CNN-Based Bangla Handwritten Character Recognition: Exploring Ekush Dataset for Performance Enhancement

BY

Marcel David Baroi
ID: 201-15-3421

This Report Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Tasnim Tabassum
Lecturer
Department of CSE
Daffodil International University

Co-Supervised By

Mohammad Asifur Rahim
Lecturer
Department of CSE
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

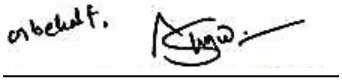
JANUARY 2024

APPROVAL

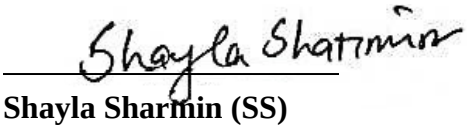
This Project titled "**CNN-Based Bangla Handwritten Character Recognition: Exploring Ekush Dataset for Performance Enhancement**", submitted by Marcel David Baroi to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on January 26, 2024.

BOARD OF EXAMINERS

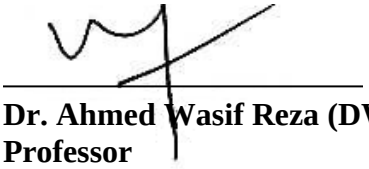
Narayan Ranjan Chakraborty (NRC) **Chairman**
Associate Professor & Associate Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University



Md. Sadekur Rahman (SR) **Internal Examiner 1**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University



Shayla Sharmin (SS) **Internal Examiner 2**
Senior Lecturer
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

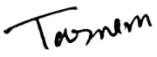


Dr. Ahmed Wasif Reza (DWR) **External Examiner 1**
Professor
Department of Computer Science and Engineering
East West University

DECLARATION

We hereby declare that, this project has been done by us under the supervision of **Tasnim Tabassum, Lecturer, Department of CSE** Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Supervised by:

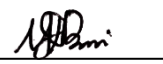


Tasnim Tabassum
Lecturer
Department of CSE
Daffodil International University

Co-Supervised by:

Mohammad Asifur Rahim
Lecturer
Department of CSE
Daffodil International University

Submitted by:



Marcel David Baroi
ID: 201-15-3421
Department of CSE
Daffodil International University

ACKNOWLEDGEMENT

First, I express our heartiest thanks and gratefulness to almighty God for His divine blessing makes us possible to complete the final year project/internship successfully.

I really grateful and wish our profound our indebtedness to **Tasnim Tabassum**, **Lecturer**, Department of CSE Daffodil International University, Dhaka. Deep Knowledge & keen interest of our supervisor in the field of machine learning to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism , valuable advice ,reading many inferior draft and correcting them at all stage have made it possible to complete this project.

I would like to express our heartiest gratitude to Dr. Sheak Rashed Haider Noori, Professor, and Head**,** Department of CSE, for his kind help to finish our project and also to other faculty member and the staff of CSE department of Daffodil International University.

I would like to thank our entire course mate in Daffodil International University, who took part in this discuss while completing the course work.

Finally, I must acknowledge with due respect the constant support and patients of our parents.

ABSTRACT

Bengali handwritten character recognition (BHCR) has not received as much research as other major languages in the world, despite being one of the world's major languages. While a lot of work has been done on Bengali Handwritten Numeral Recognition (BHNR), not as much has been done recently on BHCR. The work primarily uses elementary classification algorithms. In addition to basic classification algorithms, I have attempted to use convolutional networks with the Ekush dataset. I was able to achieve up to 97.93% accuracy using DenseNet and acceptable accuracy using other algorithms. I hope that this comparative overview will assist aspiring researchers in getting started quickly with handwritten character recognition in Bangla.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Introduction

Bangla, the fifth most popular language in the world[1], has 50 characters in its writing system, with several intricate variants. In Bangladesh and certain regions of India, particularly West Bengal, it is widely used. Furthermore, over 265 million people spoke and wrote in Bangla.

This script is used by many different applications due to its large user base, including Writer Identification, Online Banking, National ID Number, License Plate Recognition, Automation of Post Office, Bangla Optical Character Recognition (OCR), and Searching in Historical Manuscripts [2].

One distinctive quality of the Bangla character is called MATRA. MATRA connects the letters horizontally by means of a distinct line at their tops. In the case of BHCR, the classification's result is contingent upon a number of factors, including the classification's accuracy, algorithmic performance, computational resource requirements, processing time, and so forth [3]. To obtain satisfactory results, we must use algorithms of the Rasnet and Demsenet types because the image size is moderately small and the amount of data is large. Kamavisdar, Pooja, et al. [4] outline the benefits and drawbacks of classification algorithms such as ANN, SVM (Support Vector Machine), and DT (Decision Tree). Nevertheless, no empirical findings have been provided; only the theoretical perspective has been discussed. Arun and Vidushi [5] compared several classification algorithms using a multiclass dataset taken from a multispectral satellite image, and they demonstrated that KNN outperforms the other algorithms.

Upon examining the Ekaush dataset, I discovered that numerous researchers have utilized its numerical portion for their investigations, as it yields excellent outcomes for the majority of algorithms. When I initially attempted to use Rasnet 50 and MobleNet V2 for the character portion of the dataset, I received incredibly subpar results. After that, you'll see that the default image size

is much larger than the dataset. I therefore chose more user-friendly algorithms for smaller image sizes, and after removing a few extraneous images, I was able to obtain a good outcome from them.

1.2 Motivation

From a Bangladeshi perspective, the majority of our sectors are going digital. As a result, a large number of handwritten documents need to be digitally converted. Converting a handwritten document to digital format is a difficult and time-consuming process. Historically, scripts were typed by hand, a labor-intensive and expensive process. Bangladesh intends to make Bangla the official language of the government and to digitize every industry. We now need this BHR system, so this is the perfect time.

My goal is to contribute to the BHR's (Bangla Handwritten Recognition System) improvement, as it is an enormous task that is almost impossible for one person to complete alone.

In an attempt to make a tiny contribution, I have attempted to apply various models to an already-existing dataset, since I found it extremely challenging to gather sufficient data for a very successful final study of my own.

I have attempted to apply as many algorithms as I could in light of the limitations of computing power, and the following outcome is covered in the paper.

1.3 Relevance of the Study

There are various ways to explain the significance of this study:

Technological advancement in character recognition

Character recognition technology is essential to many applications in the digital age. It is noteworthy that this study focuses on applying Convolutional Neural Networks (CNNs) to handwritten Bangla character recognition. It contributes to the wider field of optical character

recognition (OCR) by being in line with the continuous advancements in machine learning and AI-based recognition systems.

Culture preservation

Bangladesh's cultural legacy is strongly ingrained in the Bangla language. The development of precise and effective systems for recognizing handwritten Bangla characters contributes to the preservation and advancement of this cultural identity. Furthermore, these systems facilitate Bengali speakers' access to digital resources and tools, which promotes inclusivity and language preservation initiatives.

Dataset exploration and performance improvement

The study's focus on utilizing the Ekaush dataset to improve performance is pertinent because there aren't many publicly accessible annotated datasets made especially for Bangla character recognition. Through performance optimization with this dataset, the research fills a critical resource shortage and advances Bangla character recognition accuracy.

Practical application in education and administration

The study's conclusions have applications, particularly in Bangladeshi educational environments and administrative work. Remarkable efficiency and accessibility can be achieved by using accurate character recognition systems to digitize educational materials, expedite administrative procedures, and make it easier to integrate technology into educational institutions.

1.4 Research Question

This study includes a variety of questions.

- Why is this study being conducted?
- What is the contribution of this study to the related field?
- Why these models are used in conjunction with this dataset?
- What is the significance of this study's outcome?

1.5 Expected Outcome

This system boasts the impressive ability to decipher individual handwritten characters in the beautiful Bangla script, focusing on its fundamental alphabet of vowels and consonants. It explores the subtleties of every stroke and shape, using machine learning models as powerful swords to cut through handwriting style variations and triumph with the accurate character classification. Its potential spans from preserving ancient documents to enriching education and streamlining data entry, unlocking a wealth of possibilities for the Bangla language in the digital realm.

1.6 Layout of the Report

The report consists of six chapters along with a reference part. This part of the report is part of the introduction section, and the following chapters are

1. Chapter 1: Introduction
2. Chapter 2: Background
3. Chapter 3: Research Methodology
4. Chapter 4: Experimental Discussion, Result, and Analysis
5. Chapter 5: Impact on Society, Environment and Sustainability
6. Chapter 6: Overview, Conclusion, and Implementation for Future Research
7. Reference

CHAPTER 2
BACKGROUND

2.1 Terminology

To decipher handwritten Bangla characters, my research explores the fascinating field of convolutional neural networks. Specifically, I use the large Ekaush dataset to achieve significant speedups. I explore the complexities of convolutional layers, optimization algorithms, and feature extraction, using them as tools to overcome problems with the data, such as class imbalance and character segmentation. My ultimate goal is to advance the field of Bangla character recognition and enable smooth digital communication in this lovely language.

2.2 Related Works

Scientists have investigated handwritten character and number recognition in Bangla, which is becoming more and more popular. Various algorithms, including SVM, ANN, Bayes Net, Naïve Bayes, Random Forest, J48, and Random Tree, have been presented in the literature for the purpose of recognizing handwritten characters in Bangla [6].

According to certain studies, classic KNN offers BHCR with exceptional accuracy (95%) [7].

On same data sets, researchers also used Genetic algorithm (GA) for sampling and Support Vector Machines (SVM) for classification with well-known optimization techniques like simulated annealing (SA) and hill climbing (HC) to maximize recognition accuracies [9]. A study demonstrates that the geometric convex hull-based feature, which offers 99.45% accuracy, can be used to effectively recognize isolated handwritten Bangla basic characters and digits [8].

ANN has been used extensively in BHCR recently, particularly in deep learning with convolutional neural networks. Working with MLP neural networks, various quadratic discriminant functions (MQDF, DLQDF), and polynomial network classifiers (PNC, CFPC, SVM), Cheng-Lin Liua and

Ching Y. Suenb discovered that DLQDF, PNC, CFPC, and SVM outperform MLP and MQDF [10].

When CNN with dropout and Gabor features outperforms the most advanced algorithms for HDBR with 98.78% accuracy, BHCR CNN, CNN combined dropout, Gaussian filters, and Gabor filters are also used [11]. Shopon et al. use convolutional neural networks (ConvNet) to achieve nearly perfect accuracy (99.50%) [11].

2.3 Comparative Analysis and Summery

Table 2.3.1 Comparative analysis

Paper No	Author and Year	Dataset	Recognition	Used Models	Height Accuracy Model	Height Accuracy
1	Basu, S., Sarkar, R., Das, N., Kundu, M., Nasipuri, M., & Basu, D. K. (2005).	database of 6000 samples	Digit	Dempster-Shafer (DS) technique		95.1%
2	Alom, Md Zahangir (2018)	CMATERdb,	digits, alphabets, and special character	CNN, VGG16,NiN, ResNet, FractalNet, DenseNet	DenseNet	99.13%

Paper No	Author and Year	Dataset	Recognition	Used Models	Height Accuracy Model	Height Accuracy
3	Shamim, S. M (2018)	dataset provided by Austrian Research Institute for Artificial Intelligence, Austria	Numeral	MPL, SVM, J48, Random Forest, Naive Bayes, Bayes Ne, Random Tree,	MLP	90.37%
4	Al-Amin, Md, Md Saiful Islam, and Shapan Das Uzzalc (2017)	collected more than 16,000 Bengali single line and multiline comments from popular blogging websites	Bengali sentiment classification	word2vec model	word2vec model	75.5%
5	Das, Nibaran (2012)		Numeral	GA, SA and HC	GA	97%

Paper No	Author and Year	Dataset	Recognition	Used Models	Height Accuracy Model	Height Accuracy
6	Liu, Cheng-Lin, and Ching Y. Suen (2009)	ISI Bangla numerals, CENPARMI Farsi numerals, and IFHCDB Farsi numerals	Numeral	MQDF, PNC, CFPC, SVM, DLQDF	DLQDF	99.73%
7	Alom, Md Zahangir(2017)	CMATERdb	Numeral	CNN	CNN	98.78%
8	Noor, Rouhan, Kazi Mejbaul Islam, and Md Jakaria Rahimi.(2018)	NumtaDB	Numeral	LeNet-5, ResNet-18, Proposed Method, Ensemble (K-Fold)	Proposed Method	96.79%
9	Shawon, Md Tanvir Rouf, Raihan Tanvir, and Md Golam	Ekush	Digit	KNeighborsClassifier, SVC, NuSVC, DecisionTreeClassifier, RandomForestClassifier, AdaBoostClassifier, GradientBoostingClassif	Convolutional Neural Network	96.707%

Paper No	Author and Year	Dataset	Recognition	Used Models	Height Accuracy Model	Height Accuracy
	Rabiul Alam. (2022)			ier, GaussianNB, LinearDiscriminantAnalysis, QuadraticDiscriminantAnalysis, Convolutional Neural Network		

2.4 Scopes and Problem

Although there are many well-established datasets and algorithms in the field of Bangla character recognition, my research explores new ground by exploring the special possibilities of the Ekush dataset. In contrast to its predecessors, Ekush provides a richer and more varied sample of complex and nuanced characters, encompassing a greater variety of writing styles. There are opportunities as well as challenges in this undiscovered gold mine. The complex variations are a challenging test for current CNN models, requiring creative modifications and architectural improvements. However, this challenge itself holds the secret to opening up new performance possibilities. By effectively utilizing Ekush's potential, I hope to advance accuracy while also adding insightful new knowledge to the field. My work could lead to the development of a new class of CNNs that are specially trained to deal with the complexities of Bangla script, facilitating smooth online interactions and a deeper comprehension of this beautiful tongue.

2.5 Challenges

Due to inconsistent power supply and restricted processing capacity in Bangladesh, I was forced to reduce the size of my dataset and training regimen, fitting everything into just five epochs. It was an exercise in creativity, a dance with limitations. However, the outcomes were unfavorable,

demonstrating the model's potential even in such a stressful situation. With better resources, the flame will truly ignite—this is just the first spark.

CHAPTER 3
RESEARCH METHODOLOGY

3.1 Design Approach

I have tried to train a simple model with the existing dataset. We used around 50000 individual images to train and 30000 images to test the CNN models. I also converted the test data to a.csv numeric pixel form to use it to train a set of ML algorithms.

