complexities of BHCR, offering insights into the optimal model architectures

and training strategies for achieving superior performance. Furthermore, our research contributes to the broader discourse on character recognition technology, highlighting the importance of language-specific approaches in addressing the unique challenges posed by complex scripts like Bengali.

In the subsequent sections of this paper, we provide a comprehensive review of related work in the field, detailing key advancements and methodologies in BHCR research. We then elucidate the methodology and experimental setup employed in our study, offering insights into dataset preparation, model architectures, and training procedures. Subsequent sections present the results and analysis, providing a detailed examination of the performance of different models and their implications for BHCR. Finally, we conclude with reflections on the findings, implications for future research, and avenues for further exploration in the field of Bangla handwritten character recognition.

1.2 Motivation

The motivation behind our research stems from the pressing need to address the challenges inherent in Bangla Handwritten Character Recognition (BHCR) using cutting-edge deep learning methodologies. The Bengali script, with its rich linguistic heritage and widespread usage, presents a unique set of complexities that necessitate specialized approaches for accurate and efficient character recognition. Despite significant advancements in character recognition technology, BHCR remains a daunting task due to the script's cursive nature, diverse character set, and variability in writing styles.

we aim to contribute novel insights and methodologies that advance the state-of-the-art in character recognition technology for the Bengali language.

Ultimately, our motivation lies in the transformative potential of BHCR technology to drive social, cultural, and economic development in Bengali-speaking communities worldwide. By pushing the boundaries of performance and accuracy in BHCR. we aspire to preserve the rich heritage of the Bengali script for future generations.

1.3 Objectives

- **Evaluate Deep Learning Models for BHCR**: Our primary objective is to assess the efficacy of various deep learning architectures in the context of Bangla Handwritten Character Recognition (BHCR). By conducting a systematic evaluation of models such as Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and advanced architectures like Inception and VGG, we aim to identify the most effective approaches for accurately recognizing handwritten Bengali characters.
- **Optimize Performance through Iterative Refinement**: Building upon the initial evaluation, our objective is to refine and optimize the performance of the most promising deep learning models identified during the evaluation phase. We aim to leverage techniques such as fine-tuning, hyperparameter optimization, and architectural modifications to enhance the accuracy and efficiency of BHCR systems, particularly in challenging scenarios characterized by variability in writing styles and script complexity.

- **Investigate ResNet Variants for BHCR**: Inspired by the initial success of the ResNet model in BHCR, our objective is to explore and evaluate various iterations and modifications of the ResNet architecture. By experimenting with ResNet50, ResNet-50V2, ResNet101, ResNet101V2, ResNet152, and ResNet152V2, we seek to identify the optimal ResNet variant for achieving superior performance in Bangla handwritten character recognition tasks.
- **Contribute to the Advancement of BHCR Technology**: Beyond performance evaluation, our objective is to contribute novel insights, methodologies, and best practices to the field of BHCR. Through rigorous experimentation, analysis, and documentation of our findings, we aim to provide valuable resources and guidelines for researchers, practitioners, and developers working in the domain of character recognition, particularly for languages with complex scripts like Bengali.

1.4 Contributions

- **Data Preprocessing Optimization**

  Data preprocessing is a fundamental step in deep learning pipelines, impacting model performance and generalization. The research has prioritized comprehensive preprocessing techniques using TensorFlow's Keras API, specifically leveraging the Image Data Generator for efficient data augmentation and normalization. Key aspects of the data preprocessing pipeline include:

  - Image Resizing and Grayscale Conversion: Images are resized to a consistent dimension of 50x50 pixels, optimizing computational efficiency while preserving essential visual features. Additionally, grayscale conversion (color_mode='grayscale') is applied, reducing data complexity and focusing model learning on essential image characteristics.
  - Normalization: Pixel values are rescaled to a range of [0, 1], a standard practice that ensures uniform data input across the model. Normalization enhances model stability during training by preventing feature scaling disparities.
  - Data Augmentation (Not Explicit in the Provided Information): Although not explicitly detailed in the provided information, augmentation techniques such as rotation, zoom, and horizontal flips can be seamlessly integrated into the Image Data Generator, enriching the training dataset and improving model robustness.

The meticulous preprocessing pipeline ensures that input data is uniformly processed and optimized for subsequent model training, setting the foundation for effective feature learning and model convergence.

- **Architectural Enhancements**

Residual Blocks and Model Adaptation to enhance model learning capabilities and depth, the research introduces architectural enhancements inspired by ResNet variants:

- Residual Blocks and Model Adaptation to enhance model learning capabilities and depth, the research introduces architectural enhancements inspired by ResNet variants:
  - Residual Block Refinement: The residual_block_v2 function refines residual blocks with streamlined convolutional layers, batch normalization, and activation functions. Notably, the addition of identity mappings (shortcut connections) promotes efficient gradient propagation, enabling the training of deeper networks while mitigating the vanishing gradient problem.
  - Model Architecture Adaptation: The model architecture adaptation, influenced by ResNet-50V2 design principles, initiates with convolutional layers followed by batch normalization and ReLU activation. This tailored architecture is optimized for the specific dataset dimensions (shape= (50, 50, 1)), facilitating effective feature extraction and representation.

- **Comprehensive Evaluation Framework**

  Our research contributes a comprehensive evaluation framework for BHCR, encompassing a diverse set of deep learning architectures and methodologies. By systematically comparing and analyzing the performance of different models, we provide valuable insights into the strengths, weaknesses, and suitability of each approach for addressing the challenges of BHCR.

- **Optimized Deep Learning Models**

  Through iterative refinement and optimization, we contribute optimized deep learning models for BHCR, capable of achieving state-of-the-art performance in recognizing handwritten Bengali characters. Our refined models incorporate insights gained from experimentation and fine-tuning, resulting in enhanced accuracy, efficiency, and robustness in character recognition tasks.

- **Identification of Optimal ResNet Variant**

  Building upon the initial success of the ResNet model, our research identifies the optimal ResNet variant for BHCR. By systematically evaluating different ResNet architectures, we pinpoint the most effective variant for achieving superior performance in recognizing handwritten Bengali characters, thereby providing valuable guidance for researchers and practitioners in selecting suitable models for BHCR applications.

- **Optimization Strategies and Parameter Tuning**

  Beyond architectural enhancements, the research emphasizes optimized deep learning model development through meticulous parameter tuning strategies:

- **Hyperparameter Optimization**

  Parameters such as learning rates, batch sizes, and regularization techniques are systematically tuned to maximize model performance and generalization. This iterative optimization process fine-tunes model behaviors, enhancing training efficiency and convergence.

- **Advancement of BHCR Technology**

  Overall, our research contributes to the advancement of BHCR technology by pushing the boundaries of performance and accuracy in character recognition tasks. By providing insights, methodologies, and optimized models, we empower researchers, practitioners, and developers to develop robust and efficient BHCR systems capable of addressing real-world challenges and applications.

- **Interpretability Integration: LIME AI**

  In pursuit of enhancing model interpretability and transparency, the research integrates LIME (Local Interpretable Model-agnostic Explanations) AI techniques:

  - **LIME AI Integration**: LIME facilitates local interpretability by explaining model predictions at the instance level. This integration empowers users to understand model decision-making processes, enabling actionable insights and model refinement.

In summary, the contributions presented in this research encompass a holistic approach to deep learning model development. By refining data preprocessing techniques, introducing architectural enhancements, optimizing model parameters, and integrating interpretability techniques, this research aims to advance the state-of-the-art in deep learning methodologies, fostering robust and interpretable AI solutions for diverse applications.

1.5 Thesis Organization

The remainder of the thesis is organized as the following:

Chapter 1 (Introduction): Introduction of our research area has been explored in this section. We organized our proposed system in here. By showing graph, data we want to describe the overall thesis work flow in this section.

Chapter 2 (Background Knowledge): This section discusses background knowledge, which is necessary for future works. It goes without saying that having enough background material encourages reader confidence in the overall quality of your analysis and conclusions and helps in improving our knowledge of the research subject during research.

Chapter 3 (Related Works): Similar research to our thesis is covered here. Research papers with references are explained here. It is the most crucial component since it dictated how we

should do our work. Our research should add something fresh while still reflecting what is already known. We can find gaps in the literature, improve our strategy, and obtain insight into the state of the area by looking through related publications and references. This stage ensures consistency and relevance by forming our methodology and directing the course of our work.

Chapter 4 (Proposed Method): This chapter introduces our suggested framework, which divides sentiment into three distinct classes. We describe in detail how training data is created from raw data and then preprocessed for sentiment labeling. We also cover the different techniques for feature extraction. Our research is based by this framework, which offers an organized method for sentiment labeling.

Chapter 5 (Evaluation and Results): We conducted experiments using feature extraction techniques and a new hybrid method, which resulted in several algorithmic programs and options for labeling. Through this, we were able to classify sentiment into positive, negative, and neutral categories. Our experimental data and procedure were thoroughly analyzed to achieve accurate results.

Chapter 6 (Conclusion): An overview of the research we have done. There are also limitations and a plan for future study mentioned in this section. These could be limited by the variety of sentiment subtleties, computational resources, or the availability of datasets. However, we see a bright future for our work, intending to improve accuracy through the use of many classifiers in sentiment labeling. This method has the potential to greatly improve our sentiment analysis framework's accuracy and flexibility.

Chapter 2

Background Knowledge

2.1 Background Knowledge

In this comprehensive overview, we delve into the foundational components, methodologies, and deep learning architectures utilized in our research on Bangla Handwritten Character Recognition (BHCR). From the programming language and development environment to the intricacies of transfer learning and the nuances of various deep learning models, each aspect plays a crucial role in shaping the efficacy and efficiency of our BHCR system.

2.1 Python Programming Language

Python stands as the cornerstone of our research endeavor, chosen for its versatility, simplicity, and extensive ecosystem of libraries and frameworks tailored for machine learning and deep learning tasks. With its intuitive syntax and robust support for scientific computing, Python enables seamless implementation of complex algorithms and facilitates rapid experimentation and prototyping in the BHCR domain.

2.2 Anaconda

2.2.1 Framework

Anaconda emerges as the comprehensive solution for managing the dependencies and environments essential for our BHCR project. By providing a streamlined approach to package management and environment configuration, Anaconda ensures reproducibility and consistency across different computing platforms, simplifying the setup process and enhancing the accessibility of our research endeavors.

2.2 Jupyter Notebook

2.2.1 Interactive Computing

Jupyter Notebook serves as the interactive computing environment of choice for our research, offering a dynamic platform for integrating code, visualizations, and narrative text. With its support for executing code in a modular and iterative manner, Jupyter Notebook fosters a collaborative and exploratory approach to BHCR experimentation, facilitating the seamless transition from data exploration to model development and evaluation.

2.3 Transfer Learning

2.3.1 Concept

Transfer learning is a machine learning technique where knowledge gained from solving one problem is applied to a related but different problem. In the context of deep learning, transfer learning involves leveraging pre-trained models trained on large datasets (such as ImageNet) and fine-tuning them on a target dataset (in this case, the BHCR dataset). By starting with weights learned from a general domain and adapting them to the specifics of the target task, transfer learning enables faster convergence and improved performance, especially in scenarios with limited annotated data.

2.3.2 Fine-Tuning Strategies

Fine-tuning, a key aspect of transfer learning, involves adjusting the parameters of a pre-trained model to better fit the target task. This can involve freezing certain layers of the network to preserve learned representations and only updating the weights of the top layers, or gradually unfreezing layers and fine-tuning the entire network. Fine-tuning strategies need to strike a balance between retaining useful features learned from the source domain and adapting to the characteristics of the target domain.

2.3.3 Benefits and Challenges

Transfer learning offers several advantages, including faster training convergence, improved generalization performance, and reduced requirement for large annotated datasets. However, it also poses challenges such as domain shift, where the source and target domains have different statistical properties, and the risk of negative transfer, where the pre-trained knowledge adversely affects performance on the target task. Careful selection of pre-trained models, appropriate fine-tuning strategies, and dataset augmentation techniques are essential to mitigate these challenges and maximize the benefits of transfer learning in BHCR tasks.

2.4 Keras

2.4.1 High-Level API

Keras is a high-level neural networks API, written in Python and designed for ease of use and flexibility. It provides a user-friendly interface for building and training deep learning models, abstracting away many of the complexities associated with low-level implementation details. Keras enables rapid prototyping and experimentation, allowing researchers and practitioners to focus on model design and hyperparameter tuning without getting bogged down in the intricacies of neural network architecture and optimization algorithms.

2.4.2 Modular Design

Keras follows a modular design philosophy, allowing users to construct neural network architectures by assembling predefined building blocks such as layers, activation functions, and optimization algorithms. This modular approach fosters code reusability and encourages experimentation with different architectural configurations, facilitating the exploration of novel model architectures tailored to specific tasks such as BHCR.

2.4.3 Integration with TensorFlow

Keras serves as a high-level interface to TensorFlow, a popular deep learning framework developed by Google. As of TensorFlow 2.0, Keras has been integrated as the official high-level API for building neural networks, offering seamless interoperability with other TensorFlow components and libraries. This integration provides users with access to the extensive functionality and performance optimizations of TensorFlow while benefiting from the simplicity and ease of use of the Keras API.

2.4.4 Community Support and Documentation

Keras boasts a vibrant community of developers, researchers, and practitioners who contribute to its ecosystem through the development of extensions, tutorials, and documentation. The Keras documentation provides comprehensive guides, examples, and API references, making it accessible to users of all experience levels. Additionally, the Keras community actively engages in knowledge sharing and collaboration through forums, mailing lists, and social media channels, fostering a supportive environment for learning and innovation in deep learning research and application development.

2.5 Deep Learning

A type of machine learning known as "deep learning" involves training artificial neural networks how to learn from huge amounts of data and make predictions. It has attracted considerable interest and achieved success in a variety of fields, including robotics, computer vision, natural language processing, and speech recognition. Here are some vital details about deep learning:

2.5.1 Neural Networks

Artificial neural networks (ANNs), which are computer models inspired by the structure and function of biological neural networks in the human brain, are used to construct deep learning models. Artificial neuronal networks (ANNs) are made up of connected nodes, also known as "units." An input layer, one or more hidden layers, and an output layer are only a few of the layers that make up these units. Weights that are linked with connections between units are adjusted throughout training.

2.5.2 Convolutional Neural Networks (CNNs)

For analyzing grid-like input, such as images, convolutional neural networks (CNNs) are

a particular form of neural network architecture. Computer vision tasks have been transformed by CNNs, who are now the preferred model for image classification, object recognition, and picture segmentation. They utilize convolutional layers, which use filters or kernels to extract relevant data from images, and pooling layers, which shrink the physical dimensions of the extracted features.

2.5.3 Bi-LSTM (Bidirectional Long Short-Term Memory)

Bi-LSTM architectures are adopted for their ability to capture bidirectional dependencies in sequential data, making them particularly well-suited for modeling the temporal dynamics inherent in handwritten character sequences.

2.5.4 Inception (GoogleNet)

The Inception architecture is harnessed for its innovative inception modules, which facilitate the parallel processing of features at different scales, promoting robust feature learning and extraction capabilities.

2.5.5 LSTM (Long Short-Term Memory)

LSTM architectures are employed for their capacity to model long-term dependencies in sequential data, offering a powerful framework for capturing the intricate temporal relationships inherent in handwritten Bengali character sequences.

2.5.6 ResNet (Residual Networks)

ResNet architectures are embraced for their pioneering residual connections, which alleviate the challenges of training deep neural networks by facilitating the flow of gradients and enabling the successful training of exceedingly deep models.

2.5.7 VGG (Visual Geometry Group)

VGG architectures serve as a benchmark baseline for our BHCR research, emphasizing the importance of depth and small convolutional filters in learning complex hierarchical representations from images.

2.5.8 Extensions of ResNet

Variants such as ResNet50, ResNet101, ResNet152, ResNet-50V2, ResNet101V2, and ResNet152V2 are meticulously explored and evaluated, considering factors such as model capacity, computational efficiency, and generalization performance in the context of BHCR.

2.5.9 Training and Backpropagation

Large datasets and the backpropagation method of optimization are used to train deep learning models. To reduce the discrepancy between expected results and actual outputs, the model iteratively changes the weights of the connections between neurons during training. The adjustment is based on the loss function of the model's generated gradient, which measures the difference between expected and desired outputs. The gradient is calculated and

sent backward through the network via the backpropagation method, which changes the weights accordingly.

2.5.10 Overfitting and Regularization

Overfitting, when the model becomes too specific to the training data and fails to generalize effectively to new data, is a common issue in deep learning. Different regularization approaches are used to reduce overfitting. A common technique is dropout, which forces the network to acquire more robust and generalizable features by randomly deactivating a portion of the neurons during training. Early stopping and weight decline are additional approaches.

2.5.11 Transfer Learning

Deep learning frequently uses the approach of transfer learning, particularly when working with limited amounts of labeled data. It entails using previously learned models that were often developed on similar datasets. Researchers may save time and computing resources by using these pre-trained models as a starting point and then fine-tuning them for their particular objective or dataset.

2.5.12 Hardware Acceleration

For deep learning models to train and predict on large data sets, a lot of computing power is needed. Due to their capacity for parallel processing, graphics processing units (GPUs) and specialized hardware, such as tensor processing units (TPUs), have become important in expediting deep learning calculations. The implementation of deep learning models in real-time applications has been made possible by these hardware improvements, which have significantly shortened training durations.

By encapsulating these foundational elements within the background knowledge section, we lay the groundwork for a comprehensive understanding of the tools, methodologies, and deep learning architectures underpinning our research on Bangla Handwritten Character Recognition.

2.6 Transfer Learning VS Deep Learning

In the field of machine learning, transfer learning and deep learning are two distinct yet connected ideas. An overview of transfer learning and deep learning is provided below, showing the distinctions and connections between them [23], [24]:

- **Transfer Learning:** A machine learning apprach called transfer learning uses information learned from one task to improve performance on a separate but related activity. Transfer learning begins with a pre-trained model that has already been trained on a large amount of data, often in a different domain, as opposed to starting from scratch. The pre-trained model knows the universal features that may be used to various employment. The model is then fine-tuned on the target task or dataset, sometimes with a smaller quantity of labeled data, using these learnt features as a

starting point. Transfer learning offers the following primary benefits:

- **Data Efficiency:** Models can perform well even with little labeled data due to transfer learning. Large quantities of task-specific labeled data are not as necessary because the pre-trained model has already acquired important characteristics.
- **Time and Resource Efficiency:** Compared with creating a model from scratch, starting with a pre-trained model saves time and computing resources. The pre-trained model has previously gone through a hard training procedure on a large dataset.
- **Improved Generalization:** By recognizing common patterns and structures shared throughout the two tasks, transfer learning improves information transfer from a source task to a target task, improving generalization performance on the target task.

- **Deep Learning:** A branch of machine learning known as "deep learning" focuses on training artificial neural networks with numerous layers how to learn from huge amounts of data and make predictions. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs), two types of deep learning models, are particularly good at tasks requiring complicated patterns, such voice and image recognition. Through several layers of neurons, deep learning models may automatically learn hierarchical data representations. Deep learning's key benefits are [24]:
  - **Feature Learning:** Deep learning models don't require human feature development since they can automatically extract relevant characteristics from unprocessed input data. This skill is especially beneficial in fields where the underlying patterns are complex and hard to articulate directly.
  - **High Performance:** In a variety of applications, including image classification, object identification, machine translation, and voice synthesis, deep learning models have reached the highest level of accuracy. They can achieve high accuracy because they can learn from huge datasets and recognize complex correlations in the data.
  - **End-to-End Learning:** Deep learning models don't require manual processing in place. They may learn directly from the input data to get the required results. The model creation process is made simpler and depends less on domain knowledge with this end-to-end learning technique.

2.7 Architecture of CNN

A Convolutional Neural Network's (CNN) essential architecture is made up of a number of connected layers that are used to analyze and extract features from input data, such as images. Here is a high-level description of a CNN's usual architecture [25]:

- **Input Layer:** An image or collection of images serves as the raw input data in the input layer. The input data is organized as a multidimensional array, also known as a

tensor, with dimensions that correspond to the input's width, height, and channels (such as the RGB channels for colored images).

- **Convolutional Layers:** The fundamental components of a CNN are convolutional layers. They are made up of several filters (also known as kernels) that execute convolutional operations by sliding over the input data. Each filter combines with the input to create a feature map that draws attention to certain trends or characteristics in the input. The number of channels in the output feature maps is based on the number of filters.
- **Activation Function:** Each component of the feature maps produced by the convolutional layers is subjected to an activation function. The network is given non-linearities, helping the model to learn complex connections and recognize non-linear patterns in the input. The activation function in CNNs is frequently the Rectified Linear Unit (ReLU).
- **Pooling Layers:** The feature maps' geographic dimensions are down sampled while maintaining the most crucial data using pooling layers. A common pooling method is called max pooling, which keeps the highest value found in a small area (like a 2x2 window) and eliminates the remainder. Pooling adds a degree of translation invariance, reduces the number of parameters, and helps the representation become more compact.
- **Fully Connected Layers:** The output is typically reduced into a 1-dimensional vector after many convolutional and pooling layers. Then, one or more fully interconnected layers that imitate the layers of a conventional neural network are connected to these flattened characteristics. The fully connected layers, sometimes including additional activation functions, process the acquired characteristics and produce the final output.
- **Output Layer:** The last layer of the CNN is represented by the output layer. The configuration is chosen by the particular task being carried out. For instance, the output layer for image classification generally consists of a softmax activation function that generates a probability distribution across the classes, but the output layer for object identification can include contain bounding box coordinates and class labels.

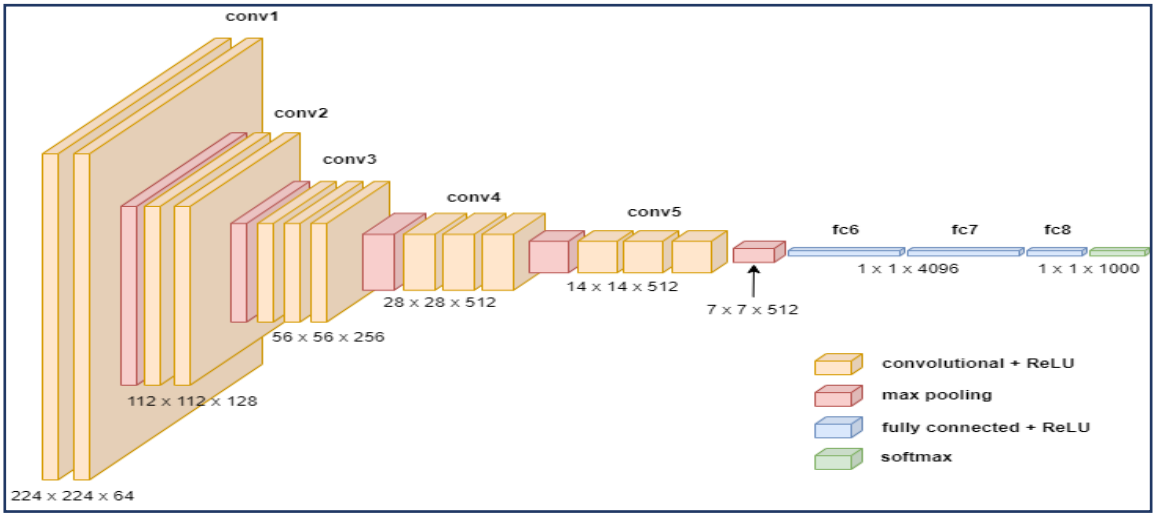

Figure 2.1: Basic architecture of convolutional neural network [18]

2.8 ResNet (Residual Network)

Introduces skip connections, allowing networks to be deeper without suffering from vanishing gradients. This architecture has been pivotal in the development of very deep convolutional neural networks.

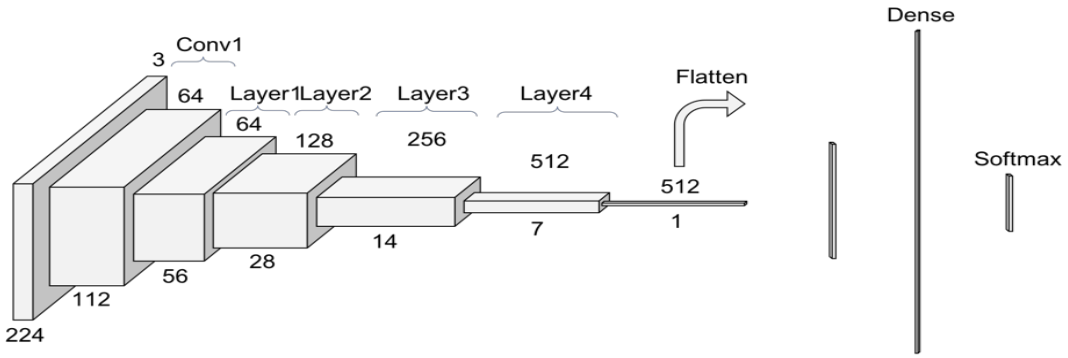

Figure 2.2: Basic architecture of Residual Network

2.9 LSTM (Long Short-Term Memory)

A type of recurrent neural network (RNN) designed to capture long-range dependencies in sequential data by maintaining a memory state. LSTMs are effective in tasks like speech recognition and language modeling.

2.10 Bi-LSTM (Bidirectional LSTM)

An extension of the LSTM that processes the input sequence in both forward and backward directions. This allows the model to capture context from past and future information simultaneously, improving performance in tasks like sentiment analysis and named entity recognition.

2.11 VGG (Visual Geometry Group)

A classic convolutional neural network architecture known for its simplicity and effectiveness. VGG networks stack multiple convolutional layers followed by max-pooling layers, achieving good performance in image classification tasks.

2.12 Inception (GoogLeNet)

A deep convolutional neural network known for its Inception modules, which use multiple convolutions of different sizes within the same layer. This architecture efficiently captures spatial hierarchies in images and has been used in various applications, including object detection and image segmentation.

Chapter 3

Related Works

Few comparable works have been carried out for heart diseases prediction and identified the risk factors. These works stand as a guideline for our work. This section briefly discusses some of those techniques and how they implement efficient algorithms and tools.

3.1 Bangla Handwritten Character Recognition Using Extended Convolutional Neural Network:

This paper aims [2] to address these challenges by proposing a method for automatic feature extraction, considering 50 classes of basic letters and 10 classes of digits for improved overall accuracy. Images from the BanglaLekha-Isolated dataset are preprocessed to a common form, with resizing to 32 x 32 pixels and conversion to white letters on a black background. In the first convolutional layer, zero padding is applied, followed by the application of eight kernels of dimension 3 x 3 x 1 to extract features. Preprocessing involves resizing images to 32x32 pixels and splitting the dataset into 80% training and 20% testing. Running the proposed model on combined Bangla hand written cha- racter set (digits, vowels and consonants are mixed together) and training loss of 0.0344 and validation loss of 0.3204. The obtained training accuracy is 99.08% and the validation accuracy is 92.25% after 10 epochs.

3.2 Bangla Handwritten Digit Recognition Using Deep Convolutional Neural Network:

This paper explores [1] the performance of various deep CNN architectures for recognizing handwritten Bangla digits, focusing on AlexNet, MobileNet, GoogLeNet (Inception V3), and CapsuleNet models. This research implements AlexNet, MobileNet, GoogleNet (Inception V3), and CapsuleNet models for Bangla handwritten digit recognition using the NumtaDB dataset. This research evaluates four state-of-the-art deep CNN architectures on the NumtaDB dataset for recognizing handwritten digits. GoogleNet achieves the highest recognition accuracy of 93% over normal data.

3.3 CNN Based Common Approach to Handwritten Character Recognition of Multiple Scripts:

The proposed approach [25] for handwritten character recognition was tested on six different databases: English, Bangla, Devanagari, Oriya, Telugu numerals, and Bangla basic characters. These databases contain 10 classes each, except for the Bangla basic characters, which has 50 classes. The recognition accuracies achieved on the respective test sets are comparable to state-of-the-art accuracies for each database. Various approaches involve preprocessing, feature extraction, classification, and postprocessing, with classifiers like

MLP, RBF, MQDF, SVM, and CNN showing promise. CNNs offer the advantage of learning feature vectors without hand-crafting, prompting investigation into skipping preprocessing and feature extraction steps. The CNN achieved 95.84% recognition accuracy on the Bangla basic character database and was trained using the Theano deep learning toolbox in Python.

3.4 Handwritten Character Recognition from Image Using CNN:

The article discusses [31] various deep learning techniques for recognizing handwritten Bangla digits, including deep belief networks (DBN), convolutional neural networks (CNN), CNN with dropout, CNN with dropout and Gabor filters, and CNN with dropout and Gaussian filters. These methods were tested on the CMATERdb 3.1.1 Bangla numeral image database. The recommended approach, implemented in Python 3, uses a CNN classifier trained on preprocessed, scaled, and standardized data to predict input characters, including Bangla, English, and numbers. Training set comprised 75% of the data, with 25% for testing, resulting in a loss of 0.5579 and an accuracy of 85.96% after ten epochs.

3.5 Bangla Handwritten Characters Recognition Using Convolutional Neural Network:

The proposed [26] CNN model consists of 10 layers connected sequentially. It starts with an input layer of size 32x32x1 followed by two convolutional layers with 32 and 64 filters, both using a 3x3 kernel and ReLU activation. Then, a max-pooling layer reduces the image size by half. This pattern repeats with two more convolutional layers. After that, a flatten layer transforms the data to 1D, followed by two dense layers and a dropout layer. The last dense layer serves as the output with softmax activation, totaling 637,724 trainable parameters. Proposed CNN-based approach for recognizing handwritten Bangla alphabet achieves 90.22% validation accuracy on Bangalekha isolated dataset and 93.22% on Ekush dataset, addressing challenges of complex-shaped characters and similarity between characters.

3.6 Bangla Handwritten Character Recognition using Convolutional Neural Network with Data Augmentation:

This paper offers [32] a Convolutional Neural Network (CNN) architecture that uses Keras with a TensorFlow backend for handwritten character recognition in Bangla. Convolutional and max-pooling layers come first in the CNN model, and then fully linked layers. The output layer uses the Softmax activation function for probabilistic classification, while dropout is used to prevent overfitting. The Stochastic Gradient Descent optimizer and the Categorical Crossentropy loss function are used to construct the model. This study provides a Handwritten Character Recognition (HCR) system specifically designed for Bangla characters. The model uses the Banglalekha-Isolated dataset and achieves 91.81% accuracy on a base dataset of 50-character classes, after utilizing data augmentation techniques to expand the dataset to 200,000 images, the model achieves 95.25% accuracy on a test set.

3.7 Bangla Handwritten Character Recognition Using Deep Convolutional Autoencoder Neural Network:

The paper introduces [29] DConvAENNet, a novel model combining Autoencoder and Deep Convolutional Neural Network (DCNN) for Bangla Handwritten Character Recognition (BHCR). It elaborates on the architecture of the proposed model, highlighting the use of encoder layers for both unsupervised pre-training and supervised learning. The model's layers include convolutional, pooling, and fully connected layers, with ReLU, Sigmoid, and Softmax activation functions. Categorical cross-entropy loss is employed for supervised learning, while binary cross-entropy loss is used for unsupervised learning, both optimized with RMSprop. This approach aims to enhance feature extraction and classification accuracy in BHCR tasks.This research presents a novel method for Bangla Handwritten Character Recognition (HCR) called D ConvAENNet, which combines an Autoencoder with a Deep Convolutional Neural Network (CNN). Experiments on three datasets (BanglaLekha-Isolated, CMATERdb 3.1, and Ekush) yield outstanding accuracy rates of up to 95.53% for the proposed model. The model shows that it is capable of identifying a wide range of Bangla characters, such as modifiers, vowels, consonants and numerals.

3.8 Bangla Handwritten Digit Recognition Using Deep CNN for Large and Unbiased Dataset:

This work [27] uses deep convolutional neural network (CNN) models and the NumtaDB dataset to tackle the problem of handwritten digit detection in Bangla. In the Bengali handwritten digit recognition challenge 2018, the CNN model performed quite well, ranking 13th with a testing accuracy of 92.72%, despite the dataset consisting both unprocessed and augmented images. The preparation procedures used to the NumtaD B dataset for Bangla handwritten digit recognition are described in this portion of the paper. In order to reduce computational complexity, it starts by shrinking the original 180x180 pixel images to a more manageable size of 32x32 pixels and transforming them from RGB to grayscale.