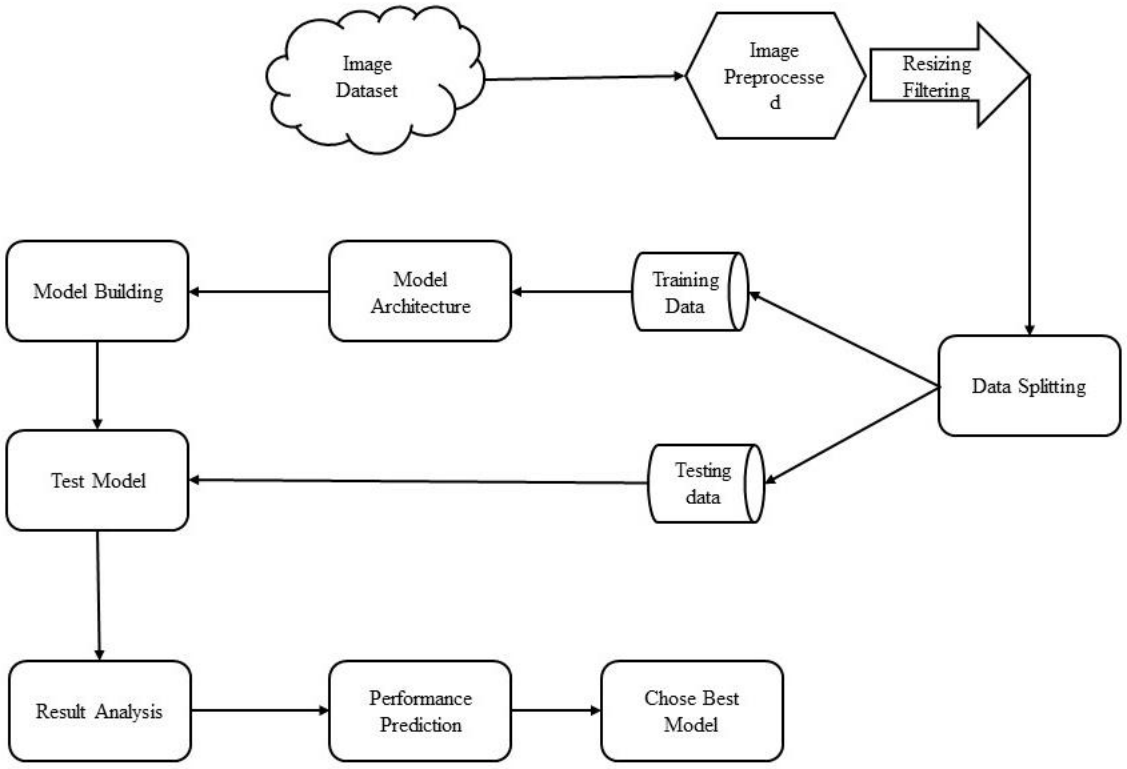


Figure 4.2.1.1Architecture of working process of proposed CNN models

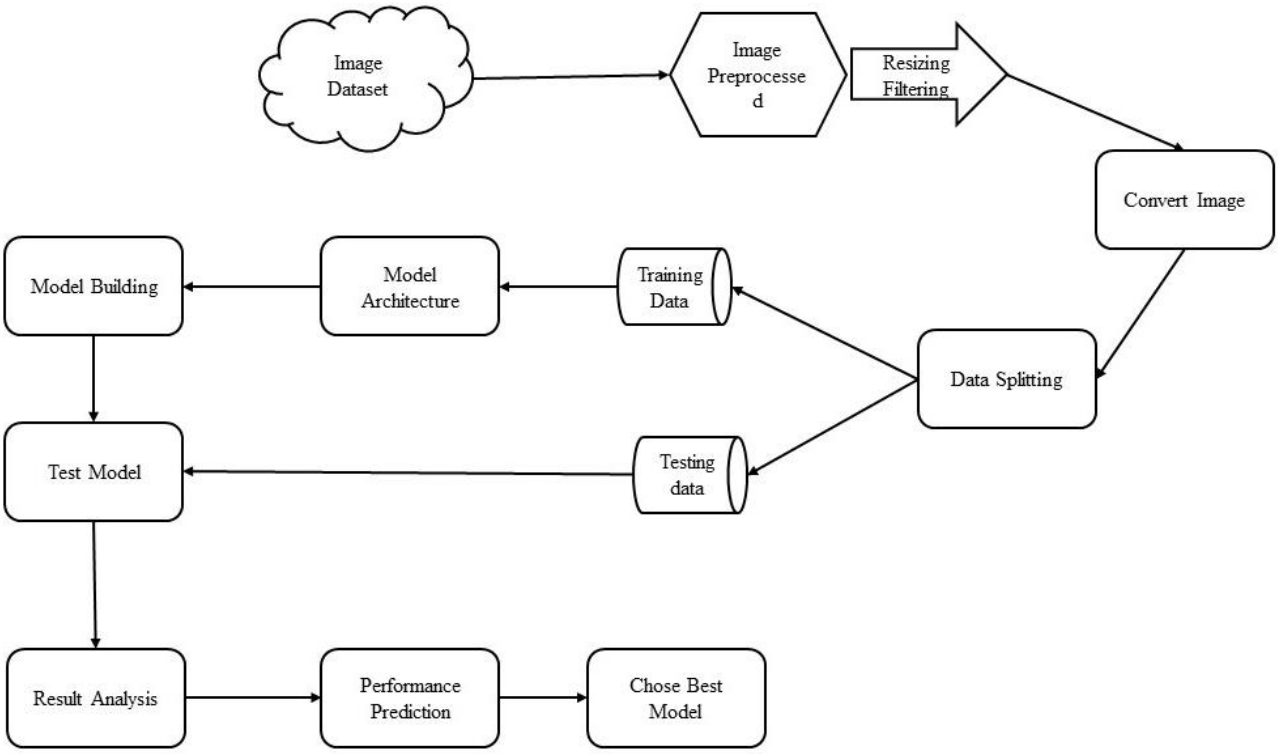


Figure 4.2.1.2Architecture of working process of proposed ML models

3.2 Dataset Description

The Ekush dataset raw image folder has 122 individual folders with characters, digits, compounds, and modifiers.

Table 4.2.1.1 Details of Ekush dataset

Type	Number of Class	Total amount of image
Characters	50	153281
Digits	10	30830
Compounds	52	151607
Modifiers	10	30770

I have worked on the character portion. although it has 153281 images, I just used 50000 images because of lack of computational resources and time.

I have separated the dataset into vowel and consonant and labeled each sub section starting from 0 having same number of images per class and then separately ran the algorithms on them.

Table 4.2.1.2 Details of part of Ekush dataset used in the study

Type	Number of classes	Amount of train data for each class	Amount of test data for each class
Vowel	11	1000	600
Consonant	39	1000	600
Total	50	50000	30000

3.3 Dataset Splitting

Recognizing the different data needs of CNNs and ML algorithms, I adopted specific train-test splits: 5:3 for CNNs to capitalize on their image learning demands, and 8:2 for ML models to avoid overfitting and leverage their data efficiency. This tailored approach balances performance, optimizes data utilization, and avoids biases when comparing these distinct algorithm types.

3.4 Proposed Methodology

I have applied 16 models separately on the vowel and consonant dataset. Among them 3 are convolution neural network models. They are briefly discussed below

3.4.1 Random Forest

Using a multitude of decision trees, Random Forest is a potent ensemble learning algorithm that produces predictions that are more reliable and accurate. It works by first creating a varied forest of trees, each of which has been trained using different feature and data sets. The algorithm is made robust and protected against overfitting by this diversity. Each tree independently voices its

prediction when a new one is required, and the algorithm harmoniously combines these voices—usually by majority voting or averaging—to generate a more definitive result. As a versatile tool in the machine learning toolbox, Random Forest performs admirably in both classification and regression tasks, handling both numerical and categorical data with grace.

3.4.2 KNN

A straightforward yet effective machine learning algorithm, KNN (K-Nearest Neighbors) works on the tenet that "birds of a feather flock together." In a labeled dataset, a new data point is classified by locating its closest neighbors and being assigned the most common class among them. This is how it operates:

1. Training Phase: In actuality, KNN does not "train" in the conventional sense. It merely keeps all of the data points with labels in its memory.
2. Phase of Prediction: KNN uses a selected distance metric, such as Euclidean distance, to determine the distance of each new, unlabeled data point to all training points.
3. Neighbor Identification: Next, based on their distances from the new location, it determines which K neighbors are the closest.
4. Majority Vote: The algorithm looks at the classes of these K closest neighbors and gives the new data point the class that occurs most frequently.

KNN is versatile, handling both classification and regression tasks, and is often used for pattern recognition, recommendation systems, and image processing.

3.4.3 SVM

Strong supervised learning algorithms like SVM (Support Vector Machine) are excellent at identifying decision boundaries that neatly divide data points into different classes. To ensure reliable classification, it maps data points into high-dimensional feature spaces and builds a hyperplane that maximizes the margin between those classes. SVMs use kernel functions, which deliberately split data into discrete forms, to handle non-linear data. It is renowned for being

accurate, adaptable, and capable of handling high-dimensional data even with a small number of training samples in both regression and classification applications.

3.4.4 Logistic Regression

When there are only two possible outcomes (such as "yes" or "no"), logistic regression is an effective machine learning method that predicts the chance that an event will occur. For this reason, it is the algorithm of choice for binary classification tasks. It calculates the likelihood of each class by fitting a logistic function—a sigmoid-shaped curve—to the data. Through a procedure known as optimization, the algorithm determines a set of coefficients and modifies the curve to best distinguish between the various classes in the training set. It uses the learnt model to calculate the probabilities of new data points, assigns them to the class with the higher probability (usually 0.5) and makes predictions. Because of its interpretability, effectiveness, and adaptability, logistic regression is a widely used technique in a variety of industries, including the social sciences, healthcare, finance, and marketing.

3.4.5 AdaBoost

Adaptive Boosting, or AdaBoost, is a potent ensemble machine learning technique that creates a strong classifier iteratively by combining several weak ones. It operates by gradually training weak classifiers on various iterations of the data, giving more weight to instances that were initially incorrectly classified. The final prediction is obtained by adding the weighted votes of all the classifiers, with each weak classifier being given a weight determined by its accuracy. When compared to solo models, this collaborative approach frequently results in much better performance.

3.4.6 Naive Bayes

The Naive Bayes classification algorithm is based on the assumption that features in a dataset are independent of one another and uses probability to generate predictions. During its training phase, it first determines the likelihood that each feature will appear within each class. Then, using the Bayes theorem, it determines the probability that newly acquired data will belong to each possible

class. The final class selected for prediction is the one with the highest likelihood. This method is well-liked for applications such as text categorization, sentiment analysis, and spam filtering because of its speed, simplicity, and frequently unexpected efficacy.

3.4.7 Extra Trees

Extra Trees, often referred to as Extremely Randomized Trees, is a potent ensemble machine learning technique that combines many decision trees' predictions for improved accuracy and robustness, effectively utilizing the wisdom of crowds. It works by building each tree separately and adding extra unpredictability to the process to increase diversity and decrease overfitting:

1. Random Sampling: To ensure distinct viewpoints and little correlation amongst trees, every tree is trained on a random subset of data.
2. Random Feature Selection: To avoid a few dominating features unduly impacting the trees, a random subset of features is taken into consideration at each split.
3. Random Split Points: To further diversity the ensemble, Extra Trees randomly selects split points for each feature rather than carefully selecting the best split point.

These elements combine to create an ensemble that often outperforms individual decision trees and rivals the popular Random Forest algorithm, while often being faster to train.

3.4.8 Bagging Classifier

By training several models on various, randomly chosen portions of the training data and then aggregating their predictions through voting or average, the Bagging Classifier maximizes the power of the collective. Seeking guidance from a diverse council of experts, each with specialized knowledge on a certain topic, is akin to that. Particularly for decision tree-type models that are prone to overfitting, this variety helps lower variance and overfitting, resulting in more precise and reliable predictions.

3.4.9 Decision Tree

In order to make a prediction or classify data, decision trees ask a series of questions about its properties, simulating human decision-making. Imagine a tree with branches, where the leaves represent the results, the trunk represents the complete dataset, and each branch split represents a decision point determined by the value of a feature. The program builds an effective tree structure by carefully selecting the most informative splits. It is highly valued for its interpretability and capacity to handle both numerical and categorical data. It is a flexible tool for both regression and classification applications.

3.4.10 Stochastic Gradient Descent

When working with large datasets, Stochastic Gradient Descent (SGD) is an agile technique that quickly determines the ideal model parameters for machine learning applications. It smartly adjusts parameters based on randomly picked individual samples, lowering processing cost and accelerating the learning process, rather than utilizing the complete dataset to calculate gradients at each step. Even though it introduces some noise, this iterative method can escape local minima and frequently results in faster convergence. SGD is a key component of neural network training and a potent tool in the machine learning toolkit because of its versatility in solving large-scale issues and online learning.

3.4.11 Nearest Centroid

Nearest centroid the "center" of each class in a dataset is initially determined by the algorithm, which is then represented by a single point known as a centroid. A fresh data point is assigned the class of the closest centroid after its distance from each centroid is determined. Assuming that data points from the same class tend to cluster together, classification is carried out in this manner based on proximity to these representative points.

3.4.12 Linear SVM

By maximizing the margin between two classes in a dataset, linear support vector machines (SVM) seek to identify the ideal hyperplane that establishes a distinct classification boundary. In order to accomplish this, it concentrates on the support vectors—data points that are closest to the dividing line—and positions the hyperplane to create the greatest possible separation between them. Due to their focus on maximizing margins, SVMs are resistant to outliers and frequently produce better generalization on unknown data. In order to accomplish this, the algorithm solves a limited optimization problem by looking for a solution that maintains the largest margin while still separating classes.

3.4.13 Multinomial Naive Bayes

Multinomial naive the "naive" assumption that words in a document are independent of one another is the basis for the Bayes text classification algorithm, which applies the Bayes theorem. Because of its proficiency with discrete word counts, it is frequently used for tasks like topic categorization, sentiment analysis, and spam detection. Based on the frequency of words in those classes in the training data, the algorithm determines the likelihood of each potential class for a given document. The document is then allocated to the class with the highest likelihood.

3.4.14 CustomCNN

A customized convolutional neural network (CNN) architecture called CustomCNN offers flexibility and potential performance gains over standard models. It is made to address a particular task or dataset. It entails carefully arranging filters, layers, activation functions, and other parameters in order to match the problem at hand precisely. Important actions usually consist of:

1. Input Layer: Takes in unprocessed image data and resizes it to fit the processing needs.
2. Convolutional Layers: Utilizing learnable filters, extract features to create feature maps that emphasize textures and patterns.

3. Pooling Layers: Downsample feature maps to cut down on computational overhead and dimensionality without sacrificing important data.
4. Fully Connected Layers: Combine features from earlier layers to carry out tasks like regression or classification.
5. Output Layer: Generates probabilities or final prediction scores for every potential class.

The architecture of the model is carefully planned to achieve the best accuracy and efficiency for the given problem, using domain expertise and experimentation as guidance.

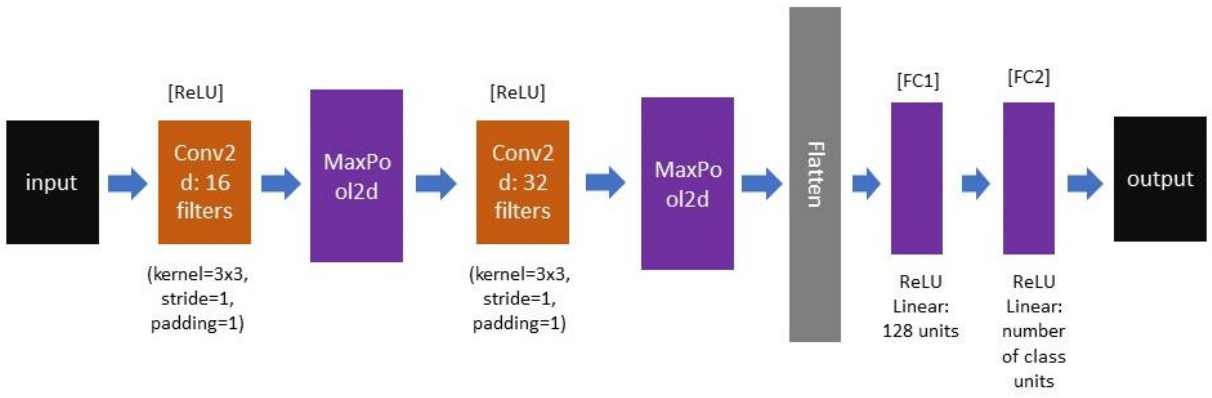


Figure 4.2.1.1Structure of the custom CNN

3.4.15 DenseNet

Dense Blocks and Transition Layers are the two main building blocks on which DenseNet lays its magic. Envision convolutional layers stacked in blocks, where each layer feeds its own features to every other layer in the block in addition to receiving information from the layer before it. As a result, a comprehensive understanding of earlier features can be built upon by later layers, creating a rich tapestry of information sharing. Transition Layers serve as bottlenecks between blocks, limiting the size of feature maps and averting computational explosions to prevent information flooding. In comparison to conventional architectures, the outcome is a highly interconnected network with reduced vanishing gradients, better feature propagation, and surprisingly fewer

parameters. This effectiveness keeps resource requirements reasonable while opening up deeper networks with greater accuracy. DenseNet's tightly packed information flow also encourages feature reuse, which lowers redundancy and improves overall performance by leveraging valuable features that are identified early in the network. In summary, DenseNet is a deep network that unifies features like a tapestry to enable accurate and efficient vision tasks. It is a masterwork of information sharing.

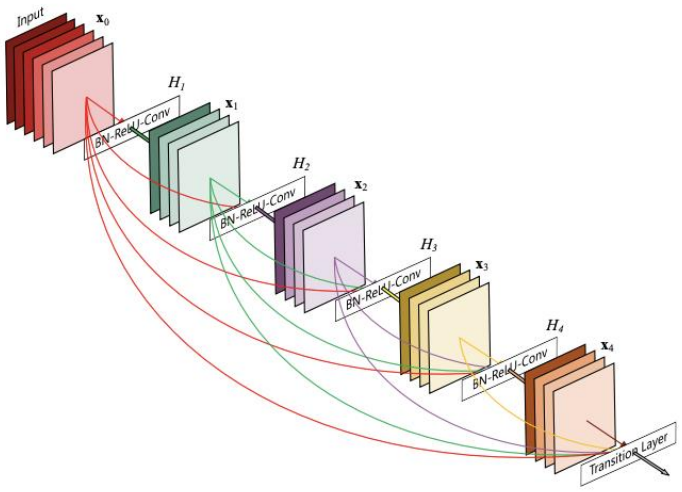


Figure 4.2.1.1 Connection of a typical DenseNet

3.4.16 LeNet

LeNet, also called LeNet-5, is a groundbreaking Convolutional Neural Network (CNN) architecture that helped to establish the foundation for contemporary deep learning techniques in image recognition. Developed by Yann LeCun et al. in the 1990s, it laid the groundwork for later CNN advancements with its ground-breaking success in handwritten digit recognition.

The salient characteristics of LeNet are as follows:

- Structure that is straightforward but efficient: It has seven layers: fully connected layers for classification, pooling layers for downsampling, and convolutional layers for feature extraction.

- early adopter of important ideas from CNN: It established the groundwork for the development of deep learning by being the first to employ pooling layers, backpropagation algorithms, and convolutional layers in CNN training.
- Specifically created for image recognition tasks, it showcases the effectiveness of CNNs in image-based applications by recognizing handwritten numbers.
- Important in real-world applications: LeNet's legacy goes beyond academia because it was notably applied to identify numbers in ATMs, demonstrating its practical utility.

While LeNet's architecture is relatively simple compared to modern CNNs, its historical significance and foundational contributions to deep learning remain undeniable.

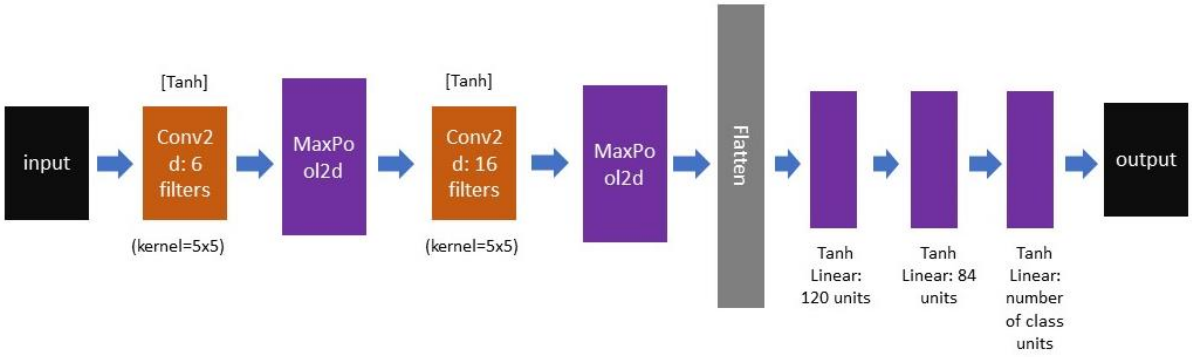


Figure 4.2.1.1Model Architecture of Lenet-5

CHAPTER 4
EXPERIMENTAL DISCUSSION, RESULT, AND ANALYSIS

4.1 Discussion

Using a differentiated epoch allocation strategy, I was able to accommodate different training requirements and guarantee a balanced comparison. Although the majority of algorithms were trained over 10 epochs, I purposefully used DenseNet's training to last only 5 epochs. Its known propensity for prolonged training and possibility for overfitting with prolonged epochs served as the basis for this decision. I wanted to optimize DenseNet's performance in a reasonable amount of time and make sure that comparisons with other algorithms are more equitable, so I gave it a customized training regimen.

Table 4.1.1 A brief summary of the model's group-wise

Group	Model	Description
Tree-Based Algorithms	Random Forest	To arrive at a final decision, these algorithms build numerous decision trees and aggregate their forecasts. They are very good at managing intricate feature interactions and non-linear relationships.
	Extra Trees	
	Bagging Classifier	
	Decision Tree	
Distance-Based Algorithms	KNN	The proximity of new data points to preexisting data points serves as the foundation for these algorithms' predictions. Although they are simple and intuitive, they can be affected by the distance metric selected and the existence of outliers.
	Nearest Centroid	

Group	Model	Description
Linear Algorithms	Logistic Regression	The target variable and features are assumed to have a linear relationship by these algorithms. They may have trouble with non-linear problems, but they are efficient and comprehensible.
	Linear SVM	
	Stochastic Gradient Des cent	
Kernel-Based Algorithms:	SVM (with non-linear kernels)	These algorithms convert data into a higher-dimensional space into which classes can be effectively divided by a linear decision boundary. They can be computationally costly, but they are effective for non-linear problems.
Probabilistic Algorithms:	Naive Bayes	Based on feature values, these algorithms predict class membership probabilities using the Bayes theorem. They work well for spam filtering and text classification.
	Multinomial Naive Bayes	
Boosting Algorithms	AdaBoost	By building an ensemble of weak learners iteratively, this algorithm concentrates on fixing the mistakes made by its predecessors. It can be prone to overfitting but frequently achieves high accuracy.
Convolutional Neural Network	CustomCNN	It provides adaptability and flexibility for particular tasks, enabling customized model design. The CustomCNN that I've been using extracts features using two convolutional layers with ReLU activation and

Group	Model	Description
		max pooling. Two fully connected layers with ReLU are then used for classification. Preprocessing involves resizing, grayscale conversion, and pixel value normalization for images.
	DenseNet	It has strong interlayer connections, which facilitate effective information propagation and feature reuse. I have used a custom DenseNet model for image classification over 5 epochs. It uses Adam Optimizer and CrossEntropyLoss, iterating through training data batches to update model parameters.
	LeNet	It is a fundamentally sound, straightforward architecture that has historical relevance for handwritten digit identification. Three fully connected layers with Tanh activation come after two convolutional layers with Tanh activation and max pooling extract features. Probabilities for the predetermined number of classes are output by the final layer.

4.2 Experimental Result

4.2.1 Random Forest

Final accuracy for Vowel: 83.64 %

Final accuracy for Consonant: 71.60%

4.2.1.1 Training accuracy

Table 4.2.1Training accuracy of Random Forest

Section	Average Training accuracy
Vowel	100.00%
Consonant	100.00%

4.2.1.1 Confusion Matrix

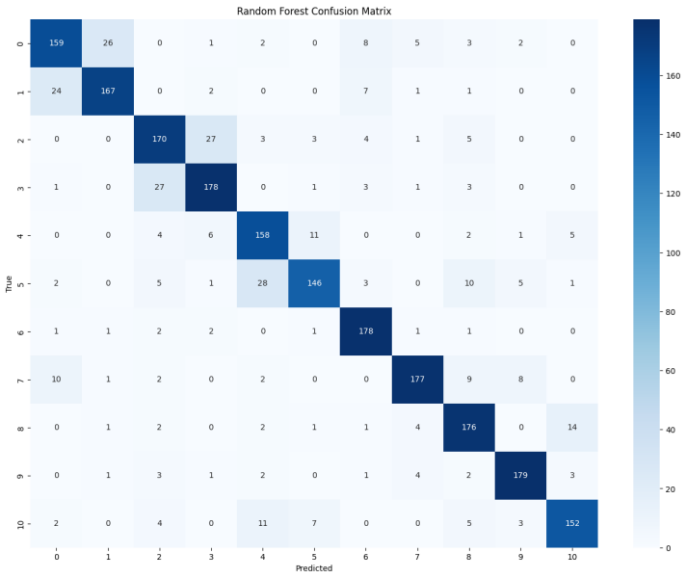


Figure 4.2.1.1 confusion matrix Random Forest (Vowel)

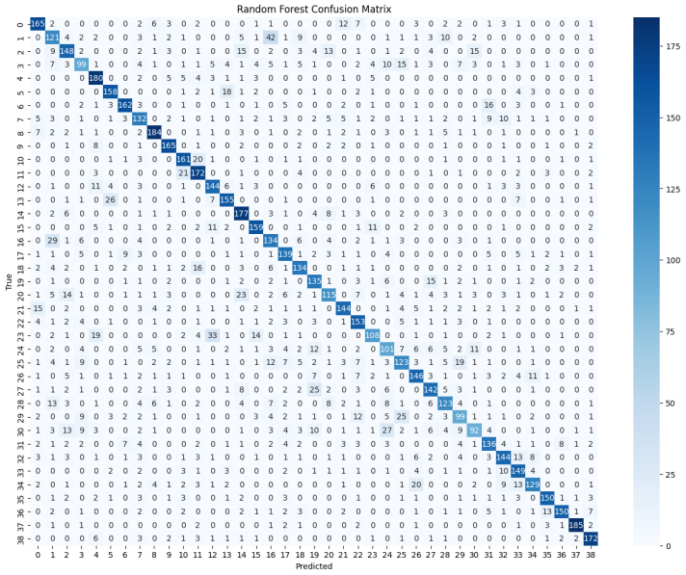


Figure 4.2.1.2 confusion matrix Random Forest (Consonant)

4.2.1.1 Classification Report

Table 4.2.2 classification report Random Forest (Vowel)

	precision	recall	f1-score	support
0	0.80	0.77	0.79	206
1	0.85	0.83	0.84	202
2	0.78	0.80	0.79	213
3	0.82	0.83	0.82	214
4	0.76	0.84	0.80	187
5	0.86	0.73	0.79	201
6	0.87	0.95	0.91	187
7	0.91	0.85	0.88	209
8	0.81	0.88	0.84	201
9	0.90	0.91	0.91	196
10	0.87	0.83	0.85	184
accuracy			0.84	2200
macro avg	0.84	0.84	0.84	2200
weighted avg	0.84	0.84	0.84	2200

Table 4.2.3 classification report Random Forest (Consonant)

	precision	recall	f1-score	support
0	0.77	0.77	0.77	214
1	0.55	0.57	0.56	213
2	0.68	0.65	0.67	226
3	0.61	0.51	0.56	193
4	0.73	0.85	0.79	211
5	0.78	0.81	0.80	194
6	0.84	0.80	0.82	203
7	0.63	0.66	0.65	199
8	0.83	0.82	0.82	225
9	0.82	0.87	0.84	190
...				
macro avg	0.72	0.72	0.71	7800
weighted avg	0.72	0.72	0.71	7800

4.2.2 KNN

Final accuracy for Vowel: 82.00%

Final accuracy for Consonant: 69.41%

4.2.2.1 Training accuracy

Figure 4.2.2.1Model Architecture of Lenet-5

Section	Average Training accuracy
Vowel	89.11
Consonant	80.75%

4.2.2.1 Confusion Matrix

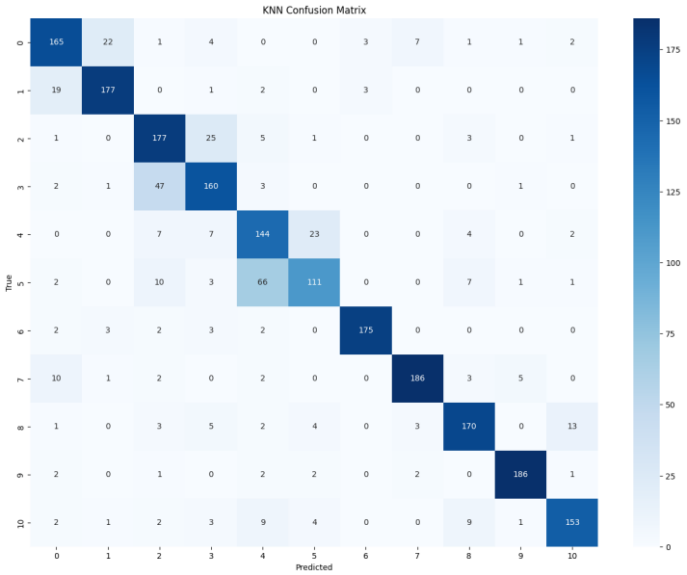


Figure 4.2.2.1 confusion matrix KNN (Vowel)

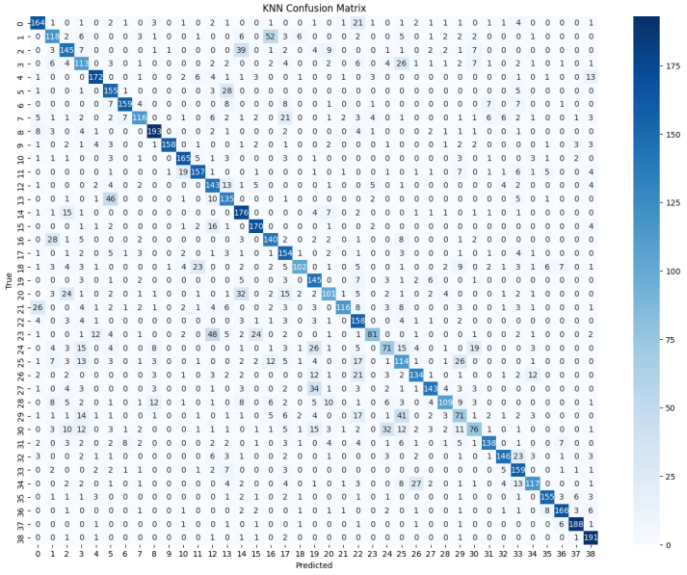


Figure 4.2.2.2 confusion matrix KNN (Consonant)

4.2.2.1 Classification Report

Table 4.2.4 classification report KNN (Vowel)

	precision	recall	f1-score	support
0	0.80	0.80	0.80	206
1	0.86	0.88	0.87	202
2	0.70	0.83	0.76	213
3	0.76	0.75	0.75	214
4	0.61	0.77	0.68	187
5	0.77	0.55	0.64	201
6	0.97	0.94	0.95	187
7	0.94	0.89	0.91	209
8	0.86	0.85	0.85	201
9	0.95	0.95	0.95	196
10	0.88	0.83	0.86	184
accuracy			0.82	2200
macro avg	0.83	0.82	0.82	2200
weighted avg	0.83	0.82	0.82	2200

Table 4.2.5 classification report KNN (Consonant)

	precision	recall	f1-score	support
0	0.73	0.77	0.75	214
1	0.61	0.55	0.58	213
2	0.60	0.64	0.62	226
3	0.50	0.59	0.54	193
4	0.84	0.82	0.83	211
5	0.60	0.80	0.68	194
6	0.88	0.78	0.83	203
7	0.78	0.58	0.67	199
8	0.84	0.86	0.85	225
9	0.98	0.83	0.90	190
...				
macro avg	0.71	0.69	0.69	7800
weighted avg	0.71	0.69	0.69	7800

4.2.3 SVM

Final accuracy for Vowel: 88.14%

Final accuracy for Consonant: 78.77 %

4.2.3.1 Training accuracy

Table 4.2.6Training accuracy of SVM

Section	Average Training accuracy
Vowel	96.44
Consonant	91.74

4.2.3.1 Confusion Matrix

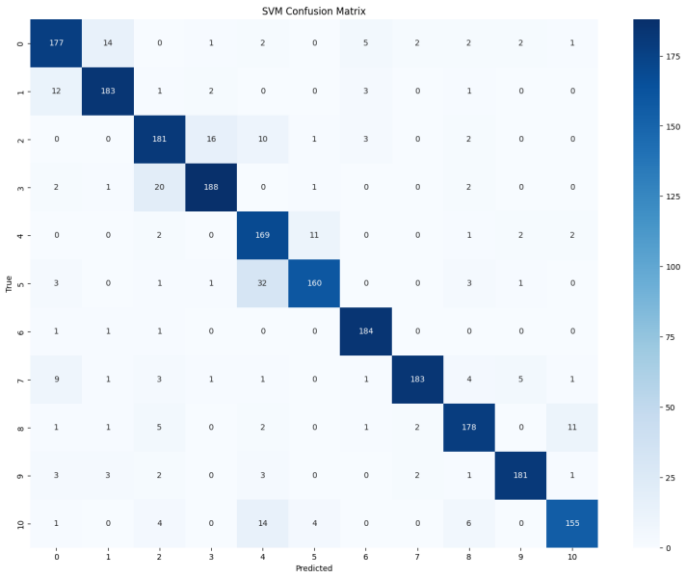


Figure 4.2.3.1 confusion matrix SVM (Vowel)

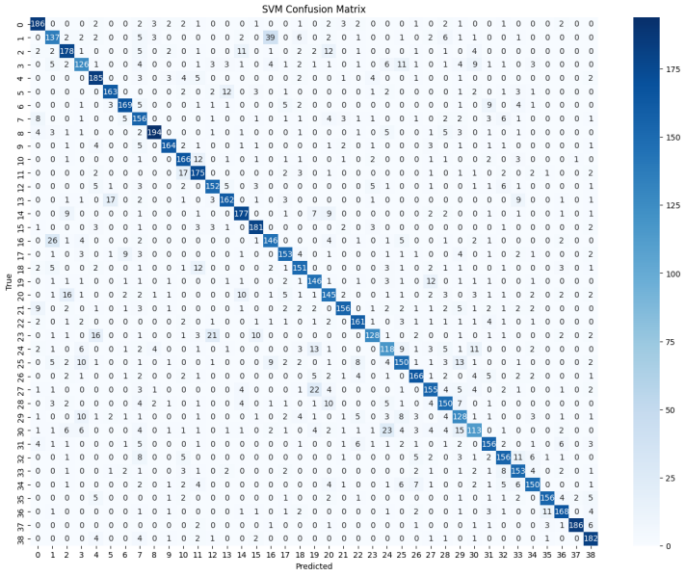


Figure 4.2.3.2 confusion matrix SVM (Consonant)

4.2.3.1 Classification Report

Table 4.2.7 classification report SVM (Vowel)

	precision	recall	f1-score	support
0	0.85	0.86	0.85	206
1	0.90	0.91	0.90	202
2	0.82	0.85	0.84	213
3	0.90	0.88	0.89	214
4	0.73	0.90	0.80	187
5	0.90	0.80	0.85	201
6	0.93	0.98	0.96	187
7	0.97	0.88	0.92	209
8	0.89	0.89	0.89	201
9	0.95	0.92	0.94	196
10	0.91	0.84	0.87	184
accuracy			0.88	2200
macro avg	0.89	0.88	0.88	2200
weighted avg	0.89	0.88	0.88	2200

Table 4.2.8 classification report SVM (Consonant)

	precision	recall	f1-score	support
0	0.83	0.87	0.85	214
1	0.70	0.64	0.67	213
2	0.76	0.79	0.78	226
3	0.71	0.65	0.68	193
4	0.79	0.88	0.83	211
5	0.87	0.84	0.85	194
6	0.88	0.83	0.85	203
7	0.63	0.78	0.70	199
8	0.93	0.86	0.90	225
9	0.91	0.86	0.88	190
...				
macro avg	0.79	0.79	0.79	7800
weighted avg	0.79	0.79	0.79	7800