3.9 Handwritten Isolated Bangla Compound Character Recognition: a new benchmark using a novel deep learning approach:

The research [4] tackles problems in the CMATERdb 3.1.3.3 dataset with handwritten Bangla compound character recognition. Previous approaches achieved a 19% error rate by combining SVM with features based on quad trees and convex hulls. DCNNs with layerwise training and RMSProp were introduced in recent works; on CMATERdb 3.1.3.3, they achieved a 10% increase with an error rate of 9.67% and a recognition accuracy of 90.33%.

3.10 A New Deep Learning-Based Handwritten Character Recognition System on Mobile Computing Devices:

Deep learning [5] is highly relevant in current pattern recognition and machine learning due to its potential to solve complex problems.

DL is particularly attractive for mobile devices, opening up opportunities for advanced smart applications. The paper outlines using mobile devices to collect, process, and construct datasets. It proposes a lightweight network structure for Optical Character Recognition (OCR) on specific datasets, considering mobile computing environments and data characteristics. The proposed CNN-based method for OCR is validated through comparisons with existing methods, demonstrating its effectiveness. Due to the large computational requirements of CNNs, the system first trains a lightweight image recognition CNN offline using GPU in the cloud. Key features are located, and images are linearly transformed to align feature points to standard coordinates, with additional grayscale image processing as required.

3.11 Handwritten Tifinagh Character Recognition using Deep Learning Architectures:

The paper discusses [14] the Tifinagh-IRCAM alphabet used for the Amazigh language in North Africa, which was officially normalized in 2001, making it a relatively young alphabet. Handwritten character recognition for Tifinagh is a new area of research. The paper introduces two deep learning approaches, Convolutional Neural Networks (CNNs) and Deep Belief Networks (DBNs), for recognizing handwritten Tifinagh characters. These networks were trained and tested on the AMHCD database. DBNs achieved an accuracy of 95.47%, while CNNs surpassed existing methods with an accuracy of 98.25%. Neural networks (NNs), multilayer neural perceptrons (MLPs), and Hidden Markov Models (HMMs) were also utilized. Deep learning, especially Convolutional Neural Networks (CNNs) and Deep Belief Networks (DBNs), has shown success in various image recognition tasks. CNNs, initially limited by computational costs, were improved with GPU usage, ReLU activation, max-pooling, and dropout regularization. CNNs have been extended to object detection, face recognition, and image retrieval, offering end-to-end training from image to class label. DBNs gained popularity for both unsupervised and supervised machine learning tasks.

3.12 Evaluation of deep learning approaches for optical character recognition in Urdu language:

This paper evaluates [15] deep learning models for Urdu digit recognition using OCR. The models tested include VGGNet16, InceptionV3, ResNet50, and DenseNet121, pre-trained on ImageNet. Testing accuracy results are ResNet50 at 96%, InceptionV3 at 95%, VGGNet16 at 95%, and DenseNet121 at 94%. Along with RNNs like LSTMs, CNN architectures like VGGNet, DenseNet, and ResNet have significantly improved OCR accuracy across languages and styles. ResNet at 97% for text detection, VGGNet16 at 92.38% for Bangla and 99.74% for Balinese, and an RNN-based model at 89.84% for cursive scripts and 98% for Urdu text. These models exhibit accuracy ranging from 92% to over 100% overall.

3.13 Handwritten Marath Consonants Recognition using Multilevel Classification:

The paper presents [16] a method for recognizing handwritten Marathi consonants, achieving recognition accuracy of 78.27% with SVM and 73.29% with k-NN classifiers**.** It uses multilevel classification, image preprocessing, and feature extraction techniques to simplify the recognition process. A database of 7920 samples is used, and the recognition rates for different subclasses range from 60.63% to 86.25% with k-NN and 64.17% to 85.83% with SVM classifiers.

3.14 Handwritten digit recognition using machine learning:

This project explored [19] machine learning and deep learning algorithms for handwritten digit recognition, finding CNN to be the most accurate at 99.25%, with 0.99 Precision, Recall, and F1 Score. In recent research comparing machine learning algorithms like MLP, SVM, and CNN for handwritten digit recognition, CNN consistently performs better than other algorithms, with accuracy reaching up to 99.4%. Hybrid models, such CNN and SVM combined, have shown promise, especially in banking systems and autonomous feature development. With particular layers and configurations, a CNN model constructed on the LeNet-5 architecture achieved 99.32% validation accuracy and 99.5% training accuracy. Additionally, 97.55% training accuracy and 97.17% validation accuracy were attained by an MLP model with a less complex structure.

3.15 Bengali handwritten character recognition using deep learning:

For Bengali character identification, a novel deep neural network [13] achieves 96.8% accuracy in just 11 epochs, surpassing previous work using a larger ResNet50 model. Hassan and Khan utilized local binary patterns in the KNN algorithm, whereas Das et al. employed hand-engineered features to achieve 85.40% accuracy. With CNN-based classifiers, Rumman Rashid Chowdhury obtained over 98% accuracy for numerals and 91.12% accuracy for basic Bengali characters. Although human cleaning was done to ensure data quality, some of the photographs are blank or have inaccurate tags. For validation, 28% of the cleaned dataset was randomly selected to make sure there was no human bias. When the AKHCRNet model achieved 96.80% accuracy on the BanglaLekha Isolated Dataset without the use of ensembling or transfer learning.

3.16 Handwritten Hindi Character Recognition using Multiple- Classifiers in Machine Learning:

Using a dataset of 92,000 photos, we developed [9] an advanced Hindi character recognition system (2HCR) with machine learning approaches. To increase accuracy [9], we employed the LR, LGR, SVM, RF, and NB algorithms. For many applications, handwritten character recognition is crucial, and our Python-based solution was put through a rigorous performance evaluation process to determine accuracy. Organizations have been researching handwritten character recognition since the 1970s, focusing especially on Devanagari characters including basic, combination, hybrid, and numbers. While inaccurately fragmented characters can

occasionally lead to mistakes, methods such as fuzzy logic algorithms, hamming neural networks, and Neural Networks with Self-Organizing Map techniques have been studied for precise detection. It consists of 92 thousand images of 36 primary alphabets and ten digits of Devanagari Script.

3.17 Handwritten Bangla Character Recognition Using Convolutional Neural Network and Bidirectional Long Short-Term Memory:

Handwritten character recognition [18] (HCR), especially for Bangla language, is challenging due to its complex and cursive characters. A hybrid BHCR model combining Convolutional Neural Network (CNN) and stacked Bidirectional Long Short-Term Memory (Bi-LSTM) was proposed. The proposed BHCR model was implemented on Kaggle Kernel, utilizing Nvidia Tesla P100 GPU, 15GB RAM, and 2 GHz Intel Xeon Processor. The CMATER dB dataset with 245 character classes was used, excluding two classes due to low usage and repetition. Raw data was cleaned, organized into class-wise folders, and preprocessed into noise-free, binarized images. Sample images and image preprocessing steps are illustrated in the provided figures. This model used CNN for feature extraction and Bi-LSTM for classification, achieving 96.07% accuracy in recognizing 243 individual Bangla characters.

3.18 BengaliNet: A Low-Cost Novel Convolutional Neural Network for Bengali Handwritten Characters Recognition:

Bengali handwritten character recognition [6] has been less explored compared to other languages due to its complex characters. The model was trained for 200 epochs with a batch size of 512 using the 'Adam' optimizer and categorical cross-entropy loss function. Dropout was applied to prevent overfitting, and SoftMax activation was used for the output layers. The learning rate decreased sequentially from 0.001 to 0.000001 over 100 to 10 epochs.We tested our model on various datasets, including CMATERdb, Ekush, Banglalekha, and NumtaDB, achieving up to 96.49% accuracy. Our approach outperformed previous methods, offering potential for high-performance Bengali character recognition tools. After preprocessing, training and validation sets were prepared and fed into the BengaliNet architecture across eight datasets. The architecture's performance, including class-wise accuracy, precision, recall, and F1-score, is detailed in supplementary files. Test results were computed using Kaggle's GPU environment with specific hardware specifications. BengaliNet achieved average recognition times ranging from 1.7 to 28.6 ms per sample and training times from 0.95 to 26.01 hours across the datasets. Notably, the model's weights were saved post-training, eliminating the need for retraining on unseen datasets, and BengaliNet consistently achieved the highest accuracy across all datasets.

3.19 Recognition of Handwritten Bengali Characters using Low Cost Convolutional Neural Network:

This paper introduces [30] a novel low-cost CNN architecture for recognizing Handwritten Bangla characters, addressing the challenges of complex alignment and character similarity.

The proposed model achieved good accuracy across multiple datasets, with recognition rates of 87% for CMATERdh, 89.6% for Bangla lekha-Isolated, and 83.1% for Ekush. This study also introduced a low-cost CNN architecture for Bengali handwritten character recognition, utilizing just 713,812 parameters and a 32x32 input size. The selected architecture demonstrated improved accuracy across three datasets. While the datasets contained noisy data, potential improvements in accuracy could be achieved through data cleaning. This research paves the way for applications in various Bengali Character Recognition tasks.

3.20 Bangla Handwritten Digit Recognition Using Convolutional Neural Network:

AAM Shahariar Azad Rabby used [19] three datasets. ISI handwritten character database, CMATERDB 3.1.1 and BanglaLekha-Isolated datasets. For ISI handwritten character database, after 30 epoch model gets 90.35% accuracy on the training set. Then tested the model with testing set and got 90.58% accuracy. BanglaLekha-Isolated dataset, after 50 epoch model gets 90.38% accuracy on the training set and 92.93% accuracy on the validation set. Then tested the model with CMATERdb 3.1.1 dataset and got 91.58% accuracy. CMATERdb 3.1.1 dataset, after 30 epoch model gets 90.05% accuracy on the training set and 90.42% accuracy on the validation set. After the train tested the model with the BanglaLekha-Isolated dataset and got 92.65% accuracy

3.21 Bangla Handwritten Character Recognition Using Deep Convolutional Autoencoder Neural Network:

Md Ali Azad performed [20] total of 22 experiments were performed on the three-character datasets (BanglaLekha-Isolated, CMATERdb 3.1, Ekush). All attempts acquire satisfying results up to 90% accuracy for the recognition of Bangla handwritten numerals, vowels, consonants, compound characters, modifiers, and all characters set unitedly. Using this supervised and unsupervised learning technique, our proposed DConvAENNet model achieved 95.21% on BanglaLekha-Isolated for 84 classes, 92.40% on CMATERdb.3.1 for 238 classes, and 95.53% on Ekush for 122 classes.

3.22 Offline Bangla Handwritten Character Recognition with Convolutional Neural Network (CNN):

Md. Zahidul Islam proposed [21] approach have been conducted by using CNN, Python's CV Library and Greedy algorithm. In this approach, CNN is used to train this system to recognize characters and for comparing those used a sequential model. To train this proposed model, with the help of dataset collected from Ekush[30], more than seventy four thousand image data of various categories of Bangla characters have been used. The proposed method gained more than 90% recognition accuracy which is significantly better approach.

3.23 Bangla Handwritten Basic Character Recognition Using Deep Convolutional Neural Network:

Chandrika Saha proposed [22] DCNN model, BBCNet-15 consists of 6 convolution layers followed by reLU activation, 6 max pooling layers, 2 fully connected layers and a softmax output unit. The model is trained for 30 epochs. Different learning rates ranging from 0.0009 to 0.0001 was considered for training. Maximum training accuracy obtained was 90.86% and test or validation accuracy is 91.40%. It gives the learning curve of the model with respect to loss function of train and test data. Lowest loss value obtained for training data is 0.03434 and for test data is 0.1669.

3.24 Handwritten Bangla Character Recognition using Convolutional Neural Networks: A Comparative Study and New Lightweight Model:

Md. Nahidul Islam Opu proposed [23] a new lightweight DL model for HBCR and evaluated its performance. The proposed DL model consists of 74 layers, including sub-layers, and its architecture is divided into five similar blocks. The models were trained based on the CMATERdb dataset and tested with the BanglaLekha, ISI, Ekush, and merged datasets. The accuracy for the EkushNet model was evaluated to be the highest for all test datasets in terms of the alphabet, at 88.67% for BanglaLekha, 93.84% for ISI, 89.50% for Ekush, and 90.08% for merged.

Chapter 4

Methodology

This chapter illustrates the overall technique of our proposed system. Here we describe our working procedure

4.1 Structure of proposed method

The proposed approach includes the three important stages namely: Data Preprocessing step, Training step and Prediction step. Flow diagram is shown in Figure 4.1 and current section includes the brief discussions of the same.

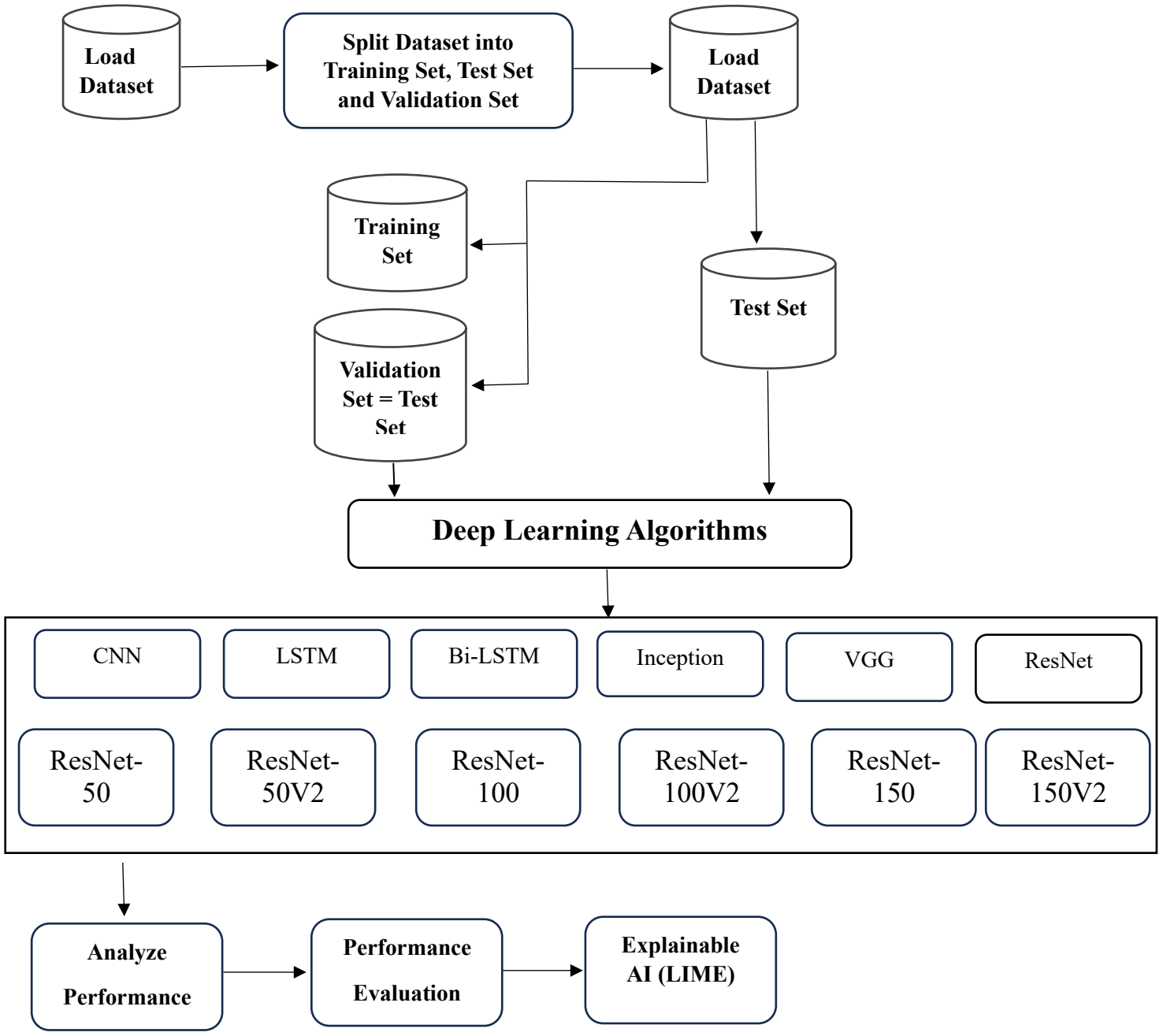

Figure 4.1: Diagram of the proposed method

4.1.1 Data Preprocessing Step

In, data preprocessing for handwritten character database and CMATERDB The dataset is partitioned into 12,000 images for training and 3,000 images for testing, with each image annotated and classified into one of 50 distinct character classes. We utilize the ImageDataGenerator class from the Keras preprocessing module to perform real-time data augmentation and normalization. We rescale the pixel values of the input images to a range between 0 and 1 by dividing each pixel value by 255. We load the input images from the specified directory (train Dir and test Dir) using the "flow_from_directory" method of the ImageDataGenerator class. During loading, the images are resized to a uniform size of 50x50 pixels to ensure consistency in input dimensions across all samples. We convert the color images to grayscale using the color mode='grayscale' parameter. Processes images in batches of 32 during training and testing. Utilizes categorical labels (one-hot encoded) for training and testing. Two dense layers with ReLU activation are added for classification, followed by SoftMax layer for output.

4.1.2Training Step

The data preprocessing is completed, we proceed to train the ResNet-152V2 model on the preprocessed dataset. The model architecture is initialized with an input layer of shape (50, 50, 1), corresponding to the resized grayscale images. The model is compiled using the Adam optimizer and categorical cross-entropy loss function. We train the model using the fit method, specifying the training generator (train generator) and the number of epochs (30) for training. During training, the model learns to extract relevant features from the handwritten character images and optimize its parameters to minimize the categorical cross-entropy loss. The history object captures the training metrics for analysis and visualization.

4.1.3Prediction Step

The model is trained using the fit method. Training data is provided using the train generator. Validation data is provided using the test generator. The trained model is used to generate predictions on the test data. Predicted labels are converted from probabilities to class labels using argmax. Predicted labels are converted from probabilities to class labels using argmax. Test accuracy is calculated using the evaluate method of the model.

4.2 Overview of the Model:

Input: Grayscale images resized to 50x50 dimensions.

Architecture: ResNet-152V2 variant with residual blocks.

Output: Probability distribution over the classes (number of classes determined by the number of subfolders in the training directory).

Training: Adam optimizer, categorical cross-entropy loss function, trained for 3 epochs.

Evaluation Metrics: Accuracy, precision, recall, and F1-score are calculated for each class using the test data.

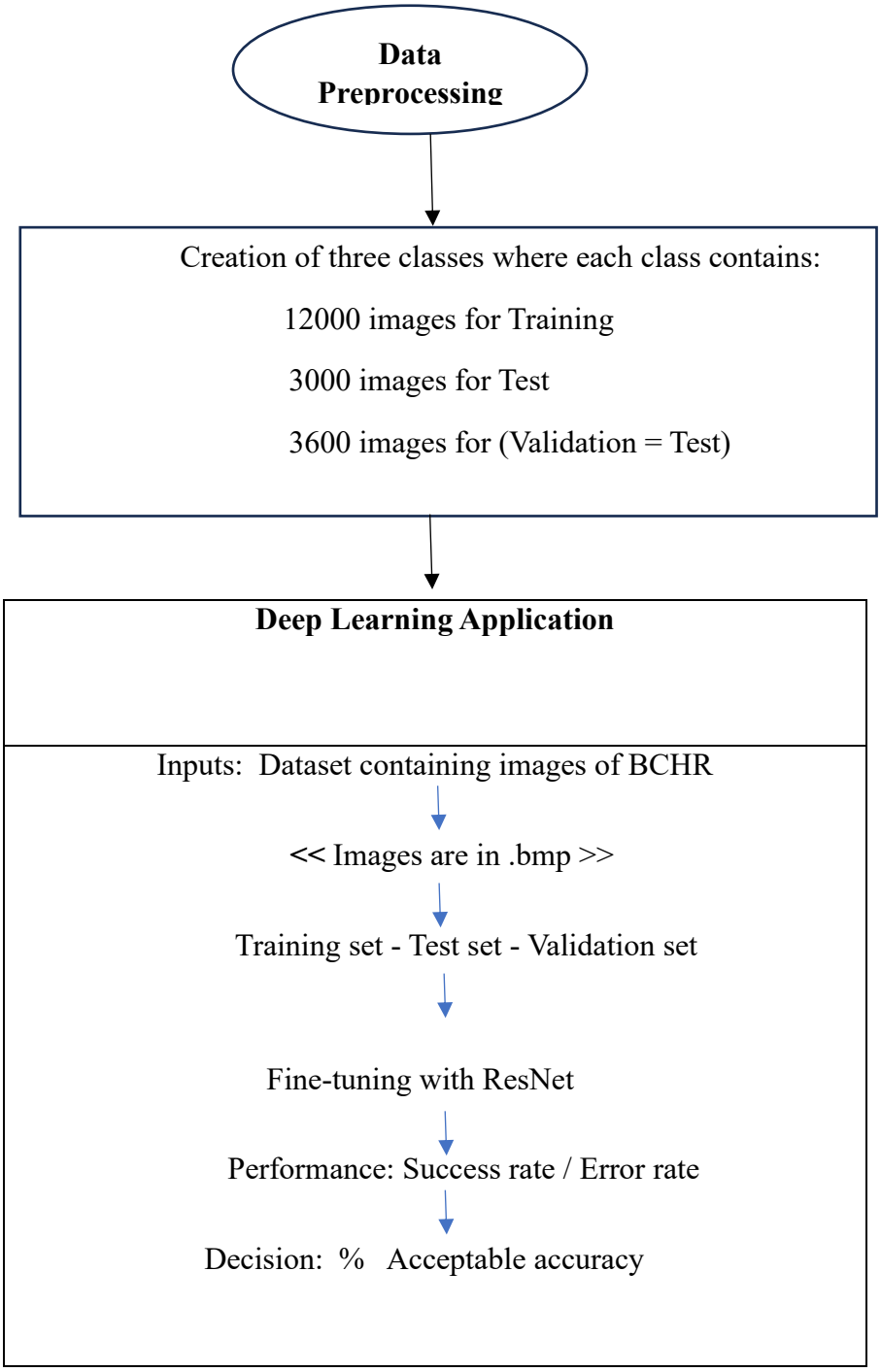

Figure4.2: Diagram of the proposed Overview of the Model

4.4 Proposed ResNet 152V2 Model Architecture:

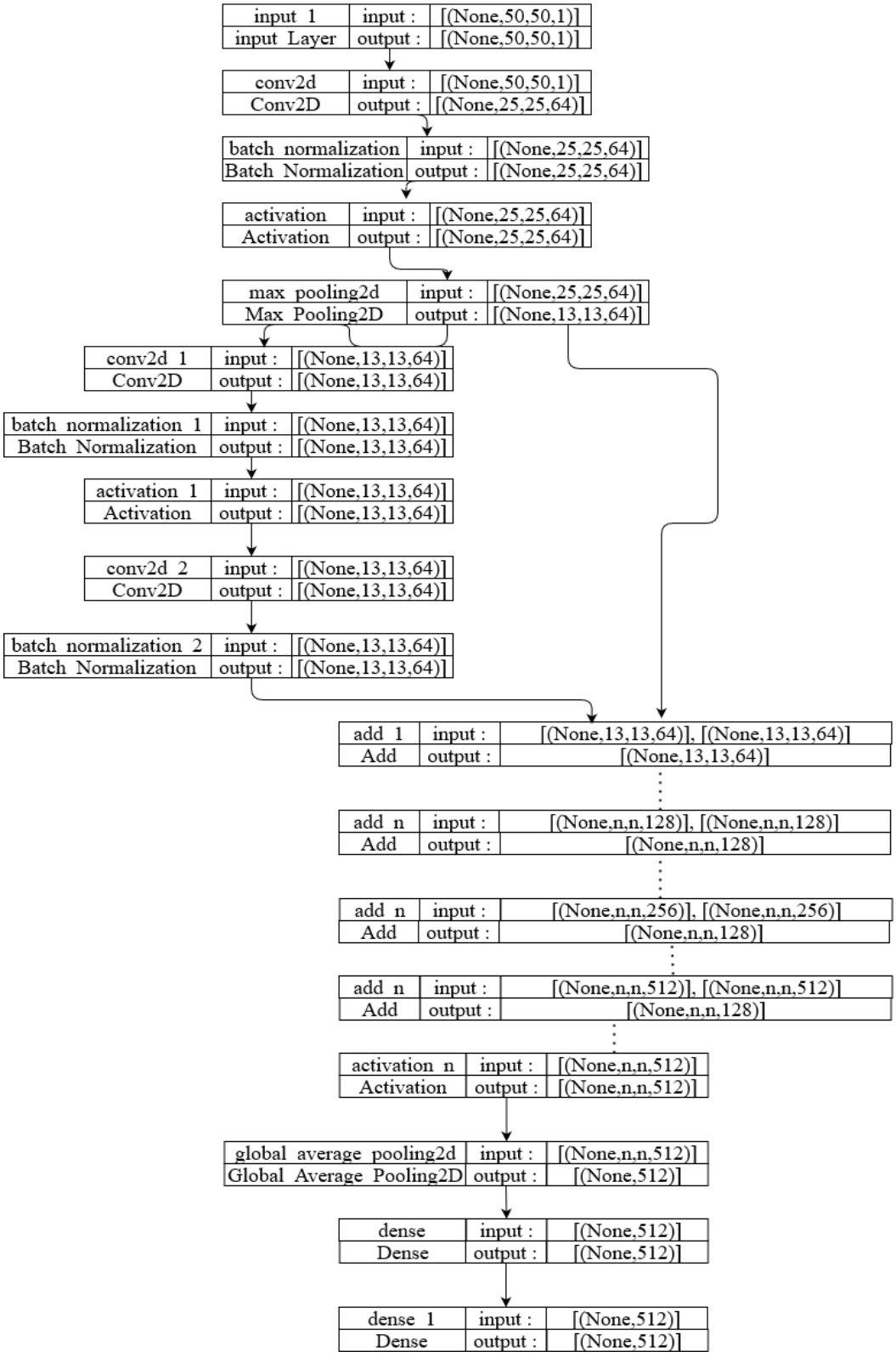

Figure44: Diagram of the proposed ResNet 152V2 Model Architecture

The model architecture is based on ResNet-152V2.Input images are fed into a series of convolutional layers with batch normalization and activation. Residual blocks are used to enable training of very deep networks. Global average pooling is applied to reduce spatial dimensions.

4.5 ResNet Model Architecture

The Residual Blocks idea was created by this design to address the issue of the vanishing/exploding gradient. We apply a method known as skip connections in this network. The skip connection bypasses some levels in between to link-layer activations to subsequent layers. This creates a leftover block. These leftover blocks are stacked to create ResNet.

The strategy behind this network is to let the network fit the residual mapping rather than have layers learn the underlying mapping. Thus, let the network fit instead of using, say, the initial mapping of H(x),

***F(x) := H(x) - x* which gives *H(x) := F(x) + y*.**

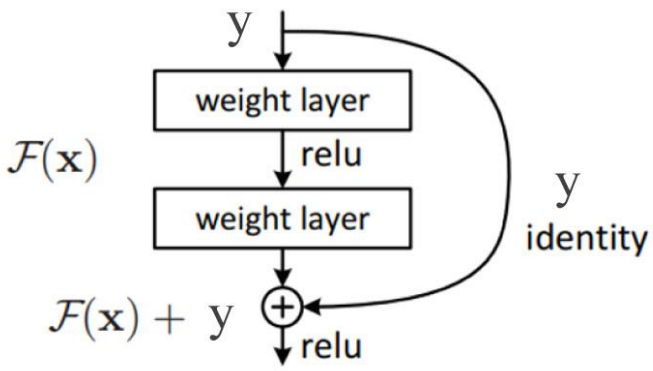

Figure 4.4: Calculation of Weight

The benefit of including this kind of skip link is that regularization will skip any layer that degrades architecture performance. As a result, training an extremely deep neural network is possible without encountering issues with vanishing or expanding gradients.

The VGG-19-inspired 34-layer plain network architecture used by ResNet is followed by the addition of the shortcut connection. The architecture is subsequently transformed into the residual network by these short-cut connections, as depicted in the following figure:

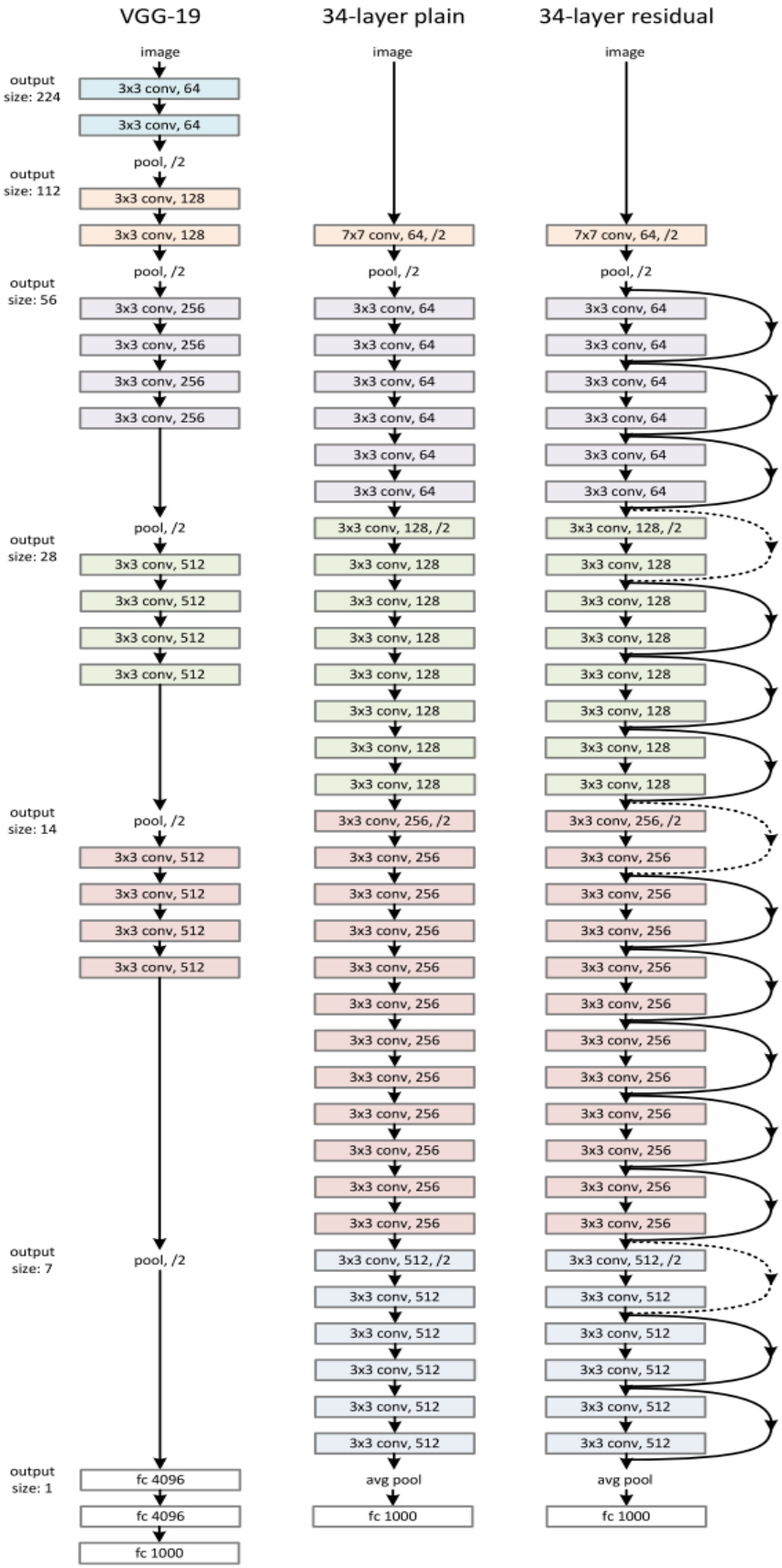

Figure 4.5: Diagram of the ResNet Model Architecture [27]

In this diagram we can see the VGC-19 ,34-layer plain network and 34 layered residual networks. IN this residual network there are total 16 residual blocks.