4.2.4 Logistic Regression

Final accuracy for Vowel: 75.86%

Final accuracy for Consonant: 59.42%

4.2.4.1 Training accuracy

Table 4.2.9Training accuracy of Logistic Regression

Section	Average Training accuracy
Vowel	85.92
Consonant	67.19

4.2.4.1 Confusion Matrix

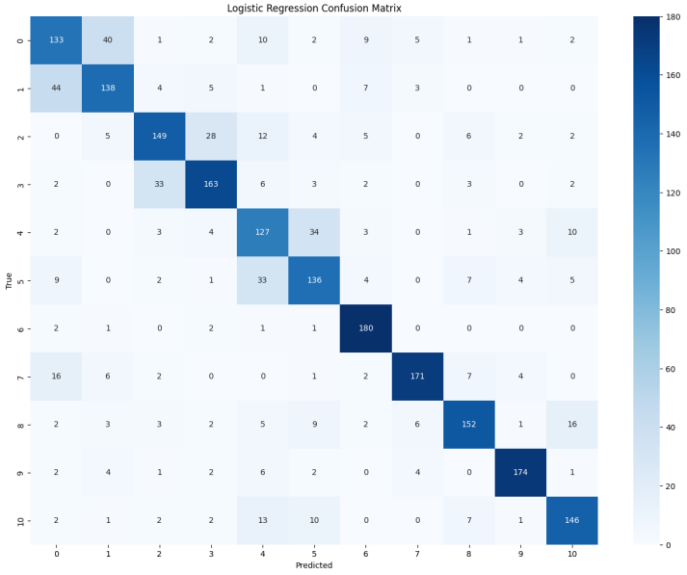


Figure 4.2.4.1 confusion matrix Logistic Regression (Vowel)

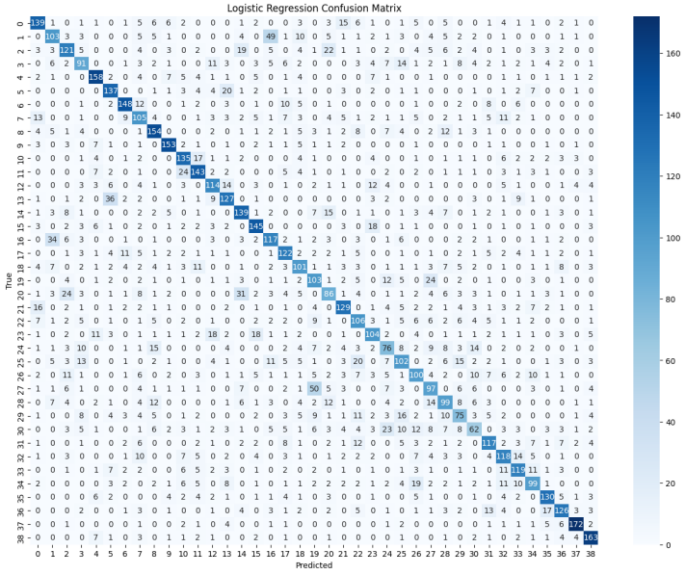


Figure 4.2.4.2 confusion matrix Logistic Regression (Consonant)

4.2.4.1 Classification Report

Table 4.2.10 classification report Logistic Regression (Vowel)

	precision	recall	f1-score	support
0	0.62	0.65	0.63	206
1	0.70	0.68	0.69	202
2	0.74	0.70	0.72	213
3	0.77	0.76	0.77	214
4	0.59	0.68	0.63	187
5	0.67	0.68	0.67	201
6	0.84	0.96	0.90	187
7	0.90	0.82	0.86	209
8	0.83	0.76	0.79	201
9	0.92	0.89	0.90	196
10	0.79	0.79	0.79	184
accuracy			0.76	2200
macro avg	0.76	0.76	0.76	2200
weighted avg	0.76	0.76	0.76	2200

Table 4.2.11 classification report Logistic Regression (Consonant)

	precision	recall	f1-score	support
0	0.67	0.65	0.66	214
1	0.57	0.48	0.52	213
2	0.56	0.54	0.55	226
3	0.52	0.47	0.50	193
4	0.73	0.75	0.74	211
5	0.63	0.71	0.66	194
6	0.76	0.73	0.74	203
7	0.48	0.53	0.50	199
8	0.66	0.68	0.67	225
9	0.77	0.81	0.79	190
...				
macro avg	0.59	0.59	0.59	7800
weighted avg	0.59	0.59	0.59	7800

4.2.5 AdaBoost

Final accuracy for Vowel: 54.23%

Final accuracy for Consonant: 28.27%

4.2.5.1 Training accuracy

Table 4.2.12Training accuracy of AdaBoost

Section	Average Training accuracy
Vowel	55.26
Consonant	28.48

4.2.5.1 Confusion Matrix

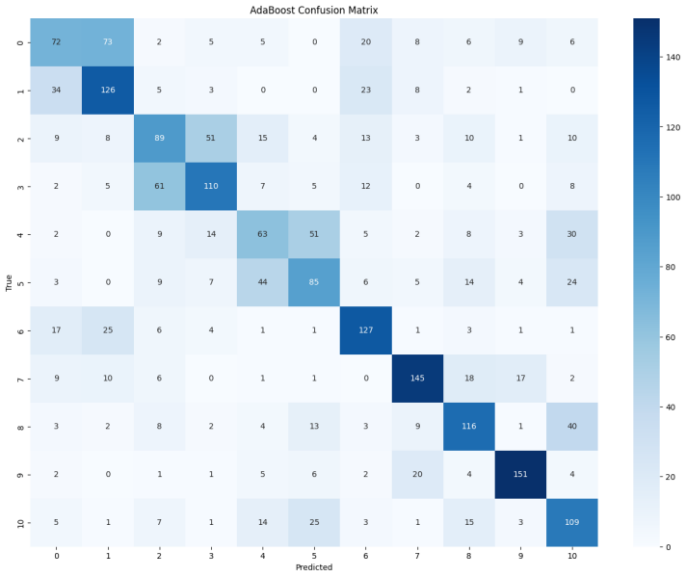


Figure 4.2.5.1 confusion matrix AdaBoost (Vowel)

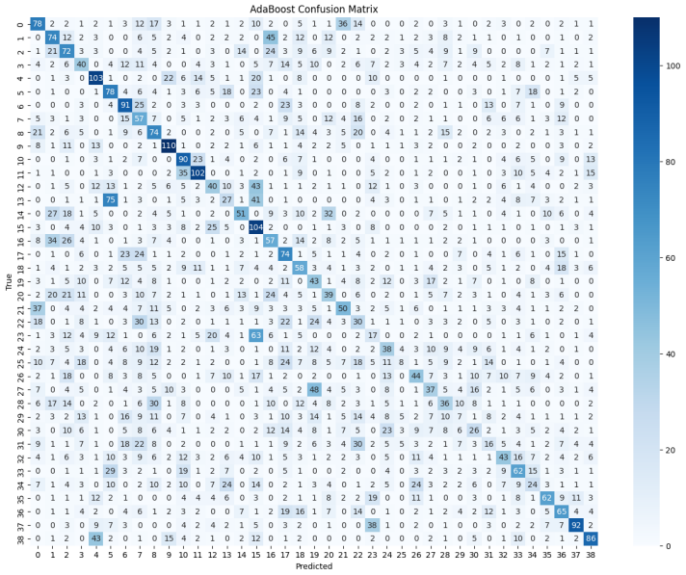


Figure 4.2.5.2 confusion matrix AdaBoost (Consonant)

4.2.5.1 Classification Report

Table 4.2.13classification report AdaBoost (Vowel)

	precision	recall	f1-score	support
0	0.46	0.35	0.40	206
1	0.50	0.62	0.56	202
2	0.44	0.42	0.43	213
3	0.56	0.51	0.53	214
4	0.40	0.34	0.36	187
5	0.45	0.42	0.43	201
6	0.59	0.68	0.63	187
7	0.72	0.69	0.71	209
8	0.58	0.58	0.58	201
9	0.79	0.77	0.78	196
10	0.47	0.59	0.52	184
accuracy			0.54	2200
macro avg	0.54	0.54	0.54	2200
weighted avg	0.54	0.54	0.54	2200

Table 4.2.14 classification report AdaBoost (Consonant)

	precision	recall	f1-score	support

```
              0       0.32      0.36      0.34       214
              1       0.31      0.35      0.33       213
              2       0.25      0.32      0.28       226
              3       0.23      0.21      0.22       193
              4       0.42      0.49      0.45       211
              5       0.27      0.40      0.32       194
              6       0.34      0.45      0.39       203
              7       0.18      0.29      0.22       199
              8       0.24      0.33      0.28       225
              9       0.51      0.58      0.54       190

...
   macro avg       0.28      0.28      0.27      7800
weighted avg       0.28      0.28      0.27      7800
```

4.2.6 Naive Bayes

Final accuracy for Vowel: 66.73%

Final accuracy for Consonant: 42.41%

4.2.6.1 Training accuracy

Table 4.2.15Training accuracy of Naive Bayes

Section	Average Training accuracy
Vowel	67.98
Consonant	43.53

4.2.6.1 Confusion Matrix

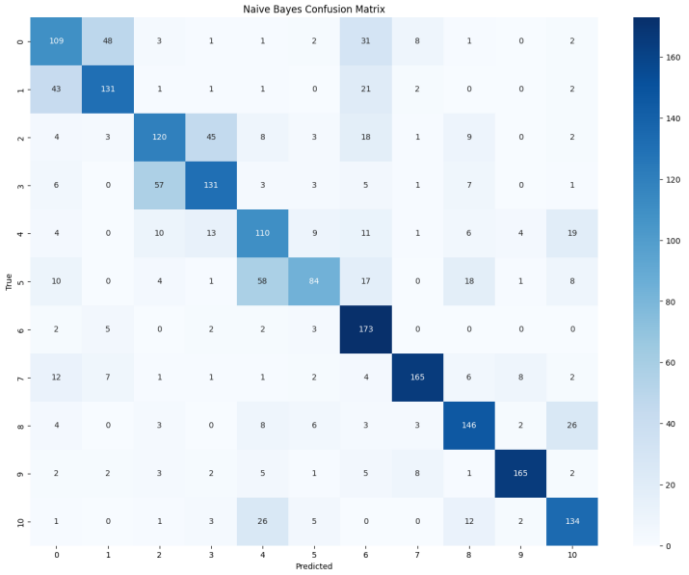


Figure 4.2.6.1 confusion matrix Naive Bayes (Vowel)

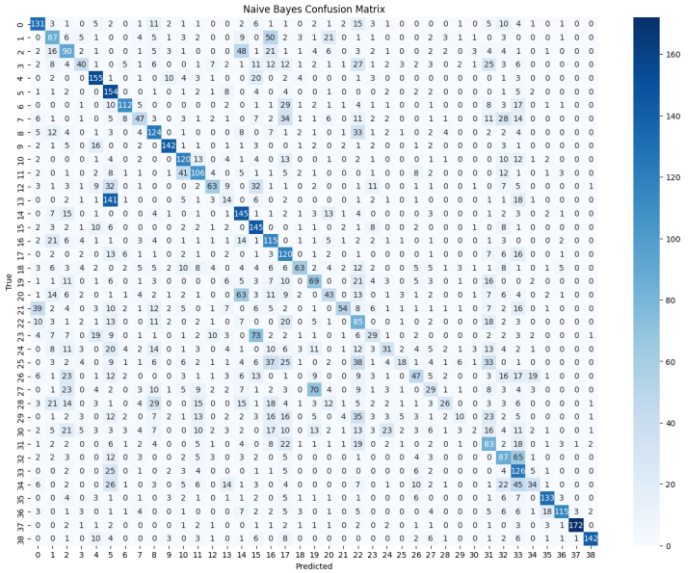


Figure 4.2.6.2 confusion matrix Naive Bayes (Consonant)

4.2.6.1 Classification Report

Table 4.2.16 classification report Naive Bayes (Vowel)

	precision	recall	f1-score	support
0	0.55	0.53	0.54	206
1	0.67	0.65	0.66	202
2	0.59	0.56	0.58	213
3	0.66	0.61	0.63	214
4	0.49	0.59	0.54	187
5	0.71	0.42	0.53	201
6	0.60	0.93	0.73	187
7	0.87	0.79	0.83	209
8	0.71	0.73	0.72	201
9	0.91	0.84	0.87	196
10	0.68	0.73	0.70	184
accuracy			0.67	2200
macro avg	0.68	0.67	0.67	2200
weighted avg	0.68	0.67	0.66	2200

Table 4.2.17 classification report Naive Bayes (Consonant)

	precision	recall	f1-score	support
0	0.54	0.61	0.58	214
1	0.36	0.41	0.38	213
2	0.31	0.40	0.35	226
3	0.47	0.21	0.29	193
4	0.61	0.73	0.66	211
5	0.28	0.79	0.41	194
6	0.73	0.55	0.63	203
7	0.42	0.24	0.30	199
8	0.45	0.55	0.49	225
9	0.80	0.75	0.77	190
...				
macro avg	0.46	0.42	0.40	7800
weighted avg	0.46	0.42	0.41	7800

4.2.7 Extra Trees

Final accuracy for Vowel: 85.05%

Final accuracy for Consonant: 72.15%

4.2.7.1 Training accuracy

Table 4.2.18Training accuracy of Extra Trees

Section	Average Training accuracy
Vowel	100.00
Consonant	100.00

4.2.7.1 Confusion Matrix

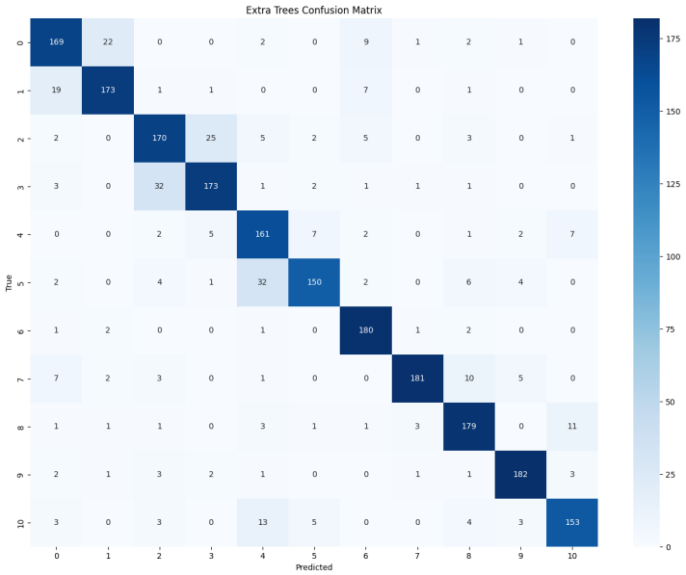


Figure 4.2.7.1 confusion matrix Extra Trees (Vowel)

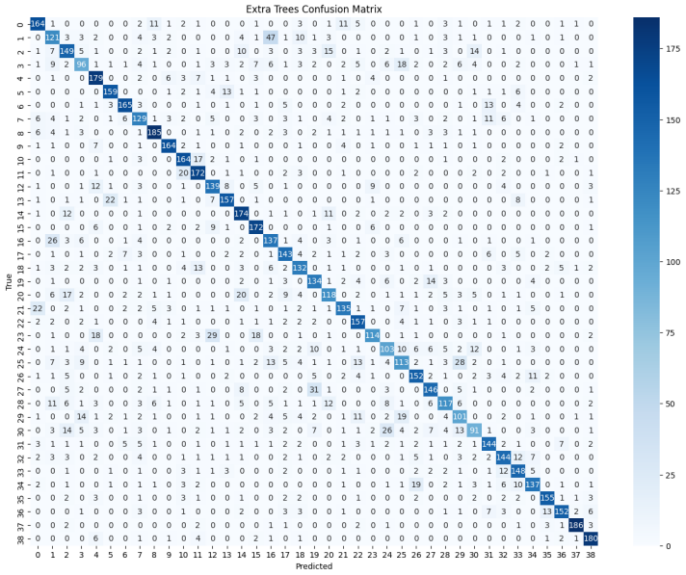


Figure 4.2.7.2 confusion matrix Extra Trees (Consonant)

4.2.7.1 Classification Report

Table 4.2.19 classification report Extra Trees (Vowel)

	precision	recall	f1-score	support
0	0.81	0.82	0.81	206
1	0.86	0.86	0.86	202
2	0.78	0.80	0.79	213
3	0.84	0.81	0.82	214
4	0.73	0.86	0.79	187
5	0.90	0.75	0.82	201
6	0.87	0.96	0.91	187
7	0.96	0.87	0.91	209
8	0.85	0.89	0.87	201
9	0.92	0.93	0.93	196
10	0.87	0.83	0.85	184
accuracy			0.85	2200
macro avg	0.85	0.85	0.85	2200
weighted avg	0.85	0.85	0.85	2200