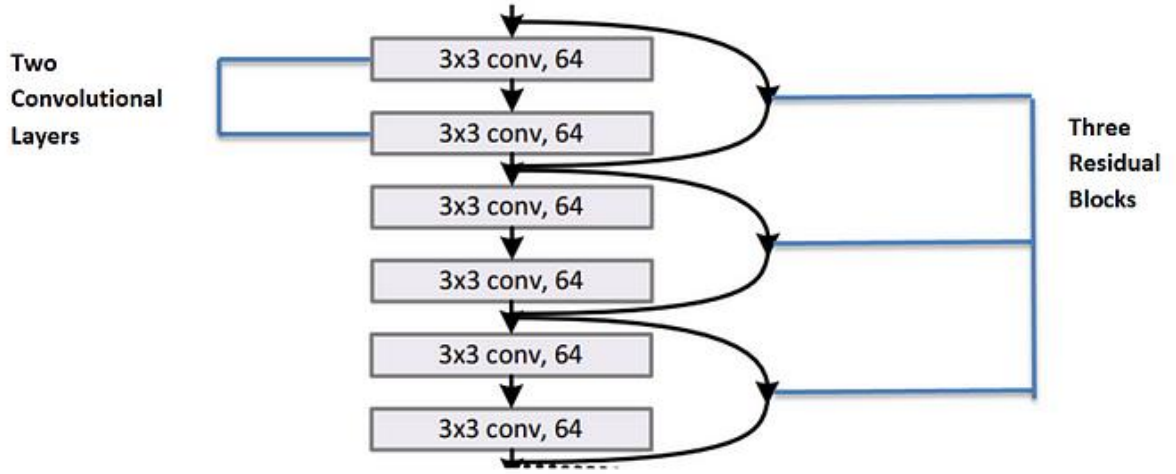

Figure 4.6: Diagram of the Residual blocks-64

The first set consists of 3 residual blocks. Each residual block consists of 2 convolution layers where each convolution layer consists of 64 filters of size 3x3 and a skip connection which performs identity mapping.

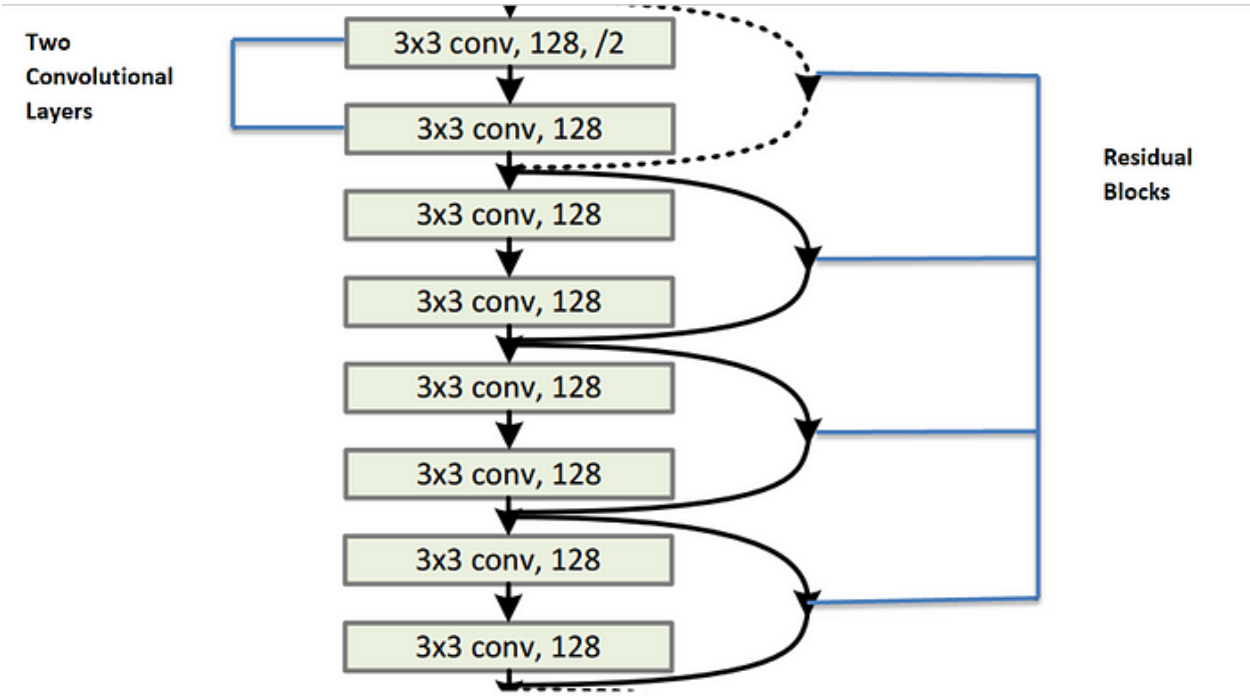

Figure 4.7: Diagram of the Residual blocks-128

Second set consists of 4 residual blocks. Each residual block contains 2 convolutional layers where each layer consists of 128 kernels of size 3x3 and a skip connection. Dotted line skip connections represent the connections when the dimension is increased then it has to match the dimension of the output of convolutional layers. When the dimensions increase (dotted line shortcuts),

we consider two options:
The first is that the shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions. The second is that the projection shortcut is used to match dimensions (done by 1×1 convolutions). For both options, when the shortcuts go across feature maps of two sizes, they are performed with a stride of 2.

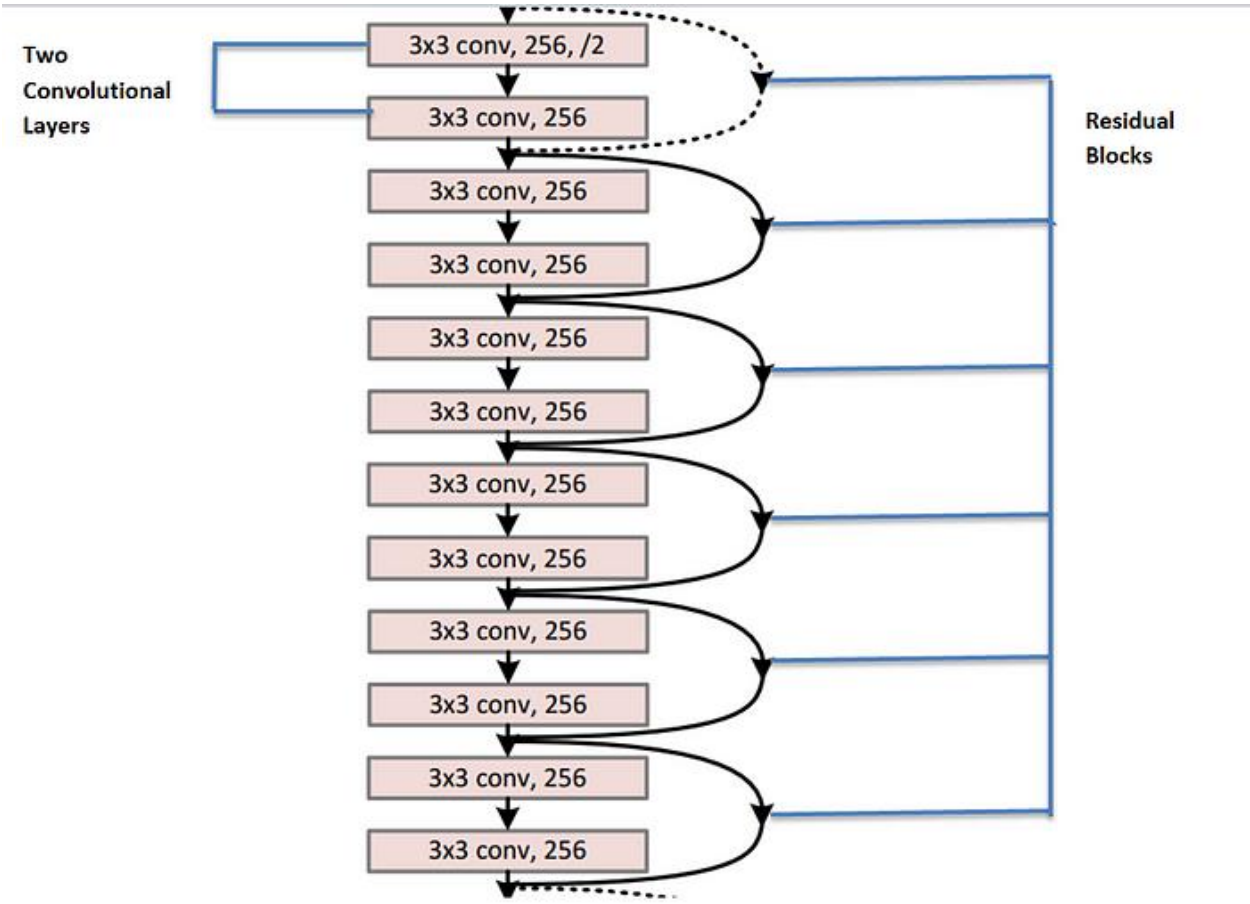

Figure 4.8: Diagram of the Residual blocks-256

Third set consists of 6 residual blocks where each residual blocks contains two convolutional layers. Each convolution layer has 256 filters of size 3x3.

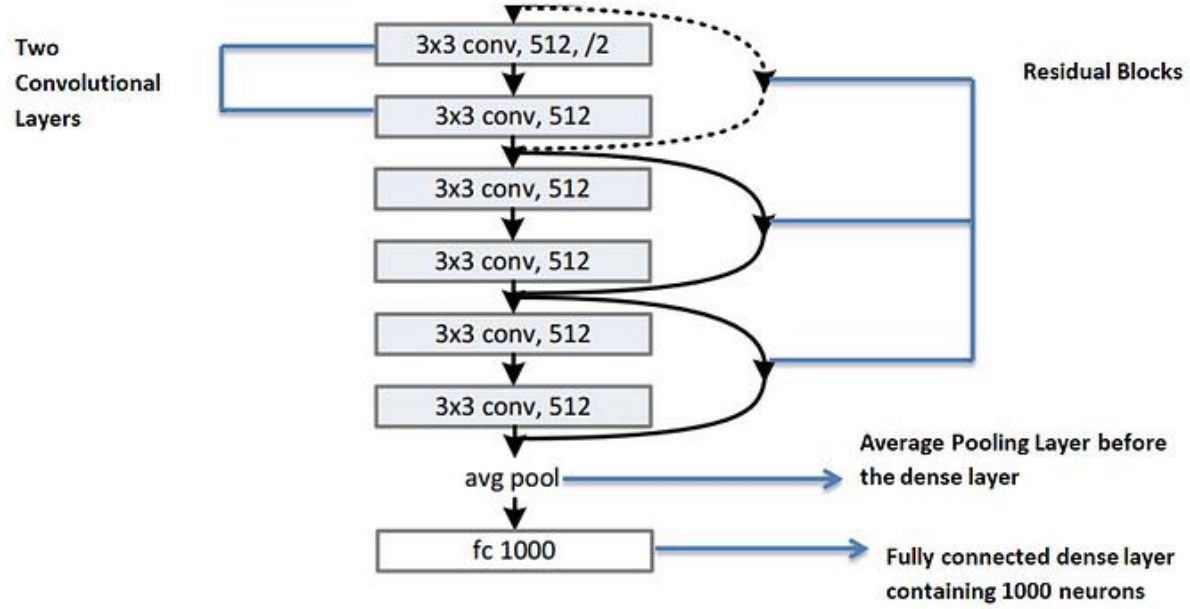

Figure 4.9: Diagram of the Residual blocks-512

The fourth set consists of 3 residual blocks where each residual blocks consists of 2 convolutional layers. Each convolutional layer contains 512 filters of 3x3 each. After that the feature map is passed through average pooling layer and then it is passes through dense layer containing 1000 neurons to classify 1000 classes.

Chapter 5

Experimental Analysis and Results

This chapter illustrate our experiment result. First, we describe about our data set. Then we describe our experimented result.

5.1 Dataset

For evaluation purpose, CMATERdb 3.1.2 [9], [10] dataset is used. CMATERdb is originally a pattern recognition dataset repository prepared at the Center for Microprocessor Ap- plications for Training Education and Research (CMATER) which is a research laboratory of Jadavpur University located at Kolkata in India. The image data of these database have

Figure 5.1: CMATERdb 3.1.2 Dataset

variations in style and some images are little bit noisy. Also the images of characters are not co-centric and size of images varies. It provides the properties of the used dataset CMATERdb 3.1.2 along with the usage of the data for evaluating proposed system. It provides some sample images from the dataset. There are total 50 basic characters in Bangla language including 39 consonants and 11 vowels. CMATERdb 3.1.2 dataset contains two folders for the train and test images. Each folder contains 50 sub-folders.

Table 5.1: Dataset Description

Dataset Name	CMATERdb 3.1.2
Total Number of Images	15000
Total Number of class	50
Total training images	12000
training images per class	240
Total test images	3000
Test images per class	60
Total Validation images	3000
Validation images per class	60

Each of these contains images for the 50 classes and the label of each class is the name of these sub-folders. In train folder, each sub- folder contains 240 images making a total of 12000 images. In test folder, each sub-folder contains 60 images making a total of 3000 images. The Total Validation images of 3000 with each sub-folder contains 60 images and evaluate precision, recall and F1 score for each class.

Figure 5.2: Randomly selected Characters from dataset

In every year this dataset is growing much. Figure demonstrate the growth rate of the data.

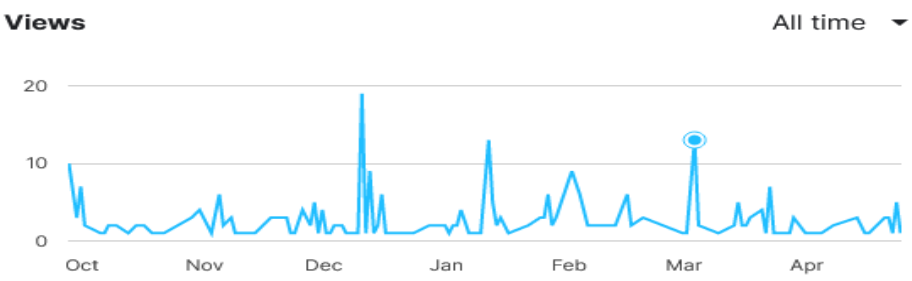

Figure 5.3: Growth rate of View

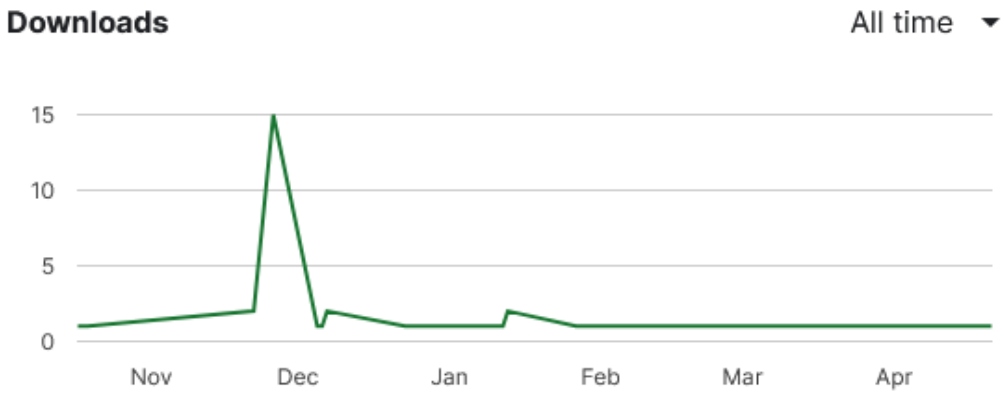

Figure 5.4: Growth rate of Downloads

5.2 Results

5.2.1 Deep Learning Models Analysis

In this experimentation, we have trained some major deep learning models. Such as CNN, Inception, VGG, LSTM, Bi-LSTM And ResNet. These are the best deep learning models for character recognition. We trained all this models for 30 epochs. In these 30 epochs we got our best validation accuracy.

Table 5.2: Deep Learning Models Result and Accuracy

Sequence	Algorithm Name	Result and Accuracy
1	LSTM-Long Short-term memory	82%
2	Bi-LSTM	80%
3	CNN-Convolutional Neural Network	90%
4	Inception	87%
5	VGG-Visual Geometry Group	82%
6	ResNet	92%

ResNet (92%) and CNN (90%) demonstrate the highest accuracy among the tested algorithms, indicating their effectiveness in recognizing Bangla handwritten characters. Inception (87%) also performs well, although slightly lower than ResNet and CNN. LSTM (82%), VGG (82%), and Bi-LSTM (80%) exhibit moderate accuracy levels, suggesting reasonable performance but potentially with room for improvement compared to the top-performing models.