Table 4.2.20 classification report Extra Trees (Consonant)

	precision	recall	f1-score	support
0	0.76	0.77	0.76	214

```
                1       0.56      0.57      0.56       213
                2       0.63      0.66      0.64       226
                3       0.59      0.50      0.54       193
                4       0.72      0.85      0.78       211
                5       0.80      0.82      0.81       194
                6       0.85      0.81      0.83       203
                7       0.63      0.65      0.64       199
                8       0.81      0.82      0.82       225
                9       0.86      0.86      0.86       190

...
   macro avg       0.72      0.72      0.72      7800
weighted avg       0.72      0.72      0.72      7800
```

4.2.8 Bagging Classifier

Final accuracy for Vowel: 75.18%

Final accuracy for Consonant: 56.29%

4.2.8.1 Training accuracy

Table 4.2.21Training accuracy of Bagging Classifier

Section	Average Training accuracy
Vowel	99.49
Consonant	99.61

4.2.8.1 Confusion Matrix

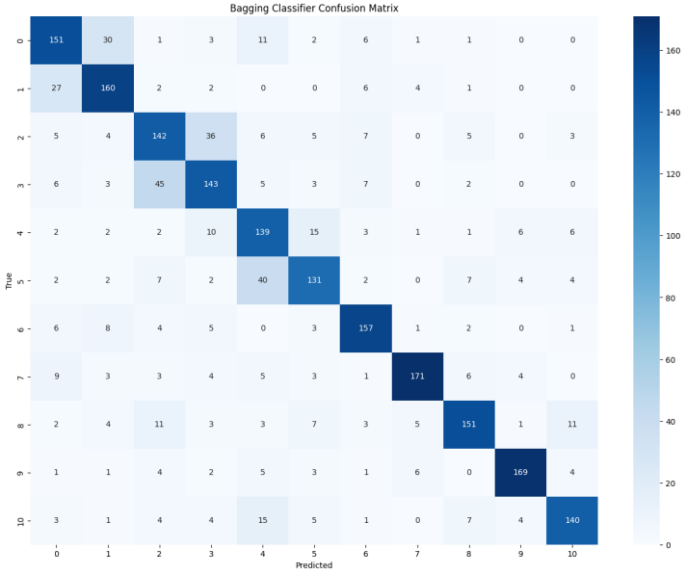


Figure 4.2.8.1 confusion matrix Bagging Classifier (Vowel)

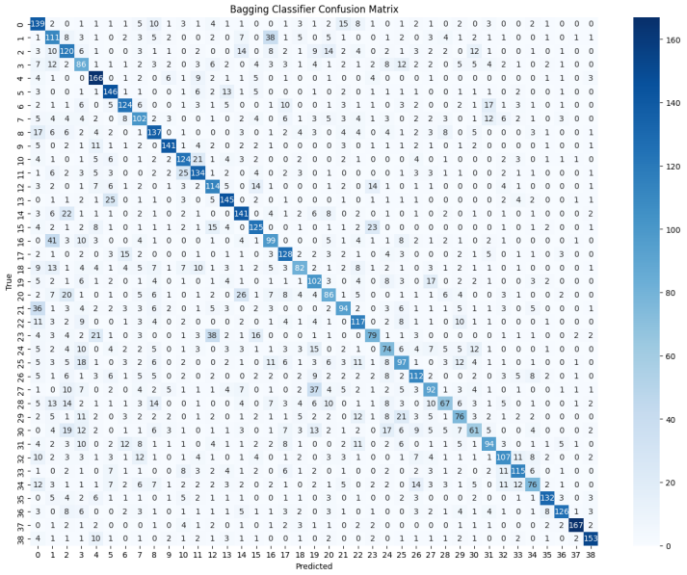


Figure 4.2.8.2 confusion matrix Bagging Classifier (Consonant)

4.2.8.1 Classification Report

Table 4.2.22 classification report Bagging Classifier (Vowel)

	precision	recall	f1-score	support
0	0.71	0.73	0.72	206
1	0.73	0.79	0.76	202
2	0.63	0.67	0.65	213
3	0.67	0.67	0.67	214
4	0.61	0.74	0.67	187
5	0.74	0.65	0.69	201
6	0.81	0.84	0.82	187
7	0.90	0.82	0.86	209
8	0.83	0.75	0.79	201
9	0.90	0.86	0.88	196
10	0.83	0.76	0.79	184
accuracy			0.75	2200
macro avg	0.76	0.75	0.75	2200
weighted avg	0.76	0.75	0.75	2200

Table 4.2.23 classification report Bagging Classifier (Consonant)

	precision	recall	f1-score	support
0	0.43	0.65	0.51	214
1	0.40	0.52	0.45	213
2	0.42	0.53	0.47	226
3	0.35	0.45	0.39	193
4	0.60	0.79	0.68	211
5	0.60	0.75	0.67	194
6	0.64	0.61	0.63	203
7	0.48	0.51	0.50	199
8	0.56	0.61	0.58	225
9	0.78	0.74	0.76	190
...				
macro avg	0.57	0.56	0.56	7800
weighted avg	0.57	0.56	0.56	7800

4.2.9 Decision Tree

Final accuracy for Vowel: 60.59%

Final accuracy for Consonant: 41.99 %

4.2.9.1 Training accuracy

Table 4.2.24Training accuracy of Decision Tree

Section	Average Training accuracy
Vowel	100.00
Consonant	100.00

4.2.9.1 Confusion Matrix

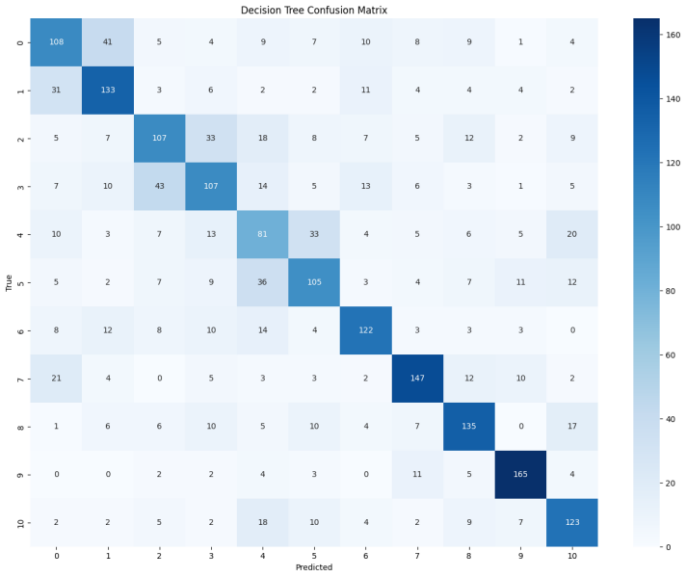


Figure 4.2.9.1 confusion matrix Decision Tree (Vowel)

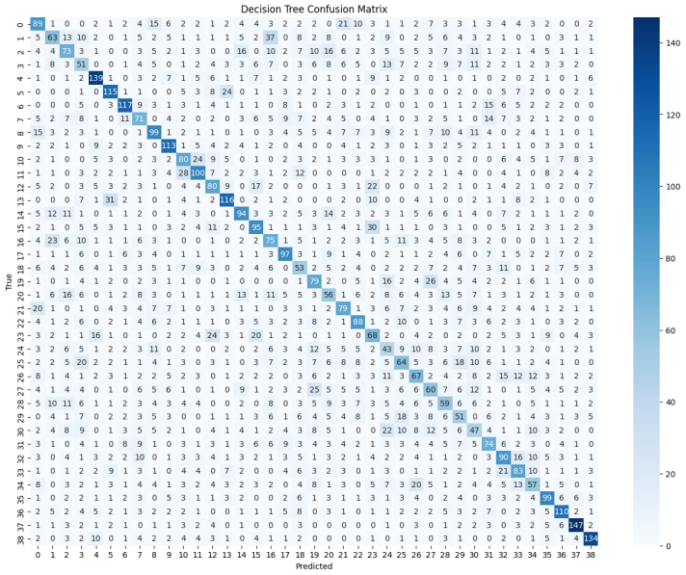


Figure 4.2.9.2 confusion matrix Decision Tree (Consonant)

4.2.9.1 Classification Report

Table 4.2.25 classification report Decision Tree (Vowel)

	precision	recall	f1-score	support
0	0.55	0.52	0.53	206
1	0.60	0.66	0.63	202
2	0.55	0.50	0.53	213
3	0.53	0.50	0.52	214
4	0.40	0.43	0.41	187
5	0.55	0.52	0.54	201
6	0.68	0.65	0.66	187
7	0.73	0.70	0.72	209
8	0.66	0.67	0.67	201
9	0.79	0.84	0.81	196
10	0.62	0.67	0.64	184
accuracy			0.61	2200
macro avg	0.61	0.61	0.61	2200
weighted avg	0.61	0.61	0.61	2200

Table 4.2.26 classification report Decision Tree (Consonant)

	precision	recall	f1-score	support
0	0.40	0.42	0.41	214
1	0.37	0.30	0.33	213
2	0.37	0.32	0.35	226
3	0.24	0.26	0.25	193
4	0.62	0.66	0.64	211
5	0.55	0.59	0.57	194
6	0.63	0.58	0.60	203
7	0.35	0.36	0.35	199
8	0.46	0.44	0.45	225
9	0.58	0.59	0.59	190
...				
macro avg	0.42	0.42	0.42	7800
weighted avg	0.42	0.42	0.42	7800

4.2.10 Stochastic Gradient Descent

Final accuracy for Vowel: 68.64 %

Final accuracy for Consonant: 42.85%

4.2.10.1 Training accuracy

Table 4.2.27Training accuracy of Stochastic Gradient Descent

Section	Average Training accuracy
Vowel	79.75
Consonant	49.77

4.2.10.1 Confusion Matrix

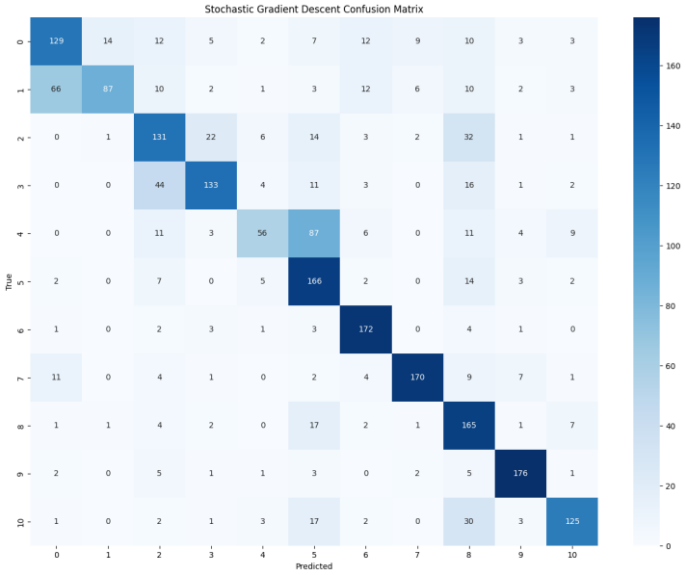


Figure 4.2.10.1 confusion matrix Stochastic Gradient Descent (Vowel)

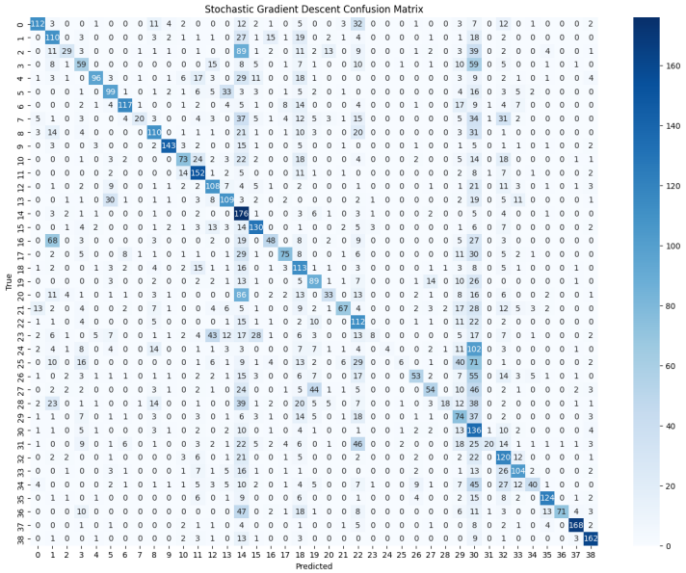


Figure 4.2.10.2 confusion matrix Stochastic Gradient Descent (Consonant)

4.2.10.1 Classification Report

Table 4.2.28 classification report Stochastic Gradient Descent (Vowel)

	precision	recall	f1-score	support
0	0.61	0.63	0.62	206
1	0.84	0.43	0.57	202
2	0.56	0.62	0.59	213
3	0.77	0.62	0.69	214
4	0.71	0.30	0.42	187
5	0.50	0.83	0.63	201
6	0.79	0.92	0.85	187
7	0.89	0.81	0.85	209
8	0.54	0.82	0.65	201
9	0.87	0.90	0.88	196
10	0.81	0.68	0.74	184
accuracy			0.69	2200
macro avg	0.72	0.69	0.68	2200
weighted avg	0.72	0.69	0.68	2200

Table 4.2.29 classification report Stochastic Gradient Descent (Consonant)

	precision	recall	f1-score	support
0	0.75	0.52	0.62	214
1	0.37	0.52	0.43	213
2	0.59	0.13	0.21	226
3	0.36	0.31	0.33	193
4	0.83	0.45	0.59	211
5	0.56	0.51	0.53	194
6	0.79	0.58	0.67	203
7	0.80	0.10	0.18	199
8	0.55	0.49	0.52	225
9	0.87	0.75	0.81	190
...				
macro avg	0.60	0.43	0.43	7800
weighted avg	0.60	0.43	0.43	7800

4.2.11 Nearest Centroid

Final accuracy for Vowel: 71.00%

Final accuracy for Consonant: 50.00 %

4.2.11.1 Training accuracy

Table 4.2.30Training accuracy of Nearest Centroid

Section	Average Training accuracy
Vowel	72.02
Consonant	50.53

4.2.11.1 Confusion Matrix

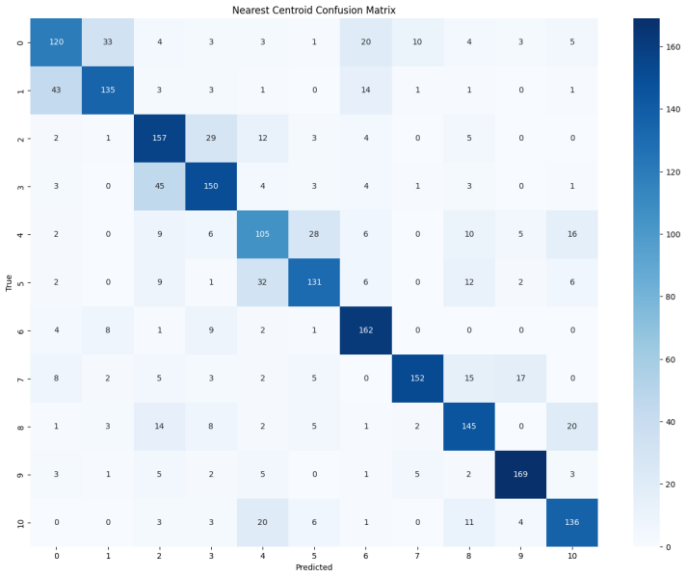


Figure 4.2.11.1 confusion matrix Nearest Centroid (Vowel)

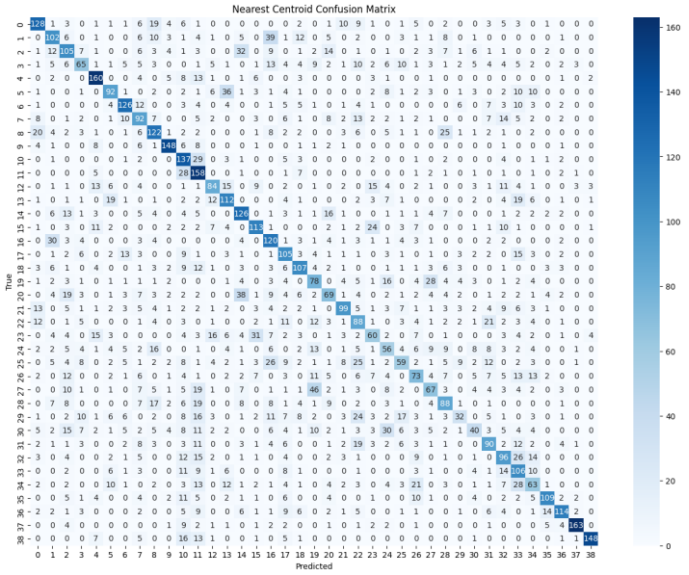


Figure 4.2.11.2 confusion matrix Nearest Centroid (Consonant)

4.2.11.1 Classification Report

Table 4.2.31 classification report Nearest Centroid (Vowel)

	precision	recall	f1-score	support
0	0.64	0.58	0.61	206
1	0.74	0.67	0.70	202
2	0.62	0.74	0.67	213
3	0.69	0.70	0.70	214
4	0.56	0.56	0.56	187
5	0.72	0.65	0.68	201
6	0.74	0.87	0.80	187
7	0.89	0.73	0.80	209
8	0.70	0.72	0.71	201
9	0.84	0.86	0.85	196
10	0.72	0.74	0.73	184
accuracy			0.71	2200
macro avg	0.71	0.71	0.71	2200
weighted avg	0.71	0.71	0.71	2200