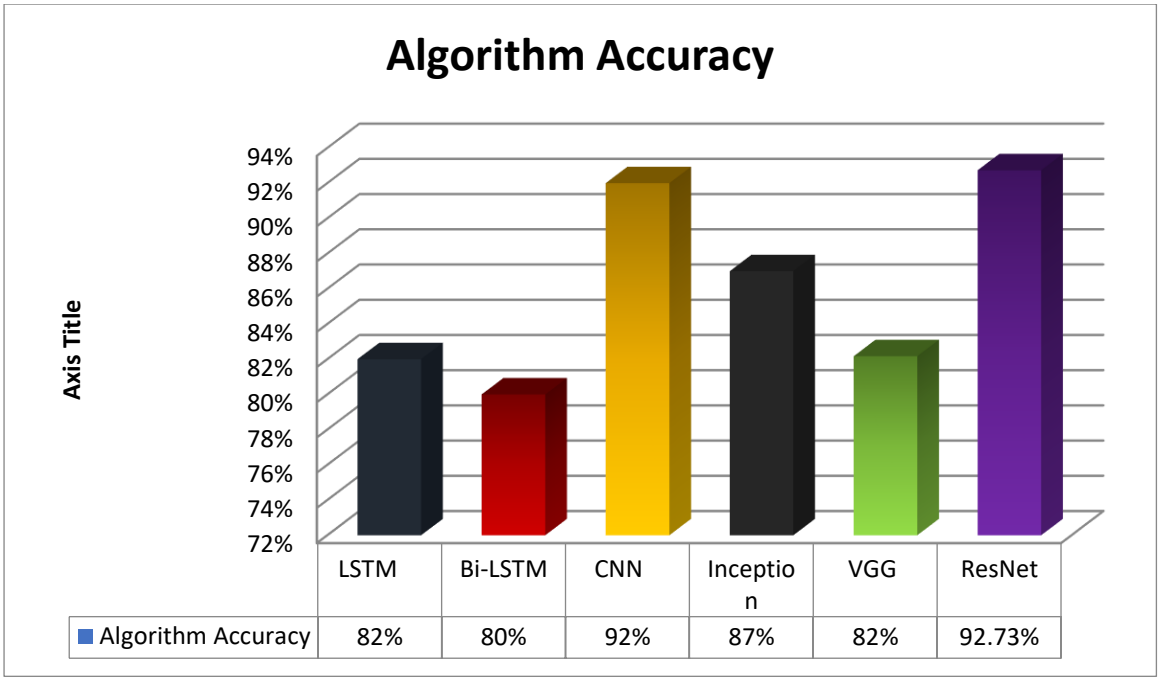

Figure 5.5: Deep Learning Models

ResNet and CNN are particularly suitable for BHCR, given their high accuracy rates. These models likely capture complex patterns and features within Bangla characters effectively.

5.2.2 Model Sustainability

ResNet (92%) and CNN (90%) illustrate the most noteworthy precision among the tried calculations, showing their adequacy in recognizing Bangla written by hand characters. So, Working with ResNet for Bangla Handwritten Character Recognition (BHCR) is a strategic choice due to several compelling reasons. ResNet, short for Residual Neural Network, has demonstrated exceptional performance across various computer vision tasks, making it a promising candidate for complex pattern recognition tasks

5.2.3 Different variants of ResNet architecture

An open-source, Python-based neural network framework called Keras may be used with TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. It is made to make deep neural network experimentation quick. The following ResNet implementations are part of Keras Applications and offer ResNet V1 and ResNet V2 with 50, 101, or 152 layers,

- ResNet50
- ResNet101
- ResNet152
- ResNet50V2
- ResNet101V2

- ResNet152V2

ResNetV2 and the original ResNet (V1) vary primarily in that V2 applies batch normalisation before each weight layer.

5.2.4 ResNet50 and ResNet101 versions of Residual Net

ResNet50 (97.83%) and ResNet101 (98.29%), these models exhibit remarkably high accuracy rates, surpassing the 95% threshold and indicating robust performance in recognizing Bangla characters.

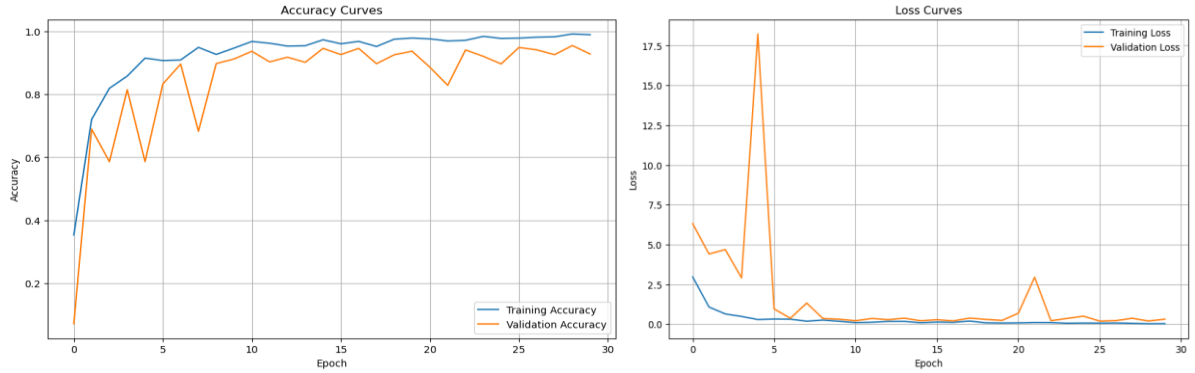

Figure 5.6: ResNet50 Graph of Accuracy Graph and Loss Curve

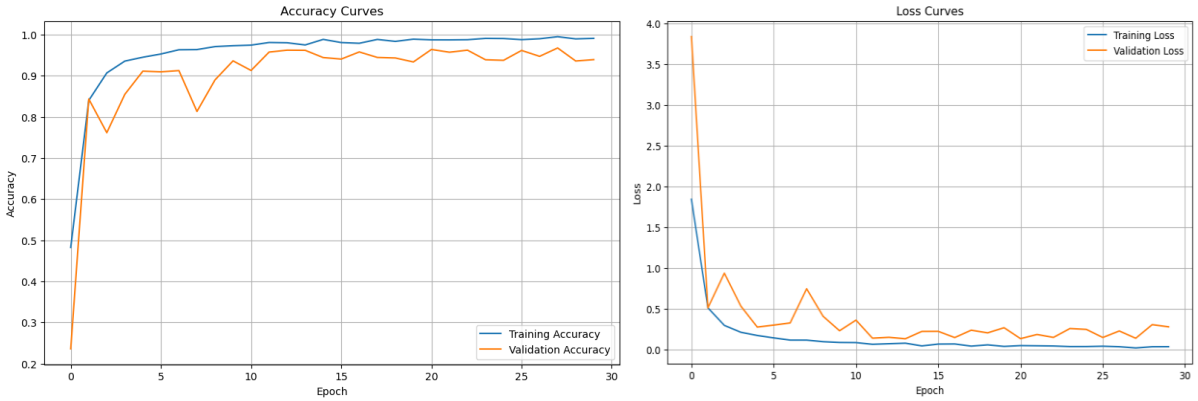

Figure 5.7: ResNet101 Graph of Accuracy Graph and Loss Curve

5.2.5 ResNet50V2 and ResNet101V2 versions of Residual Net

ResNet50V2 (95.60%) and ResNet101V2 (96.73%) offer slightly lower but still impressive accuracies. The "V2" variants of ResNet incorporate improvements and optimizations over the original architecture, potentially enhancing performance in specific contexts or scenarios.

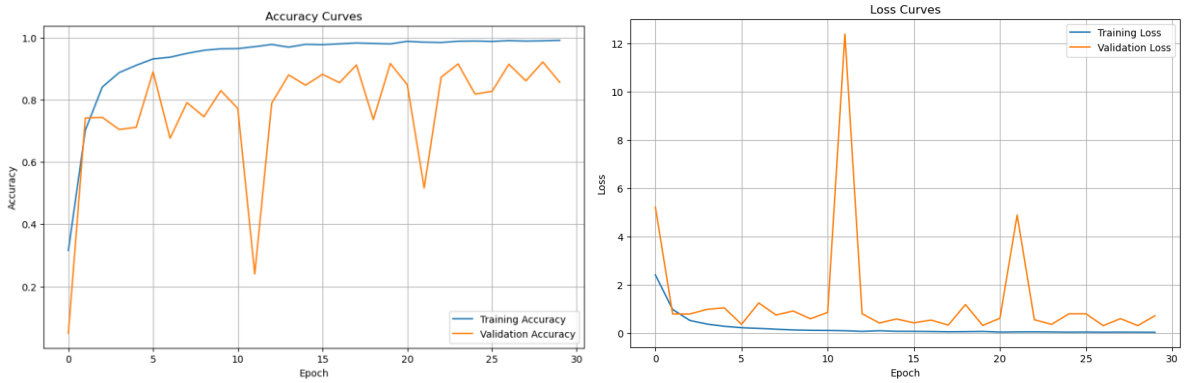

Figure 5.8: ResNet50V2 Graph of Accuracy Graph and Loss Curve

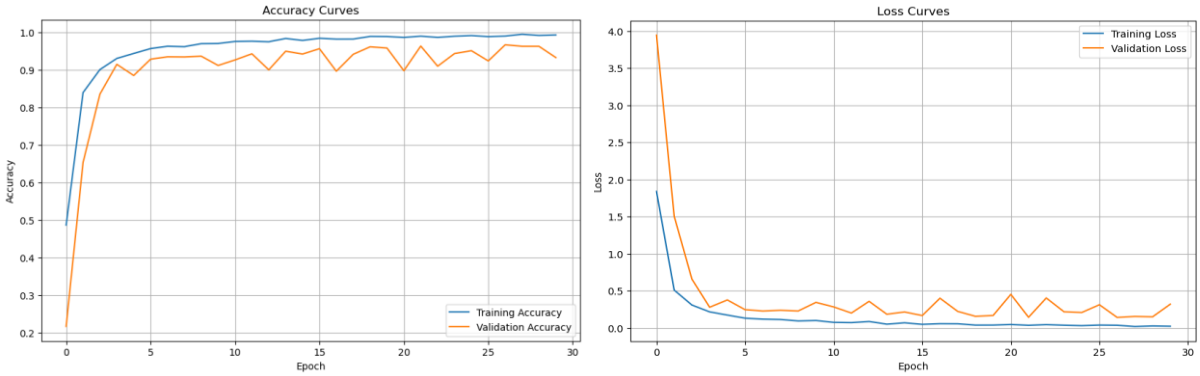

Figure 5.9: ResNet101V2 Graph of Accuracy Graph and Loss Curve

5.2.6 ResNet152 version of Residual Net

ResNet152 (94.40%), although slightly lower in accuracy compared to ResNet50 and ResNet101, remains a strong contender with its deeper architecture capable of capturing more complex patterns and features within Bangla characters.

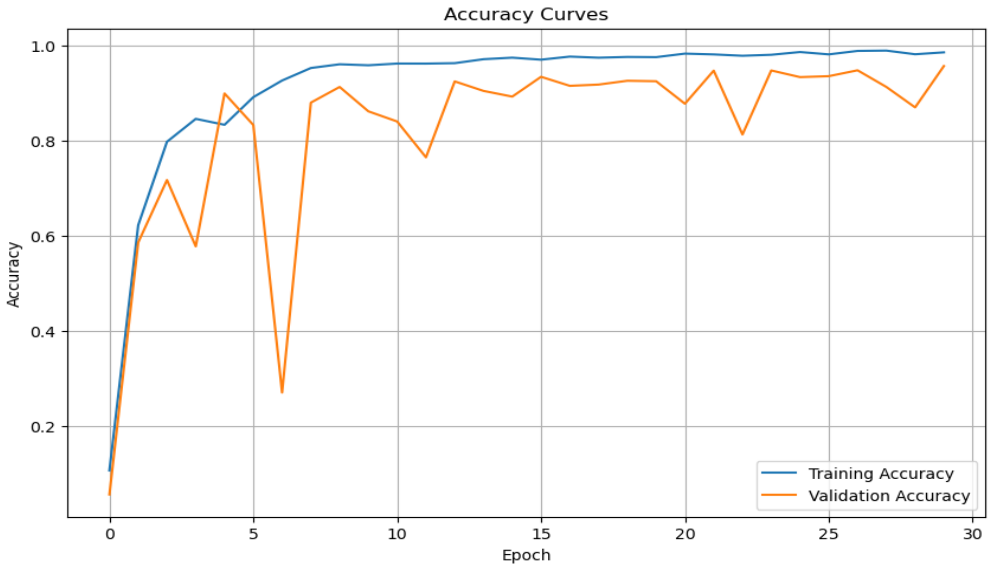

Figure 5.10: ResNet152 Graph of Accuracy Graph

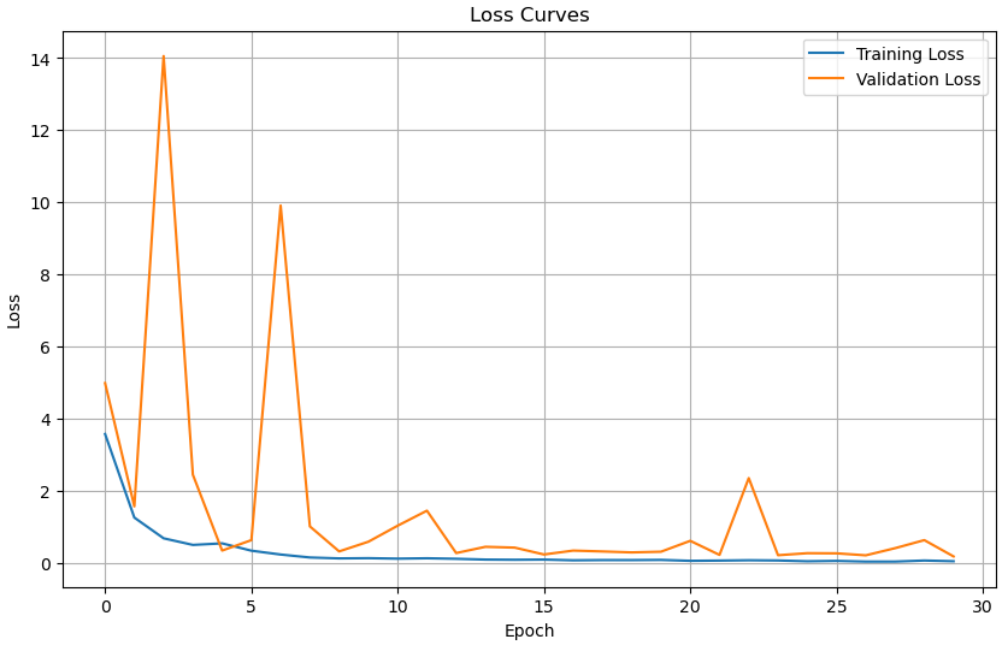

Figure 5.11: ResNet152 Graph of Loss Curve

5.2.7 ResNet152V2 version of Residual Net

The standout performer, the ResNet12V2 (98.76%). This variant, likely referring to ResNet152V2, achieves the highest accuracy among the listed ResNet models, surpassing even the 98.76% mark. ResNet152V2 combines the depth of ResNet152 with additional optimizations, resulting in exceptional performance in BHCR tasks where precision and accuracy are paramount.