Table 4.2.32 classification report Nearest Centroid (Consonant)

	precision	recall	f1-score	support
0	0.60	0.60	0.60	214
1	0.51	0.48	0.49	213
2	0.41	0.46	0.44	226
3	0.47	0.34	0.39	193
4	0.67	0.76	0.71	211
5	0.54	0.47	0.51	194
6	0.65	0.62	0.64	203
7	0.41	0.46	0.43	199
8	0.50	0.54	0.52	225
9	0.79	0.78	0.79	190
...				
macro avg	0.51	0.50	0.50	7800
weighted avg	0.51	0.50	0.50	7800

4.2.12 Linear SVM

Final accuracy for Vowel: 65.41%

Final accuracy for Consonant: 36.60%

4.2.12.1 Training accuracy

Table 4.2.33Training accuracy of Linear SVM

Section	Average Training accuracy
Vowel	87.75
Consonant	45.60

4.2.12.1 Confusion Matrix

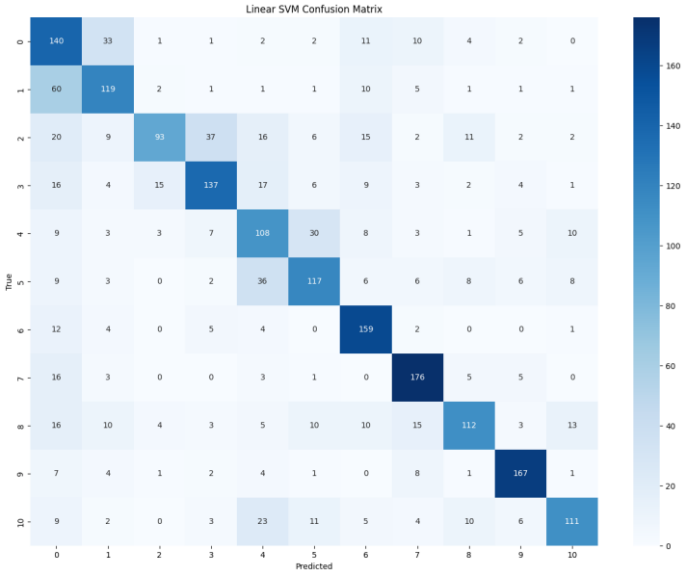


Figure 4.2.12.1 confusion matrix Linear SVM (Vowel)

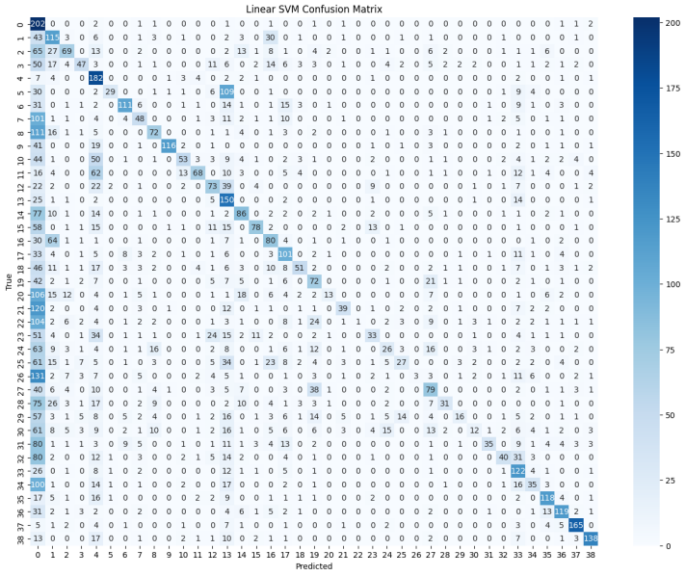


Figure 4.2.12.2 confusion matrix Linear SVM (Consonant)

4.2.12.1 Classification Report

Table 4.2.34 classification report Linear SVM (Vowel)

	precision	recall	f1-score	support
0	0.45	0.68	0.54	206
1	0.61	0.59	0.60	202
2	0.78	0.44	0.56	213
3	0.69	0.64	0.67	214
4	0.49	0.58	0.53	187
5	0.63	0.58	0.61	201
6	0.68	0.85	0.76	187
7	0.75	0.84	0.79	209
8	0.72	0.56	0.63	201
9	0.83	0.85	0.84	196
10	0.75	0.60	0.67	184
accuracy			0.65	2200
macro avg	0.67	0.66	0.65	2200
weighted avg	0.67	0.65	0.65	2200

Figure 4.2.12.1 classification report Linear SVM (Consonant)

	precision	recall	f1-score	support
0	0.09	0.94	0.16	214
1	0.30	0.54	0.39	213
2	0.52	0.31	0.38	226
3	0.57	0.24	0.34	193
4	0.30	0.86	0.44	211
5	0.83	0.15	0.25	194
6	0.74	0.55	0.63	203
7	0.45	0.24	0.31	199
8	0.54	0.32	0.40	225
9	0.96	0.61	0.75	190
...				
macro avg	0.58	0.36	0.37	7800
weighted avg	0.57	0.37	0.37	7800

4.2.13 Multinomial Naive Bayes

Final accuracy for Vowel: 71.00%

Final accuracy for Consonant: 48.12 %

4.2.13.1 Training accuracy

Table 4.2.35Training accuracy of Multinomial Naive Bayes

Section	Average Training accuracy
Vowel	71.33
Consonant	49.14

4.2.13.1 Confusion Matrix

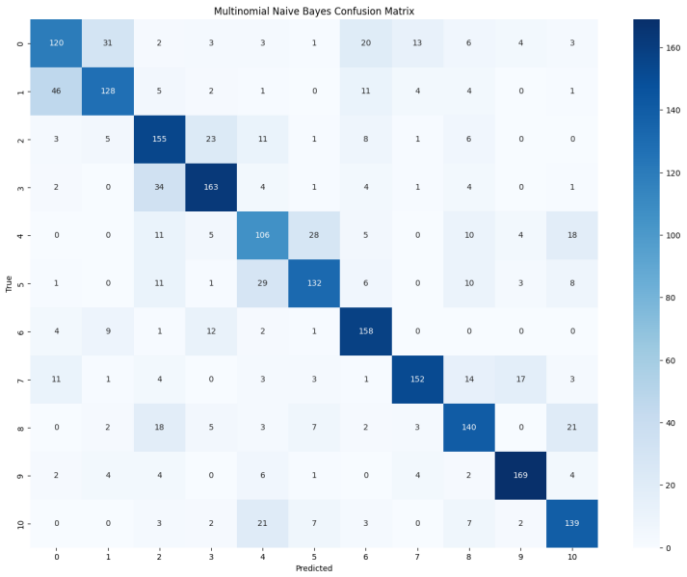


Figure 4.2.13.1 confusion matrix Multinomial Naive Bayes (Vowel)

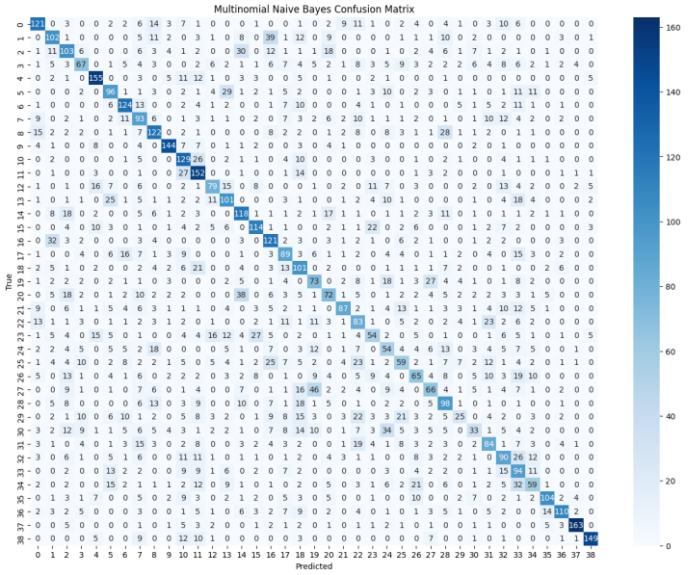


Figure 4.2.13.2 confusion matrix Multinomial Naive Bayes (Consonant)

4.2.13.1 Classification Report

Table 4.2.36 classification report Multinomial Naive Bayes (Vowel)

	precision	recall	f1-score	support
0	0.63	0.58	0.61	206
1	0.71	0.63	0.67	202
2	0.62	0.73	0.67	213
3	0.75	0.76	0.76	214
4	0.56	0.57	0.56	187
5	0.73	0.66	0.69	201
6	0.72	0.84	0.78	187
7	0.85	0.73	0.79	209
8	0.69	0.70	0.69	201
9	0.85	0.86	0.86	196
10	0.70	0.76	0.73	184
accuracy			0.71	2200
macro avg	0.71	0.71	0.71	2200
weighted avg	0.71	0.71	0.71	2200

Table 4.2.37classification report Multinomial Naive Bayes (Consonant)

	precision	recall	f1-score	support
0	0.60	0.57	0.58	214
1	0.50	0.48	0.49	213
2	0.42	0.46	0.44	226
3	0.48	0.35	0.40	193
4	0.69	0.73	0.71	211
5	0.44	0.49	0.47	194
6	0.60	0.61	0.60	203
7	0.35	0.47	0.40	199
8	0.52	0.54	0.53	225
9	0.78	0.76	0.77	190
...				
macro avg	0.49	0.48	0.48	7800
weighted avg	0.49	0.48	0.48	7800

4.2.14 CustomCNN

Final accuracy for Vowel: 92.96%

Final accuracy for Consonant: 87.72%

4.2.14.1 Training accuracy

Table 4.2.38Training accuracy of CustomCNN (Vowel)

Epoch No	Training accuracy	Training loss
[1/10]	0.7128	76.59%
[2/10]	0.2616	91.64%
[3/10]	0.1785	94.29%
[4/10]	0.1262	95.95%
[5/10]	0.0931	97.13%
[6/10]	0.0733	97.67%
[7/10]	0.0515	98.45%
[8/10]	0.0388	98.90%
[9/10]	0.0308	99.08%
[10/10]	0.0292	99.07%

Table 4.2.39 Training accuracy of CustomCNN (Consonant)

Epoch No	Training accuracy	Training loss
[1/10]	0.9576	73.45%
[2/10]	0.4347	87.54%
[3/10]	0.3134	90.79%
[4/10]	0.2331	93.09%
[5/10]	0.1818	94.40%
[6/10]	0.1379	95.69%
[7/10]	0.1089	96.47%
[8/10]	0.0849	97.30%
[9/10]	0.0734	97.61%
[10/10]	0.0595	98.07%

4.2.14.1 Confusion Matrix

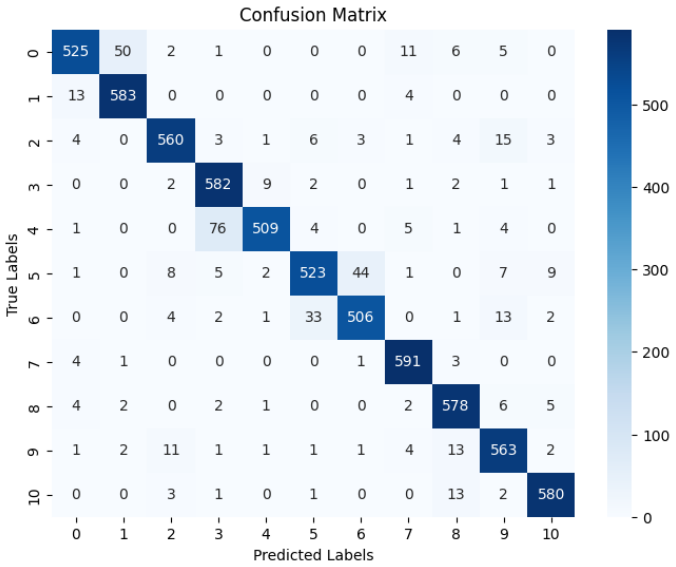


Figure 4.2.14.1 confusion matrix CustomCNN (Vowel)

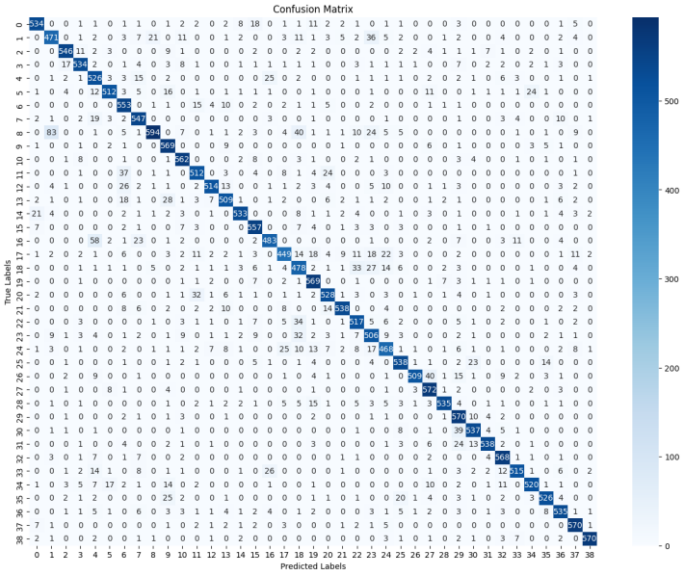


Figure 4.2.14.2 confusion matrix CustomCNN (Consonant)

4.2.14.1 Classification Report

Table 4.2.40 classification report CustomCNN (Vowel)

	precision	recall	f1-score	support
0	0.95	0.88	0.91	600
1	0.91	0.97	0.94	600
2	0.95	0.93	0.94	600
3	0.86	0.97	0.91	600
4	0.97	0.85	0.91	600
5	0.92	0.87	0.89	600
6	0.91	0.90	0.91	562
7	0.95	0.98	0.97	600
8	0.93	0.96	0.95	600
9	0.91	0.94	0.93	600
10	0.96	0.97	0.97	600
accuracy			0.93	6562
macro avg	0.93	0.93	0.93	6562
weighted avg	0.93	0.93	0.93	6562

Table 4.2.41 classification report CustomCNN (Consonant)

	precision	recall	f1-score	support
0	0.92	0.89	0.90	600
1	0.80	0.79	0.79	600
2	0.93	0.91	0.92	600
3	0.91	0.89	0.90	600
4	0.78	0.88	0.82	600
5	0.93	0.85	0.89	600
6	0.80	0.92	0.85	600
7	0.85	0.91	0.88	600
8	0.95	0.74	0.83	800
9	0.82	0.95	0.88	600
10	0.88	0.94	0.91	600
...				
accuracy			0.88	23612
macro avg	0.88	0.88	0.88	23612
weighted avg	0.88	0.88	0.88	23612

4.2.15 DenseNet

Final accuracy for Vowel: 97.93%

Final accuracy for Consonant: 95.99%

4.2.15.1 Training accuracy

Table 4.2.42Training accuracy of DenseNet (Vowel)

Epoch No	Training accuracy	Training loss
[1/5]	0.3268	89.43%
[2/5]	0.1077	96.79%
[3/5]	0.0845	97.58%
[4/5]	0.0582	98.41%
[5/5]	0.0636	98.24%

Figure 4.2.15.1 Training accuracy of DenseNet (Consonant)

Epoch No	Training accuracy	Training loss
[1/5]	0.4761	86.80%
[2/5]	0.1769	94.99%
[3/5]	0.1440	95.97%
[4/5]	0.1200	96.64%
[5/5]	0.1051	97.02%

4.2.15.1 Confusion Matrix

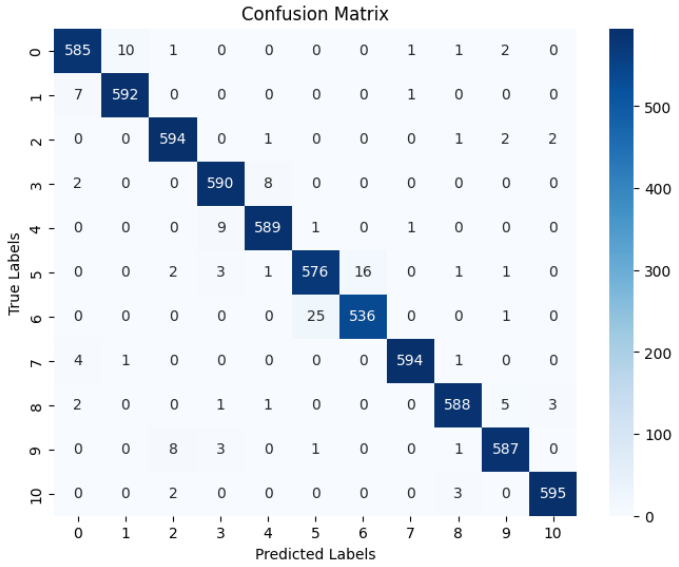


Figure 4.2.15.1 confusion matrix DenseNet (Vowel)