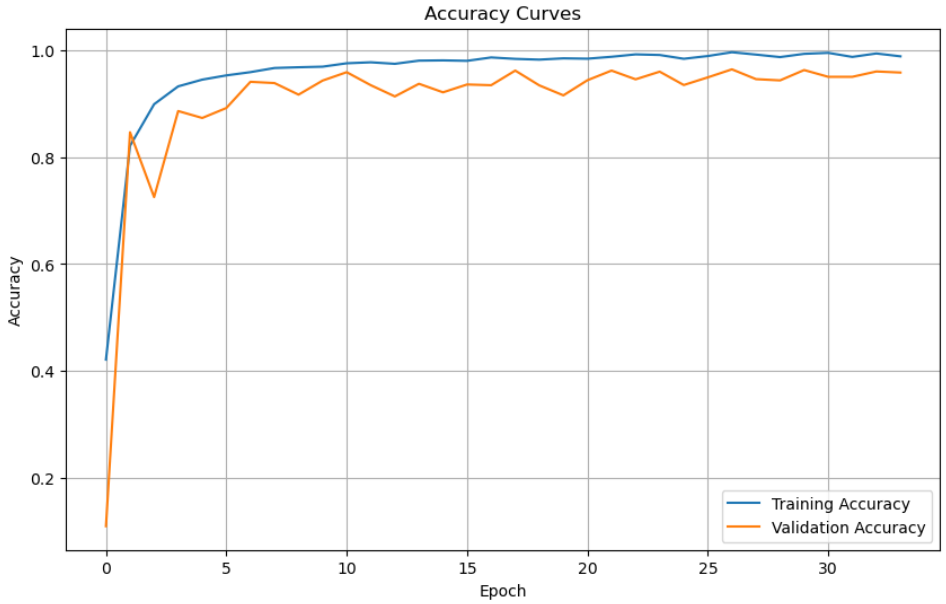

Figure 5.12: ResNet152V2 Graph of Accuracy Graph

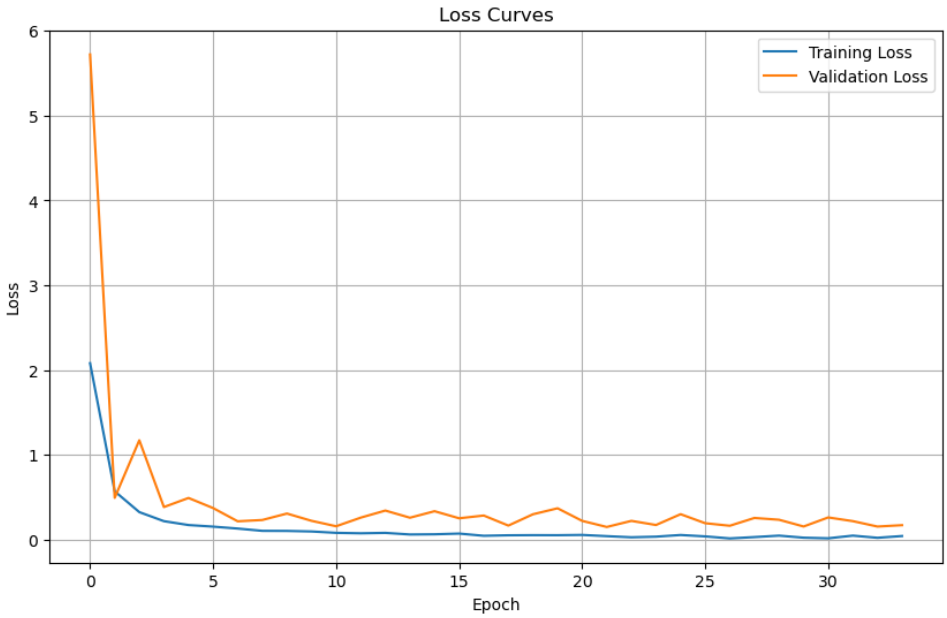

Figure 5.13: ResNet152V2 Graph of Loss Curve

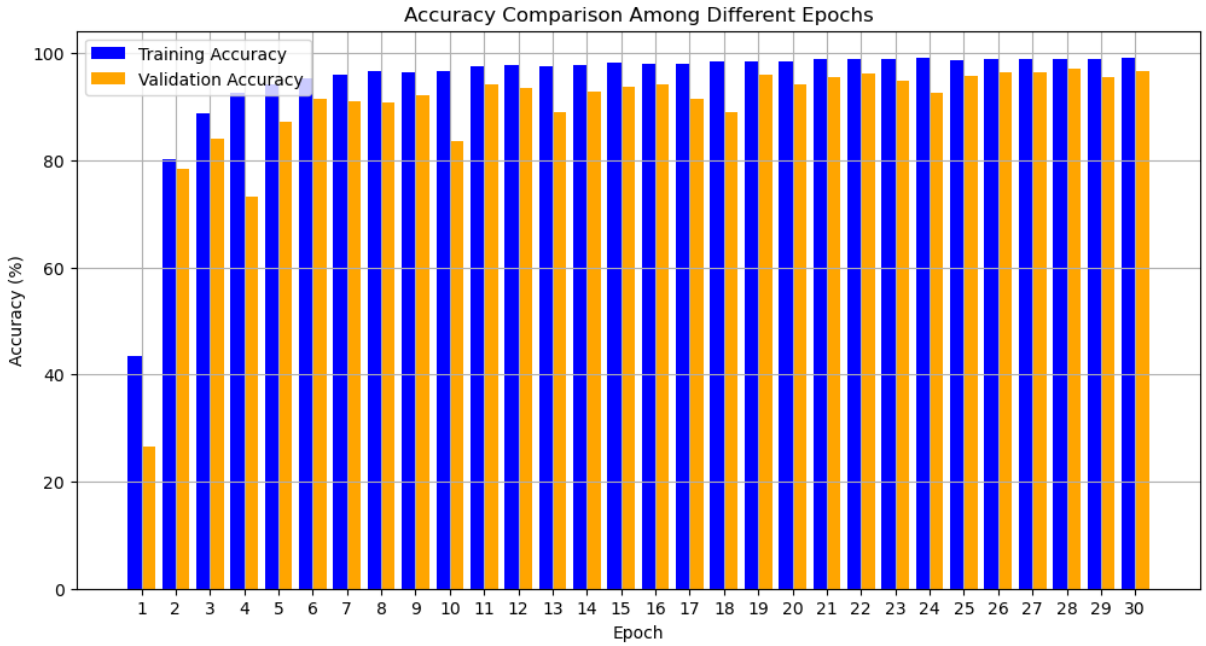

Figure 5.14: Graph of Accuracy Comparison Among Different Epochs

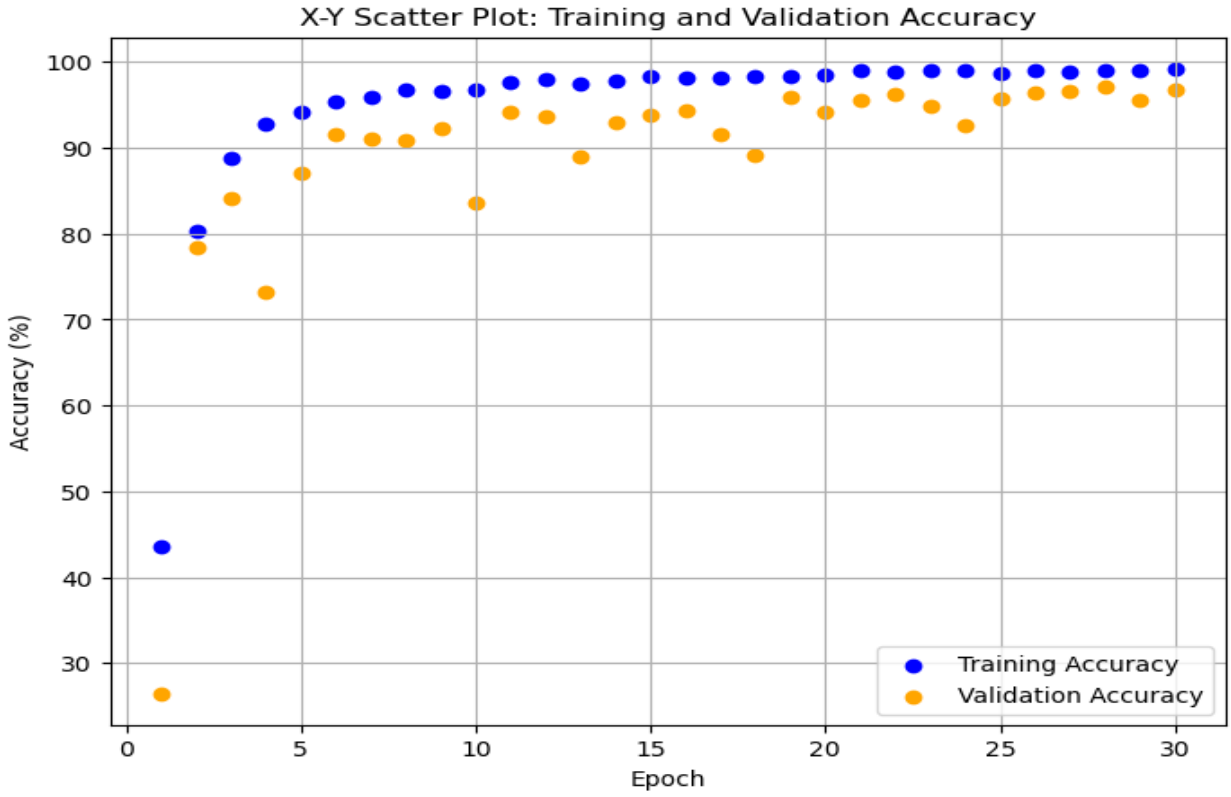

Figure 5.15: Graph of X-Y Scatter Plot

5.2.7 Confusion Matrix

Figure 5.7 represents the confusion matrix. In the confusion matrix, the number of correct and incorrect predictions are compared with count values and divided into each class. Test accuracy and F1score are calculated by the formula:

$$Accuracy = \frac{\text{True Positive+ True Negative}}{\text{True Positive+ True Negative+ False Positive+ False Negative}} \quad (5.1)$$

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall} \quad (5.2)$$

$$Precision = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \quad (5.3)$$

$$Recall = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \quad (5.4)$$

True Positive: The outcome is positive and is predicted to be positive.

False Negative: The outcome is positive but is predicted negative.

True Negative: The outcome is negative and is predicted to be negative.

False Positive: The outcome is negative but is predicted positive.

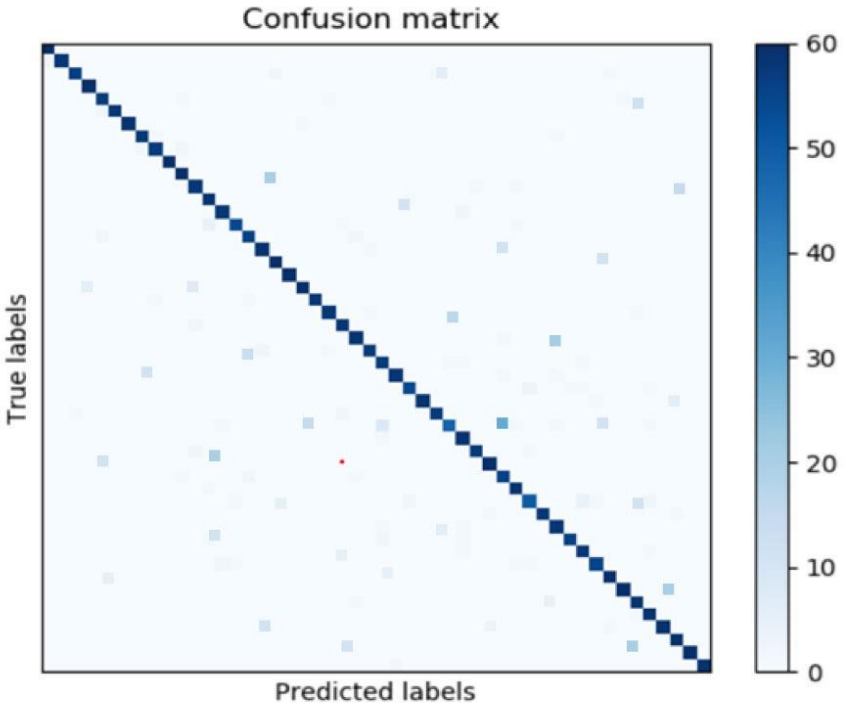

Figure 5.16: Confusion Matrix

Fig. 6 depicts a confusion matrix for all 50 classes. In the confusion matrix, for some classes, lighter color indicates some miss classified data. Similarly, we can see from Fig. 3, some classes have lower F1 scores than other. Some of these characters with a relatively more error rate is given in Fig. 7 with the number of labels.

Actual Class	Misclassified Class
ঘ (15)	খ (13)
ণ (26)	ন (31)
থ (28)	য (37)
ন (31)	ণ (26)
য (37)	ষ, য় (41, 46)

Figure 5.17: Some wrongly classified classes

be noted, all the miss classified classes are structurally very similar especially class 26, 31 and class 37, 41 and 46. As handwriting varies human to human, the similarity of these characters makes it even harder to classify. Despite these facts, the model performed excellently.

Character	P	R	F1	Character	P	R	F1	Character	P	R	F1	Character	P	R	F1	Character	P	R	F1
অ	.97	1.0	.98	ঔ	.94	1.0	.97	ঞ	1.0	.97	.98	ন	.98	.80	.88	ষ	.91	.97	.94
আ	1.0	.97	.98	ক	.94	.97	.95	ট	.97	.97	.97	প	.91	.98	.94	স	.96	.92	.94
ই	.98	.93	.96	খ	.91	.98	.94	ঠ	.95	.97	.96	ফ	.98	.95	.97	হ	.97	1.0	.98
ঈ	1.0	1.0	1.0	গ	.95	.97	.96	ড	.94	.98	.96	ব	.98	1.0	.99	ড়	.98	1.0	.99
উ	.93	.95	.94	ঘ	.96	.90	.93	ঢ	.97	.95	.96	ভ	.97	.93	.95	ঢ়	.97	.98	.98
ঊ	1.0	.97	.98	ঙ	.95	.93	.94	ণ	.83	.95	.88	ম	.94	.98	.96	য়	.92	.98	.95
ঋ	1.0	.98	.99	চ	.97	.98	.98	ত	.98	.97	.97	য	.91	.83	.87	ৎ	1.0	.98	.99
এ	.98	.97	.97	ছ	.97	1.0	.98	থ	.95	.90	.92	র	1.0	.97	.98	ং	.98	1.0	.99
ঐ	.97	.95	.96	জ	1.0	1.0	1.0	দ	1.0	.98	.99	ল	.95	.97	.96	ঃ	1.0	1.0	1.0
ও	1.0	1.0	1.0	ঝ	.98	1.0	.99	ধ	.98	.95	.97	শ	.98	.93	.96	ঁ	1.0	.98	.99

Figure 5.18: Experimental results for each class. (Here, P = Precision, R = Recall and F1 = F1-score)

Precision, recall and F1 score is computed for each classes using the values gained from the confusion matrix which is shown in Fig. 3. Classes are labeled from 1 to 50 while implementing. However, for better illustration printed characters of each class are given in Fig. 3. Average precision obtained is 0.96, average recall is 0.96 and average F1 score is 0.96 all out of 1 which is excellent result considering the complexity of character's shapes and large number of classes. The most miss classified class is class number 37 with a F1 score of 0.87 which is miss classified as class 41 and 46. Class 26 and 31 with F1 score of 0.88 has been miss classified as class 31 and 26 respectively. Class 15 and 28 with F1 scores of 0.93 and 0.92 is confused with class 13 and 37 in some cases. If we observe, it can