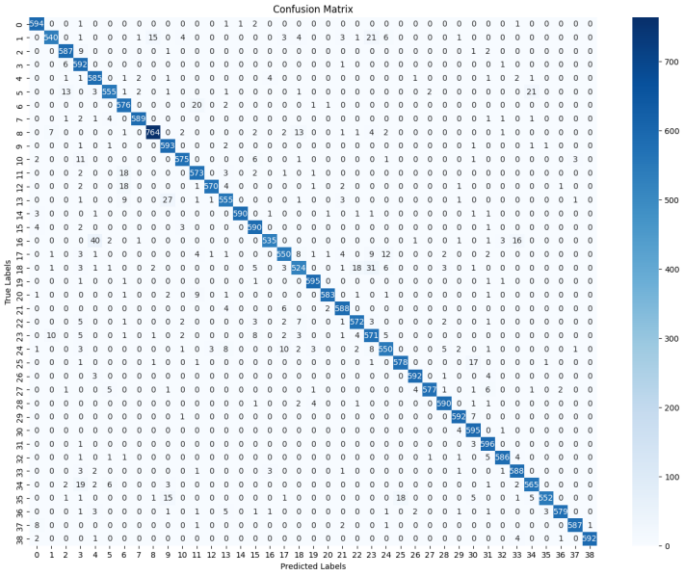


Figure 4.2.15.2 confusion matrix DenseNet (Consonant)

4.2.15.1 Classification Report

Table 4.2.43 classification report DenseNet (Vowel)

	precision	recall	f1-score	support
0	0.97	0.97	0.97	600
1	0.98	0.99	0.98	600
2	0.98	0.99	0.98	600
3	0.97	0.98	0.98	600
4	0.98	0.98	0.98	600
5	0.96	0.96	0.96	600
6	0.97	0.95	0.96	562
7	0.99	0.99	0.99	600
8	0.99	0.98	0.98	600
9	0.98	0.98	0.98	600
10	0.99	0.99	0.99	600
accuracy			0.98	6562
macro avg	0.98	0.98	0.98	6562
weighted avg	0.98	0.98	0.98	6562

Table 4.2.44 classification report DenseNet (Consonant)

	precision	recall	f1-score	support
0	0.97	0.99	0.98	600
1	0.97	0.90	0.93	600
2	0.96	0.98	0.97	600
3	0.88	0.99	0.93	600
4	0.91	0.97	0.94	600
5	0.97	0.93	0.94	600
6	0.92	0.96	0.94	600
7	0.99	0.98	0.99	600
8	0.97	0.95	0.96	800
9	0.92	0.99	0.95	600
10	0.97	0.96	0.97	600
...				
accuracy			0.96	23612
macro avg	0.96	0.96	0.96	23612
weighted avg	0.96	0.96	0.96	23612

4.2.16 LeNet

Final accuracy for Vowel: 91.63%

Final accuracy for Consonant: 85.35%

4.2.16.1 Training accuracy

Table 4.2.45Training accuracy of LeNet (Vowel)

Epoch No	Training loss	Training accuracy
[1/10]	0.9535	68.70%
[2/10]	0.4037	87.26%
[3/10]	0.2874	91.13%
[4/10]	0.2202	93.07%
[5/10]	0.1825	94.34%
[6/10]	0.1465	95.65%
[7/10]	0.1230	96.15%
[8/10]	0.0920	97.25%
[9/10]	0.0724	97.94%
[10/10]	0.0597	98.41%

Figure 4.2.16.1Training accuracy of LeNet (Consonant)

Epoch No	Training loss	Training accuracy
[1/10]	1.4828	61.61%
[2/10]	0.6942	80.71%
[3/10]	0.5324	84.84%
[4/10]	0.4464	87.35%
[5/10]	0.3885	88.87%
[6/10]	0.3443	90.00%
[7/10]	0.3104	90.99%
[8/10]	0.2779	92.04%
[9/10]	0.2517	92.64%
[10/10]	0.2302	93.29%

4.2.16.1 Confusion Matrix

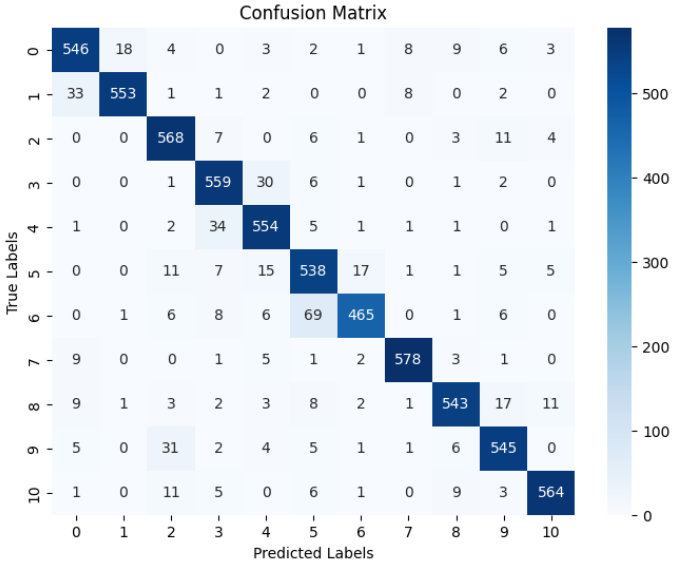


Figure 4.2.16.1 confusion matrix LeNet (Vowel)

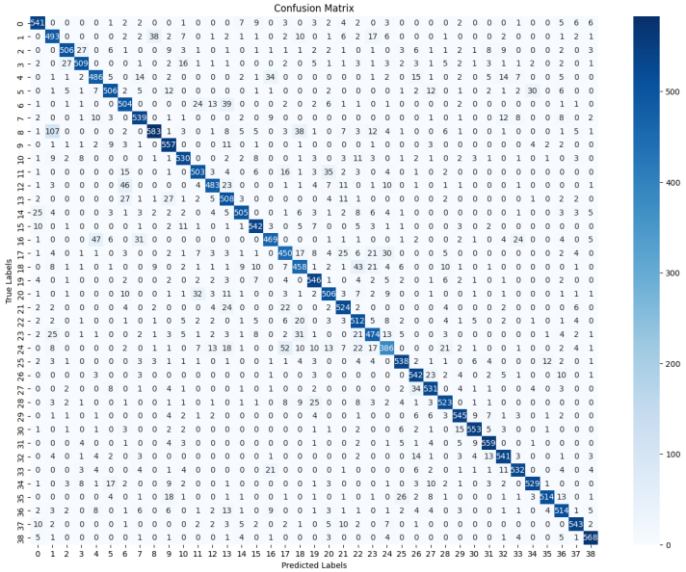


Figure 4.2.16.2 confusion matrix LeNet (Consonant)

4.2.16.1 Classification Report

Table 4.2.46 classification report LeNet (Vowel)

	precision	recall	f1-score	support
0	0.90	0.91	0.91	600
1	0.97	0.92	0.94	600
2	0.89	0.95	0.92	600
3	0.89	0.93	0.91	600
4	0.89	0.92	0.91	600
5	0.83	0.90	0.86	600
6	0.95	0.83	0.88	562
7	0.97	0.96	0.96	600
8	0.94	0.91	0.92	600
9	0.91	0.91	0.91	600
10	0.96	0.94	0.95	600
accuracy			0.92	6562
macro avg	0.92	0.92	0.92	6562
weighted avg	0.92	0.92	0.92	6562

Table 4.2.47 classification report LeNet (Consonant)

	precision	recall	f1-score	support
0	0.87	0.90	0.89	600
1	0.72	0.82	0.77	600
2	0.91	0.84	0.87	600
3	0.89	0.85	0.87	600
4	0.84	0.81	0.83	600
5	0.89	0.84	0.86	600
6	0.79	0.84	0.82	600
7	0.86	0.90	0.88	600
8	0.91	0.73	0.81	800
9	0.82	0.93	0.87	600
10	0.87	0.88	0.88	600
...				
accuracy			0.85	23612
macro avg	0.86	0.85	0.85	23612
weighted avg	0.86	0.85	0.85	23612

4.3 Result Analysis

4.3.1 Accuracy

Table 4.3.1 Accuracy Comparison of All Models of the Study

Model	Accuracy (%)	
	Vowel	Consonant
Random Forest	83.64	71.60
KNN	82.00	69.41
SVM	88.14	78.77
Logistic Regression	75.86	59.42
AdaBoost	54.23	28.48
Naive Bayes	66.73	43.53
Extra Trees	85.05	72.15
Bagging Classifier	75.18	56.29
Decision Tree	60.59	41.99
Stochastic Gradient Descent	68.64	42.85
Nearest Centroid	71.11	50.00
Linear SVM	87.75	40.60
Multinomial Naive Bayes	71.00	48.12
CustomCNN	92.96	87.72
DenseNet	97.93	95.99
LeNet	91.63	85.35

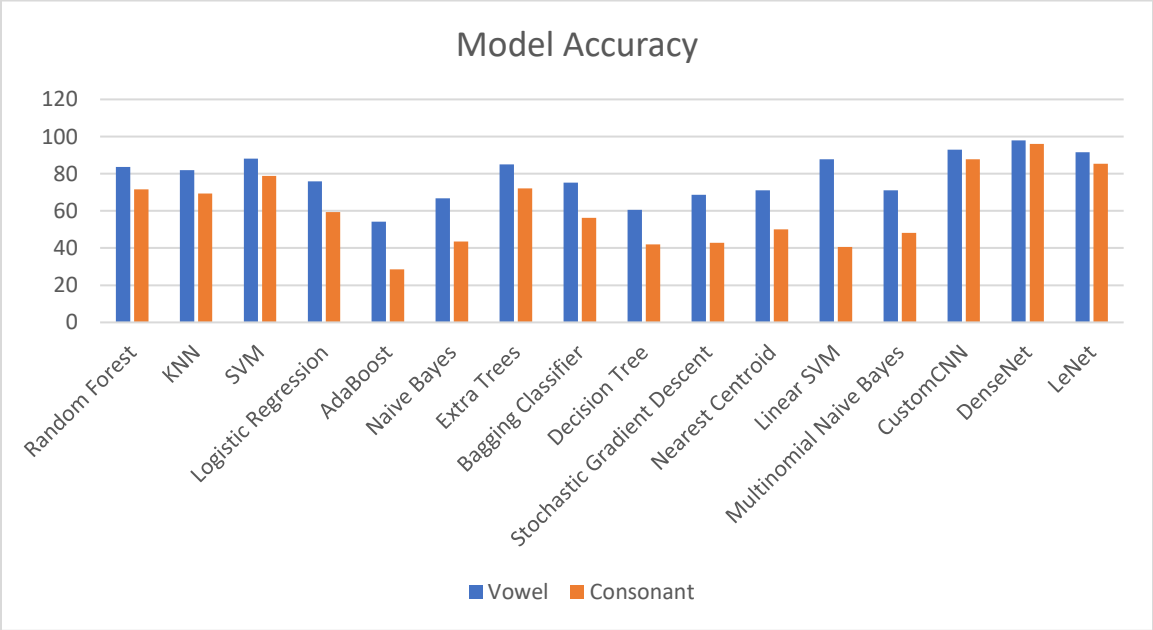


Figure 4.2.16.1 Accuracy Comparison of All Models of the Study

4.3.2 Classification Report

4.3.2.1 Vowel accuracy

Table 4.3.2 Classification Report accuracy (Vowel)

Model	F1-score	Support
Random Forest	0.84	2200
KNN	0.82	2200
SVM	0.88	2200
Logistic Regression	0.76	2200
AdaBoost	0.54	2200
Naive Bayes	0.67	2200
Extra Trees	0.85	2200
Bagging Classifier	0.75	2200
Decision Tree	0.61	2200
Stochastic Gradient Descent	0.69	2200
Nearest Centroid	0.71	2200
Linear SVM	0.65	2200
Multinomial Naive Bayes	0.71	2200
CustomCNN	0.93	6562
DenseNet	0.98	6562
LeNet	0.92	6562

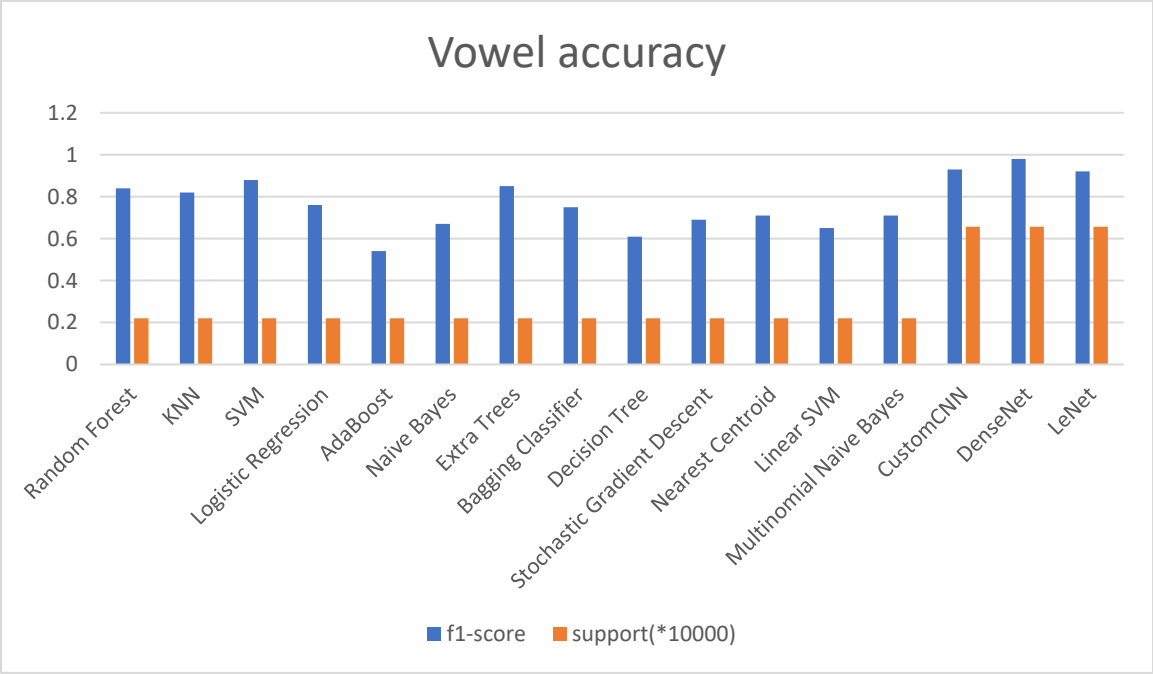


Figure 4.3.2.1 Classification Report accuracy (Vowel)

4.3.2.1 Vowel macro avg

Table 4.3.3 Classification Report macro avg (Vowel)

Model	Precision	Recall	F1-score	Support
Random Forest	0.84	0.84	0.84	2200
KNN	0.83	0.82	0.82	2200
SVM	0.89	0.88	0.88	2200
Logistic Regression	0.76	0.76	0.76	2200
AdaBoost	0.54	0.54	0.54	2200
Naive Bayes	0.68	0.67	0.67	2200
Extra Trees	0.85	0.85	0.85	2200
Bagging Classifier	0.76	0.75	0.75	2200
Decision Tree	0.61	0.61	0.61	2200
Stochastic Gradient Descent	0.72	0.69	0.68	2200
Nearest Centroid	0.71	0.71	0.71	2200
Linear SVM	0.67	0.66	0.65	2200
Multinomial Naive Bayes	0.71	0.71	0.71	2200
CustomCNN	0.93	0.93	0.93	6562
DenseNet	0.98	0.98	0.98	6562
LeNet	0.92	0.92	0.92	6562

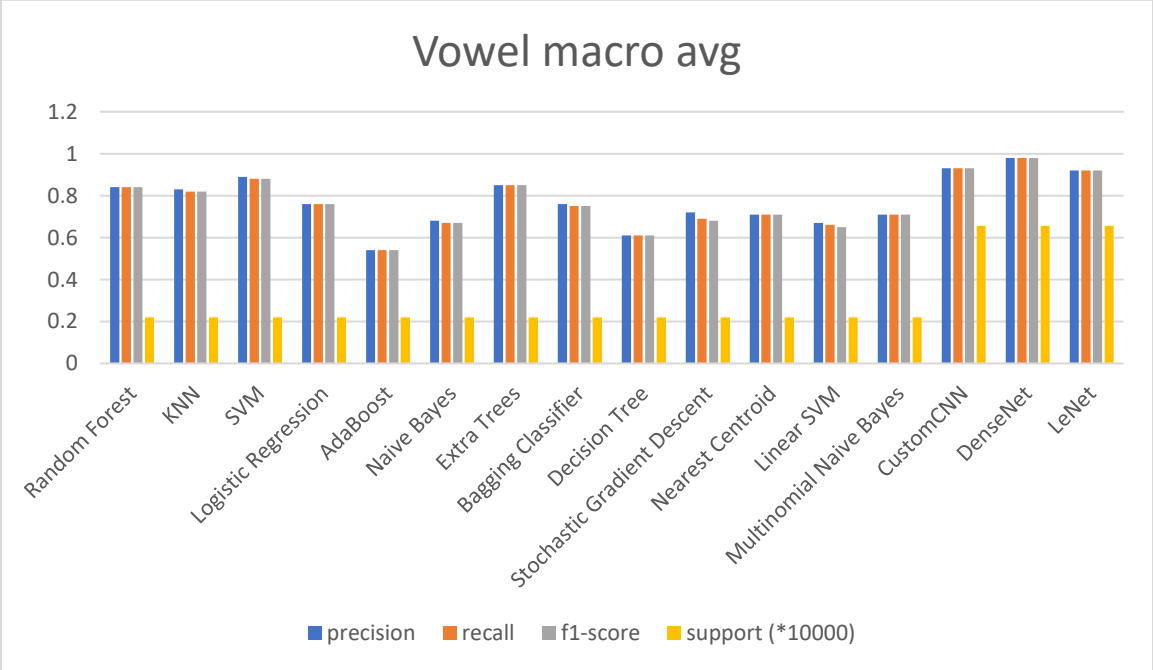


Figure 4.3.2.1 Classification Report macro avg (Vowel)