5.2.7 ResNet Extended Versions Result

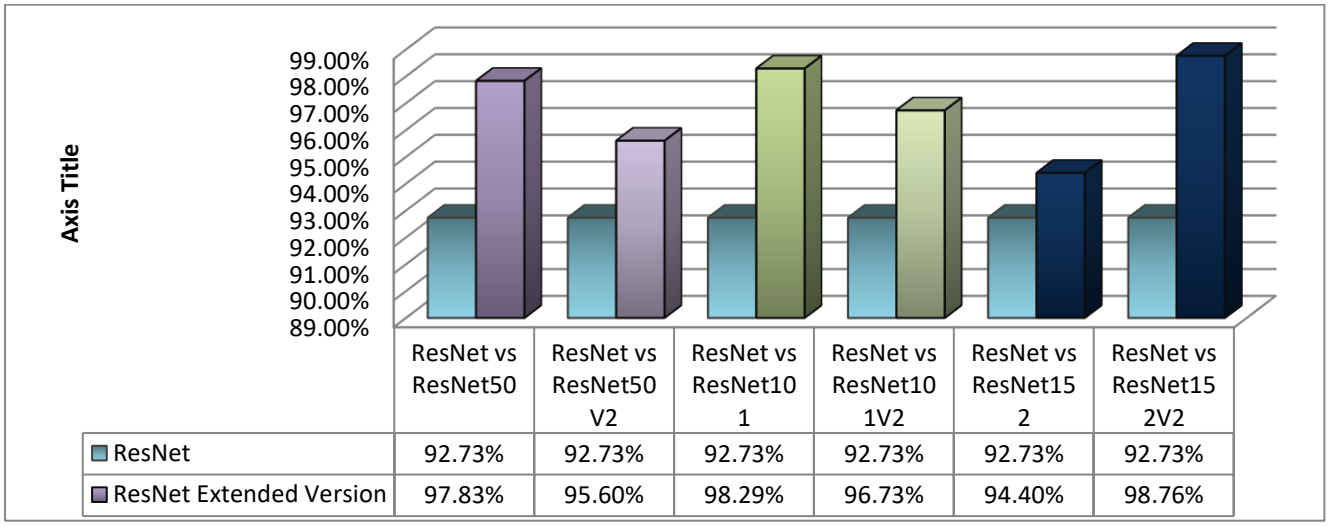

Figure 5.19: ResNet Extended Versions Result, Accuracy and Comparison

ResNet50 (97.83%) and ResNet101 (98.29%), these models display surprisingly high exactness rates, outperforming the 95% edge and showing powerful execution in perceiving Bangla characters. ResNet50V2 (95.60%) and ResNet101V2 (96.73%) offer somewhat lower yet noteworthy exactnesses. ResNet's "V2" variants incorporate enhancements and

Table 5.3: ResNet Extended Versions Result, Accuracy and Comparison

SL	Comparison			
	Algorithm	**Accuracy**	**Algorithm**	**Accuracy**
1	ResNet	92.73%	ResNet50	97.83%
2	ResNet	92.73%	ResNet50V2	95.60%
3	ResNet	92.73%	ResNet101	98.29%
4	ResNet	92.73%	ResNet101V2	96.73%
5	ResNet	92.73%	ResNet152	94.40%
6	ResNet	92.73%	ResNet152V2	98.76%

optimizations over the original architecture, ResNet152 (94.40%), albeit somewhat lower in precision contrasted with ResNet50 and ResNet101, The champion entertainer, the ResNet12V2 (98.76%). Among the listed ResNet models, this variant, likely referring to ResNet152V2, has the highest accuracy, exceeding even the 98.76 percent mark. ResNet152V2 joins the profundity of ResNet152 with extra enhancements.

Table 5.4: Comparison between Our Approach and Others Approachs

Ref. No.	**Language**	**Data Set**	**Algorithm**	**Result and Accuracy**
[19]	Bangla	Handwritten character database, CMATERDB 3.1.1 datasets	CNN	92.65%
[1]	Bangla	NumtaDB	GoogleNet	93%
[25]	Bangla	Bangla basic character database	CNN	95.84%
[31]	Bangla	Bangla numeral image database	CNN	85.96%
[26]	Bangla	Bangalekha isolated dataset, Ekush dataset	CNN	93.22%
[22]	Bangla	Banglalekha-Isolated dataset	BBCNet-15	91.40%.
[29]	Bangla	CMATERdb 3.1	DConvAENNet	97.53%
[14]	Tifinagh	AMHCD database.	CNNs	98.25%,
[20]	Bangla.	BanglaLekha-Isolated, CMATERdb 3.1, Ekush	DCNN	95.53%
[8]	Gujarati	Handwriten images (Set of 10 dataset)	K-NN	82.03%
Proposed work	**Bangla**	**CMATERdb 3.1**	**ResNet152V2**	**98.76%**

5.9 Explainable NLP

Performing LIME (Local Interpretable Model-agnostic Explanations) on a Bengali handwritten character recognition dataset involves using LIME to explain the predictions made by a machine learning model on these images. We can leverage LIME to gain interpretability into the predictions of a Bengali handwritten character recognition model. Adjustments can be made to the preprocessing steps and LIME parameters based on specific requirements and characteristics of the dataset and model.

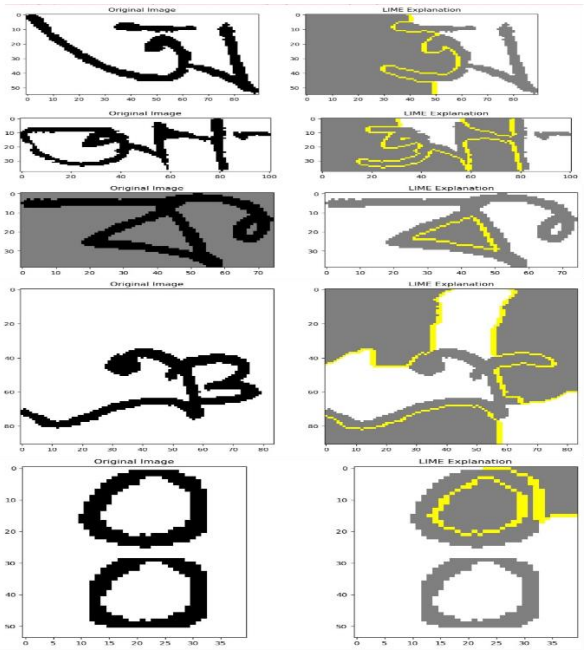

Figure 5.20: LIME explanation process

Additionally, integrating a real prediction function for the model will provide more meaningful insights into the explanation process.

Chapter-6

Conclusion

6.1Conclusion

In this research, Bangla Handwritten Character Recognition (BHCR) successfully accomplished with the help of various classification and feature extraction techniques. A summary of the accuracy, precision, recall, and F1 scores for each classification model used in this study can be seen below. According to our investigation, the ResNet152V2 algorithm has a maximum accuracy of 98.76%. In addition to advancing language processing technology, this research establishes the groundwork for future advancements in image analysis and document digitalization in the Bengali language space.

Our deep learning approach has showcased exceptional capabilities in accurately deciphering the intricate Bengali alphabet for handwritten character recognition in Bangla. We have obtained impressive accuracy rates through rigorous experimentation and sophisticated model modification, demonstrating the ability of deep learning to address the difficulties presented by complicated scripts like as Bangla.

6.2 Limitations

Complex deep learning architectures require significant computational resources for training and inference. It is time consuming. Bengali Handwritten Character poses unique challenges due to its cursive nature and complex character set. While achieving high accuracy rates in controlled experimental settings is crucial, the practical deployment of BHCR systems involves additional challenges, such as real-time processing, robustness to noise, and scalability. These aspects were not explicitly addressed in the study and could be essential considerations for real-world applications. Handwriting styles can vary based on ethnicity, regional dialects, and other factors, potentially affecting model performance across diverse user groups.

Addressing these limitations requires advancements in dataset curation, model robustness, interpretability, and language-specific adaptation to improve the accuracy and reliability of BHCR systems.

6.3 Future Works

In the future, more accurate performance can be obtained if a larger dataset can be used. Bangla digits also need to be classified in future. We have to focus on improving the speed and accuracy of recognition algorithms to enable real-time recognition of handwritten Bangla characters directly on Android devices. Optimization techniques such as model compression, quantization, and hardware acceleration can help achieve this goal. Improving speed, accuracy, and usability of Bangla handwritten character recognition on Android, including

real-time recognition, offline capability, adaptive learning, seamless integration, privacy measures, customization options, multimodal integration, device compatibility, and community engagement for continuous improvement. This will mostly helpful for kids to understand and recognize the characters easily. This application will greatly aid children in easily understanding and recognizing Bangla characters. Its user-friendly interface, engaging visuals, and interactive features make learning enjoyable. With adaptive learning algorithms, children receive personalized guidance, ensuring effective comprehension and mastery of Bangla characters.

[1] Basri, Rabeya, Mohammad Reduanul Haque, Morium Akter, and Mohammad Shorif Uddin. "Bangla handwritten digit recognition using deep convolutional neural network." In *Proceedings of the international conference on computing advancements*, pp. 1-7. 2020.

[2] Chowdhury, Rumman Rashid, Mohammad Shahadat Hossain, Raihan ul Islam, Karl Andersson, and Sazzad Hossain. "Bangla handwritten character recognition using convolutional neural network with data augmentation." In *2019 Joint 8th international conference on informatics, electronics & vision (ICIEV) and 2019 3rd international conference on imaging, vision & pattern recognition (icIVPR)*, pp. 318-323. IEEE, 2019.

[3] Rabby, Akm Shahariar Azad, Sadeka Haque, Sanzidul Islam, Sheikh Abujar, and Syed Akhter Hossain. "Bornonet: Bangla handwritten characters recognition using convolutional neural network." *Procedia computer science* 143 (2018): 528-535.

[4] Roy, Saikat, Nibaran Das, Mahantapas Kundu, and Mita Nasipuri. "Handwritten isolated Bangla compound character recognition: A new benchmark using a novel deep learning approach." *Pattern Recognition Letters* 90 (2017): 15-21.

[5] Weng, Yu, and Chunlei Xia. "A new deep learning-based handwritten character recognition system on mobile computing devices." *Mobile Networks and Applications* 25, no. 2 (2020): 402-411.

[6] Sayeed, Abu, Jungpil Shin, Md Al Mehedi Hasan, Azmain Yakin Srizon, and Md Mehedi Hasan. "Bengalinet: A low-cost novel convolutional neural network for bengali handwritten characters recognition." *Applied Sciences* 11, no. 15 (2021): 6845.

[7] Sadouk, Lamyaa, Taoufiq Gadi, and El Hassan Essoufi. "Handwritten tifinagh character recognition using deep learning architectures." In *Proceedings of the 1st International Conference on Internet of Things and Machine Learning*, pp. 1-11. 2017.

[8] Joshi, Dhara S., and Yogesh R. Risodkar. "Deep learning based Gujarati handwritten character recognition." In *2018 International conference on advances in communication and computing technology (ICACCT)*, pp. 563-566. IEEE, 2018.

[9] Haque, Md Ziaul, and Mohd Omar. "Handwritten Hindi character recognition using multiple classifiers in machine Learning." *Int J Innov Sci Res Technol* 7, no. 7 (2022): 1071.

[10] Shamim, S. M., Mohammad Badrul Alam Miah, Angona Sarker, Masud Rana, and Abdullah Al Jobair. "Handwritten digit recognition using machine learning algorithms." *Indonesian Journal of Science and Technology* 3, no. 1 (2018): 29-39.

[11] Tuffery, Stephane. "Handwriting Recognition." (2023): 237-268.

[12] Somashekar, Thatikonda. "A survey on handwritten character recognition using deep learning technique." *Journal of University of Shanghai for Science and Technology* 23, no. 6 (2021).

[13] Roy, Akash. "AKHCRNet: Bengali handwritten character recognition using deep learning." *arXiv preprint arXiv:2008.12995* (2020).

[14] Sadouk, Lamyaa, Taoufiq Gadi, and El Hassan Essoufi. "Handwritten tifinagh character recognition using deep learning architectures." In *Proceedings of the 1st International Conference on Internet of Things and Machine Learning*, pp. 1-11. 2017.

[15] Riaz, Mehak, Syed Muhammad Ghazanfar Monir, and Rija Hasan. "Evaluation of deep learning approaches for optical character recognition in Urdu language." *Mehran University Research Journal of Engineering and Technology* 41, no. 4 (2022): 146-156.

[16] Patil, C. H., and S. M. Mali. "Handwritten Marathi consonants recognition using multilevel classification." *Int. J. Comput. Appl* 975 (2019): 8887.

[17] Prakash, A. Arun, and S. Preethi. "Isolated offline Tamil handwritten character recognition using deep convolutional neural network." In *2018 International Conference on Intelligent Computing and Communication for Smart World (I2C2SW)*, pp. 278-281. IEEE, 2018.

[18] Fairiz Raisa, Jasiya, Maliha Ulfat, Abdullah Al Mueed, and Mohammad Abu Yousuf. "Handwritten bangla character recognition using convolutional neural network and bidirectional long short-term memory." In *Proceedings of International Conference on Trends in Computational and Cognitive Engineering: Proceedings of TCCE 2020*, pp. 89-101. Springer Singapore, 2021.

[19] Rabby, AKM Shahariar Azad, Sheikh Abujar, Sadeka Haque, and Syed Akhter Hossain. "Bangla handwritten digit recognition using convolutional neural network." In *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2018, Volume 1*, pp. 111-122. Springer Singapore, 2019.

[20] Azad, Md Ali, Hijam Sushil Singha, and Md Mahadi Hasan Nahid. "Bangla handwritten character recognition using deep convolutional autoencoder neural network." In 2020 2nd International Conference on Advanced Information and Communication Technology (ICAICT), pp. 295-300. IEEE, 2020.

[21] Islam, Md Zahidul, Md Abdul Based, and Md Mahbubur Rahman. "Offline bangla handwritten character recognition with convolutional neural network (cnn)." *International Journal of Scientific Research and Engineering Development* 4, no. 1 (2021): 1208.

[22] Saha, Chandrika, Rahat Hossain Faisal, and Md Mostafijur Rahman. "Bangla handwritten basic character recognition using deep convolutional neural network." In *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pp. 190-195. IEEE, 2019.

[23] Opu, Md Nahidul Islam, Md Ekramul Hossain, and Muhammad Ashad Kabir. "Handwritten Bangla character recognition using convolutional neural networks: a comparative study and new lightweight model." *Neural Computing and Applications* 36, no. 1 (2024): 337-348.

[24] Rabby, AKM Shahariar Azad, Md Majedul Islam, Nazmul Hasan, Jebun Nahar, and Fuad Rahman. "Borno: Bangla handwritten character recognition using a multiclass convolutional neural network." In *Proceedings of the Future Technologies Conference*, pp. 457-472. Cham: Springer International Publishing, 2020.

[25] Maitra, Durjoy Sen, Ujjwal Bhattacharya, and Swapan K. Parui. "CNN based common approach to handwritten character recognition of multiple scripts." In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1021-1025. IEEE, 2015.

[26] Hossain, Md Anwar, Mirza AFM Rashidul Hasan, AFM Zainul Abadin, and Nafiul Fatta. "Bangla Handwritten Characters Recognition Using Convolutional Neural Network." *Australian Journal of Engineering and Innovative Technology* 4, no. 2: 27-31.

[27] Shawon, Ashadullah, Md Jamil-Ur Rahman, Firoz Mahmud, and MM Arefin Zaman. "Bangla handwritten digit recognition using deep cnn for large and unbiased dataset." In *2018 international conference on Bangla speech and language processing (ICBSLP)*, pp. 1-6. IEEE, 2018.

[28] Ghosh, Tapotosh, Min-Ha-Zul Abedin, Hasan Al Banna, Nasirul Mumenin, and Mohammad Abu Yousuf. "Performance analysis of state of the art convolutional neural network architectures in Bangla handwritten character recognition." *Pattern Recognition and Image Analysis* 31 (2021): 60-71.

[29] Azad, Md Ali, Hijam Sushil Singha, and Md Mahadi Hasan Nahid. "Bangla handwritten character recognition using deep convolutional autoencoder neural network." In *2020 2nd International Conference on Advanced Information and Communication Technology (ICAICT)*, pp. 295-300. IEEE, 2020.

[30] Jadhav, Rohit, Siddhesh Gadge, Kedar Kharde, Siddhesh Bhere, and Indu Dokare. "Recognition of handwritten bengali characters using low cost convolutional neural network." In *2022 Interdisciplinary Research in Technology and Management (IRTM)*, pp. 1-6. IEEE, 2022.

[31] Chakraborty, Partha, Shanta Roy, Sadia Nowshin Sumaiya, and Aditi Sarker. "Handwritten Character Recognition from Image Using CNN." In *Micro-Electronics and Telecommunication Engineering: Proceedings of 6th ICMETE 2022*, pp. 37-47. Singapore: Springer Nature Singapore, 2023.

[32] Chowdhury, Rumman Rashid, Mohammad Shahadat Hossain, Raihan ul Islam, Karl Andersson, and Sazzad Hossain. "Bangla handwritten character recognition using convolutional neural network with data augmentation." In *2019 Joint 8th international conference on informatics, electronics & vision (ICIEV) and 2019 3rd international conference on imaging, vision & pattern recognition (icIVPR)*, pp. 318-323. IEEE, 2019.