4.3.2.1 Vowel weighted avg

Table 4.3.4 Classification Report weighted avg (Vowel)

Model	Precision	Recall	F1-score	Support
Random Forest	0.84	0.84	0.84	2200
KNN	0.83	0.82	0.82	2200
SVM	0.89	0.88	0.88	2200
Logistic Regression	0.76	0.76	0.76	2200
AdaBoost	0.54	0.54	0.54	2200
Naive Bayes	0.68	0.67	0.66	2200
Extra Trees	0.85	0.85	0.85	2200
Bagging Classifier	0.76	0.75	0.75	2200
Decision Tree	0.61	0.61	0.61	2200
Stochastic Gradient Descent	0.72	0.69	0.68	2200
Nearest Centroid	0.71	0.71	0.71	2200
Linear SVM	0.67	0.65	0.65	2200
Multinomial Naive Bayes	0.71	0.71	0.71	2200
CustomCNN	0.93	0.93	0.93	6562
DenseNet	0.98	0.98	0.98	6562
LeNet	0.92	0.92	0.92	6562

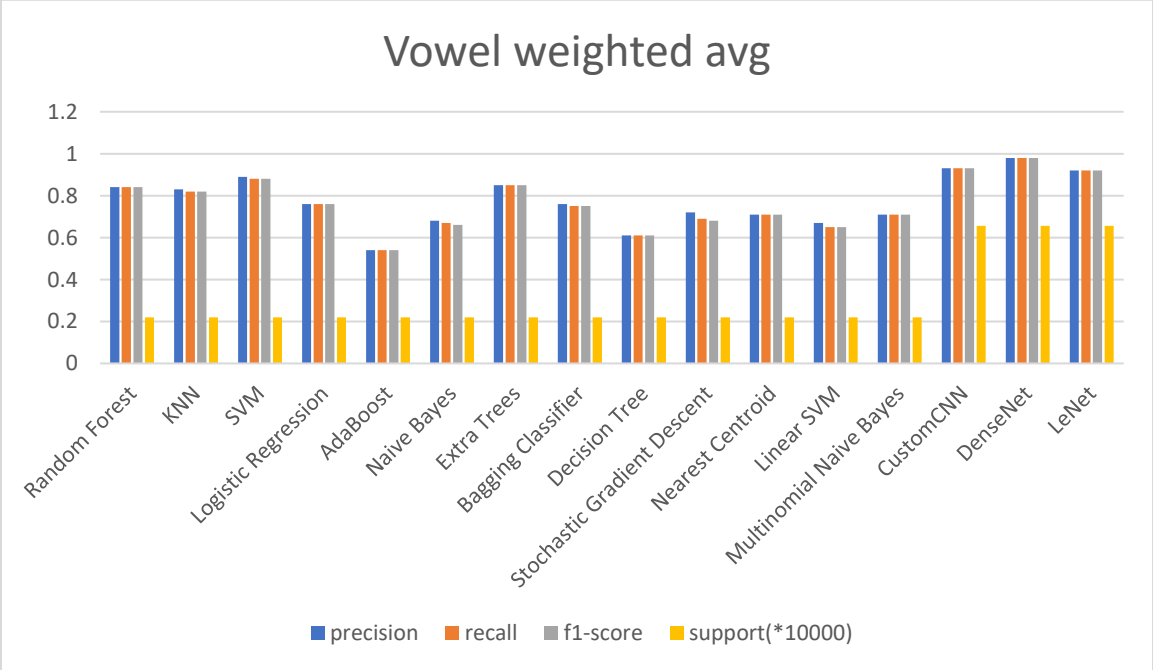


Figure 4.3.2.1 Classification Report weighted avg (Vowel)

4.3.2.1 Consonant macro avg

Table 4.3.5 Classification Report macro avg (Consonant)

Model	Precision	Recall	F1-score	Support
Random Forest	0.72	0.72	0.71	7800
KNN	0.71	0.69	0.69	7800
SVM	0.79	0.79	0.79	7800
Logistic Regression	0.59	0.59	0.59	7800
AdaBoost	0.28	0.28	0.27	7800
Naive Bayes	0.46	0.42	0.40	7800
Extra Trees	0.72	0.72	0.72	7800
Bagging Classifier	0.57	0.56	0.56	7800
Decision Tree	0.42	0.42	0.42	7800
Stochastic Gradient Descent	0.60	0.43	0.43	7800
Nearest Centroid	0.51	0.50	0.50	7800
Linear SVM	0.58	0.36	0.37	7800
Multinomial Naive Bayes	0.49	0.48	0.48	7800
CustomCNN	0.88	0.88	0.88	23612
DenseNet	0.96	0.96	0.96	23612
LeNet	0.86	0.85	0.85	23612

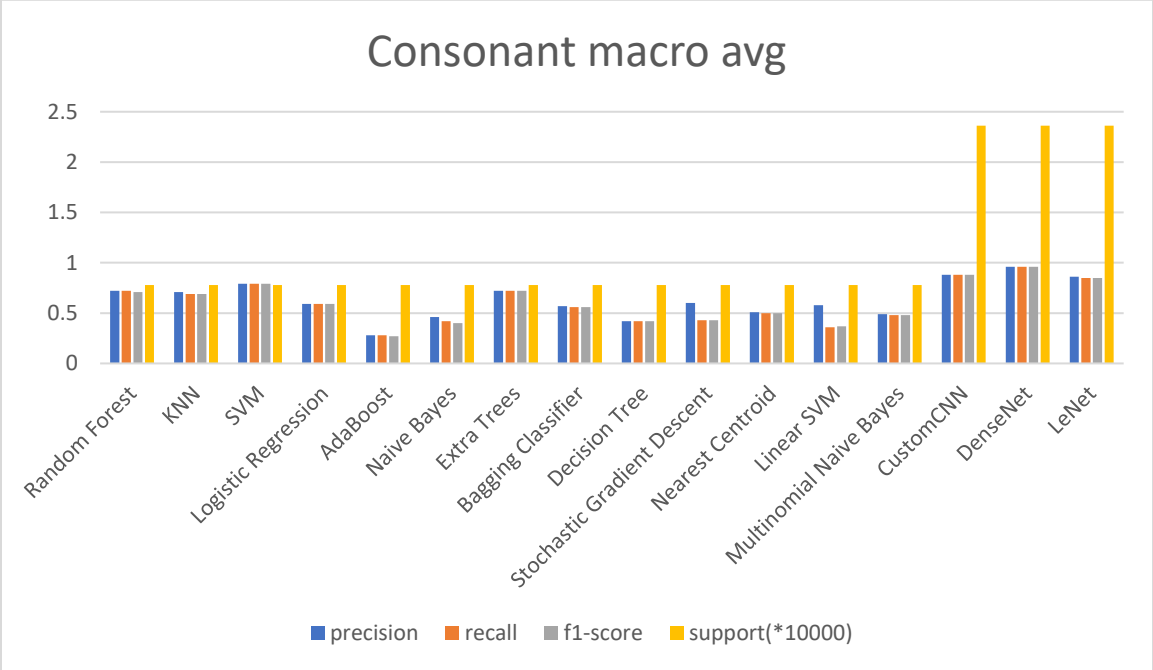


Figure 4.3.2.1 Classification Report macro avg (Consonant)

4.3.2.1 Consonant weighted avg

Table 4.3.6 Classification Report weighted avg (Consonant)

Model	Precision	Recall	F1-score	Support
Random Forest	0.72	0.72	0.71	7800
KNN	0.71	0.69	0.69	7800
SVM	0.79	0.79	0.79	7800
Logistic Regression	0.59	0.59	0.59	7800
AdaBoost	0.28	0.28	0.27	7800
Naive Bayes	0.46	0.42	0.41	7800
Extra Trees	0.72	0.72	0.72	7800
Bagging Classifier	0.57	0.56	0.56	7800
Decision Tree	0.42	0.42	0.42	7800
Stochastic Gradient Descent	0.60	0.43	0.43	7800
Nearest Centroid	0.51	0.50	0.50	7800
Linear SVM	0.57	0.37	0.37	7800
Multinomial Naive Bayes	0.49	0.48	0.48	7800
CustomCNN	0.88	0.88	0.88	23612
DenseNet	0.96	0.96	0.96	23612
LeNet	0.86	0.85	0.85	23612

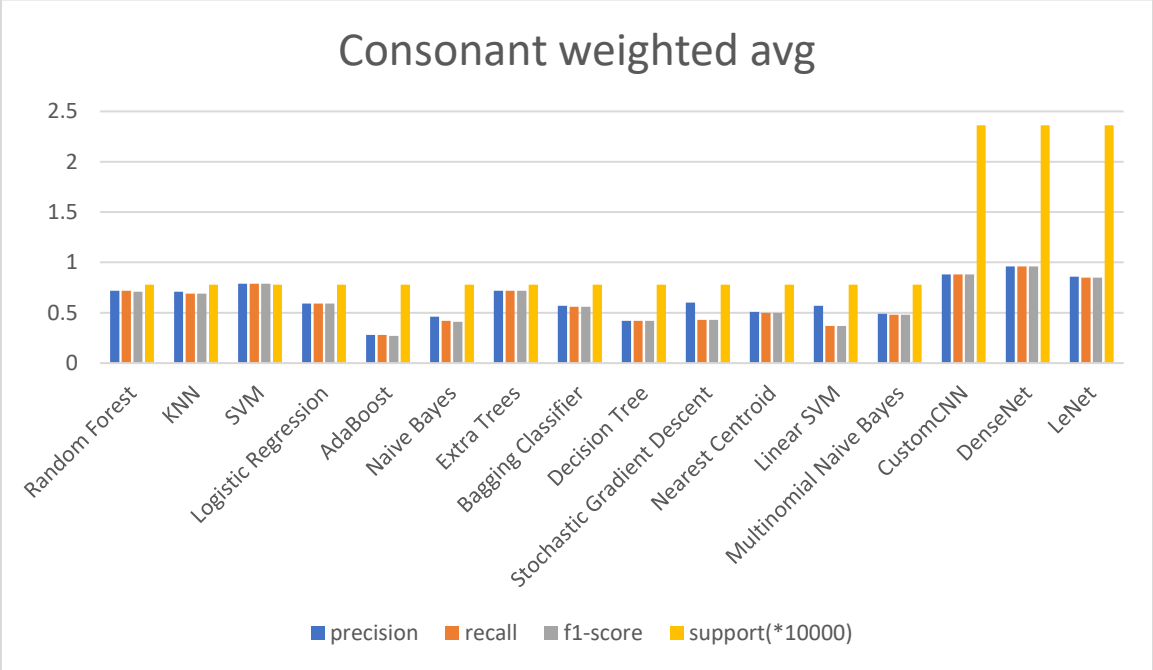


Figure 4.3.2.1 Classification Report weighted avg (Consonant)

CHAPTER 5
IMPACT ON SOCIETY, ENVIRONMENT AND SUSTAINABILITY

5.1 Impact on Society

This kind of research can have considerable impact on numerous aspects of Bangladeshi society:

5.1.1 Education:

- Literacy assessment and feedback in schools could be revolutionized by increased speed and accuracy of Bangla handwriting recognition.
- Learning experiences could be improved by the creation of digital educational resources like interactive e-books and tablets with handwriting recognition.
- By automating the grading of handwritten exams, educators can free up more time and resources to concentrate on individualized instruction.

5.1.2 Accessibility:

- People who have trouble typing or seeing can find digital interfaces and services more accessible with handwriting recognition.
- The creation of assistive technology, such as screen readers that can recognize Bangla, has the potential to empower people with disabilities.

5.1.3 Language preservation:

- Preserving cultural heritage through the digitization of historical documents and literature in Bangla can be made easier with accurate handwriting recognition.
- The creation of machine translation programs that can recognize handwritten text could improve cross-language communication and information access.

5.1.4 Economic development:

- Effective handwriting recognition in Bangla has the potential to enhance document processing and automation across multiple industries, thereby augmenting productivity and efficiency.
- The creation of new handwriting recognition-based apps and services may spur innovation in the tech industry and lead to the creation of jobs.

5.1.5 Social inclusion:

- By utilizing Bangla handwriting recognition to make digital tools and services accessible, we can close the digital divide and give marginalized communities more power.
- By encouraging Bangla usage in a variety of settings, recognition technologies can support inclusivity and cultural identity.

All things considered, this research has the potential to greatly increase educational accessibility, support language preservation, improve accessibility, stimulate economic growth, and alleviate social inequality in Bangladesh.

5.2 Impact on Environment

Positive as well as negative impacts on the environment are possible. This is an explanation:

Positive Impacts:

- Decreased use of paper: By reducing the need for paper forms and documents, accurate and effective handwriting recognition can help save trees and other resources.

- Digital document storage: Transforming handwritten documents into digital formats minimizes the amount of energy needed for preservation while also saving physical storage space and the need for paper archives.
- Better delivery and logistics: By automating address recognition and sorting, handwriting recognition can improve delivery and logistics procedures and possibly lower transportation emissions.

- Accessibility for remote communities: Handwriting recognition can lessen the need for travel and the emissions that go along with it by providing digital access to services and information for communities with limited physical accessibility.

Negative Impacts:

- Higher energy consumption: Complex CNN models require a lot of processing power to train and operate, which raises the energy consumption in data centers.
- E-waste generation: If handwriting recognition software is used more frequently, it may lead to an increase in e-waste if it is not managed properly.
- Environmental impact of hardware production: Resource extraction and processing result in an environmental impact during the manufacturing of computer hardware and associated infrastructure for handwriting recognition technologies.

Overall, how the technology is created, implemented, and used will determine how CNN-based Bangla handwriting recognition affects the environment. Researchers and developers can optimize the positive environmental benefits while mitigating the negative ones by emphasizing sustainability and responsible practices.

5.3 Ethical Aspect

CNN-developed handwriting recognition technology for Bangla holds promise for bettering education, increasing accessibility, and preserving the language. However, there are many ethical issues to be concerned about, such as potential exclusion, privacy concerns, and bias in the data.

Technology can be developed responsibly by putting diversity, justice, and inclusivity first in order to maximize benefits and minimize drawbacks.

5.4 Sustainability

The environmental impact of handwriting recognition is reduced by using renewable energy sources and carefully selecting hardware. Ensuring accessibility and inclusivity for individuals from diverse socioeconomic backgrounds and dialects is crucial in order to optimize the positive effects on Bangladeshi society. Equality of access and green technology are essential to sustainability.

CHAPTER 6
OVERVIEW, CONCLUSION, AND IMPLEMENTATION FOR FUTURE RESEARCH

6.1 Overview of the Study

I went on a quest of exploration, delving into the complex world of algorithms and the Ekush dataset. Every step and every investigation uncovered a new aspect of this intricate puzzle. The abundance of images appeared to be the secret to accuracy, a harmonious arrangement of brushstrokes creating a more lucid image. However, vowels, their individual voices, came through clearly, while consonants, like naughty twins, mirrored and mimicked one another, creating confusion and lowering the brightness of accuracy. Small image-capable algorithms gracefully traversed this complex landscape, and distance-based techniques—especially the intriguing SVM—unlocked the dataset's mysteries with unexpected ease. These insights hint at the way forward: broadening the chorus within the dataset, separating the consonant twins, and fostering the symbiotic relationship between algorithms and data. Because the secret to releasing the enormous potential of Bangla handwriting recognition is hidden within this complex dance. This transformation has the potential to revolutionize education, accessibility, and the very fabric of language in Bangladesh.

While delivering the main conclusions of your research, this narrative approach is more captivating and expressive than the bullet point format.

6.2 Conclusion

During my exploration of the Ekush dataset, I trained more than sixteen models, all of which were expertly constructed and modified by me. Of them, DenseNet was the clear winner, achieving remarkable accuracy in both vowel and consonant recognition (97.93% for vowels and 95.99% for consonants). It truly demonstrated the mastery of neural networks! However, SVM—the brave seasoned pro in machine learning—remained the top performer in the traditional model class, with

accuracy of 88.14% and 78.77%, respectively. But a strange pattern became apparent: all other models, in spite of their best attempts, appeared to prefer the dataset's vowel section, finding it more difficult to handle the consonant world's subtle complexities. This intriguing observation invites more research and opens up possibilities.

6.3 Implication for Future Study

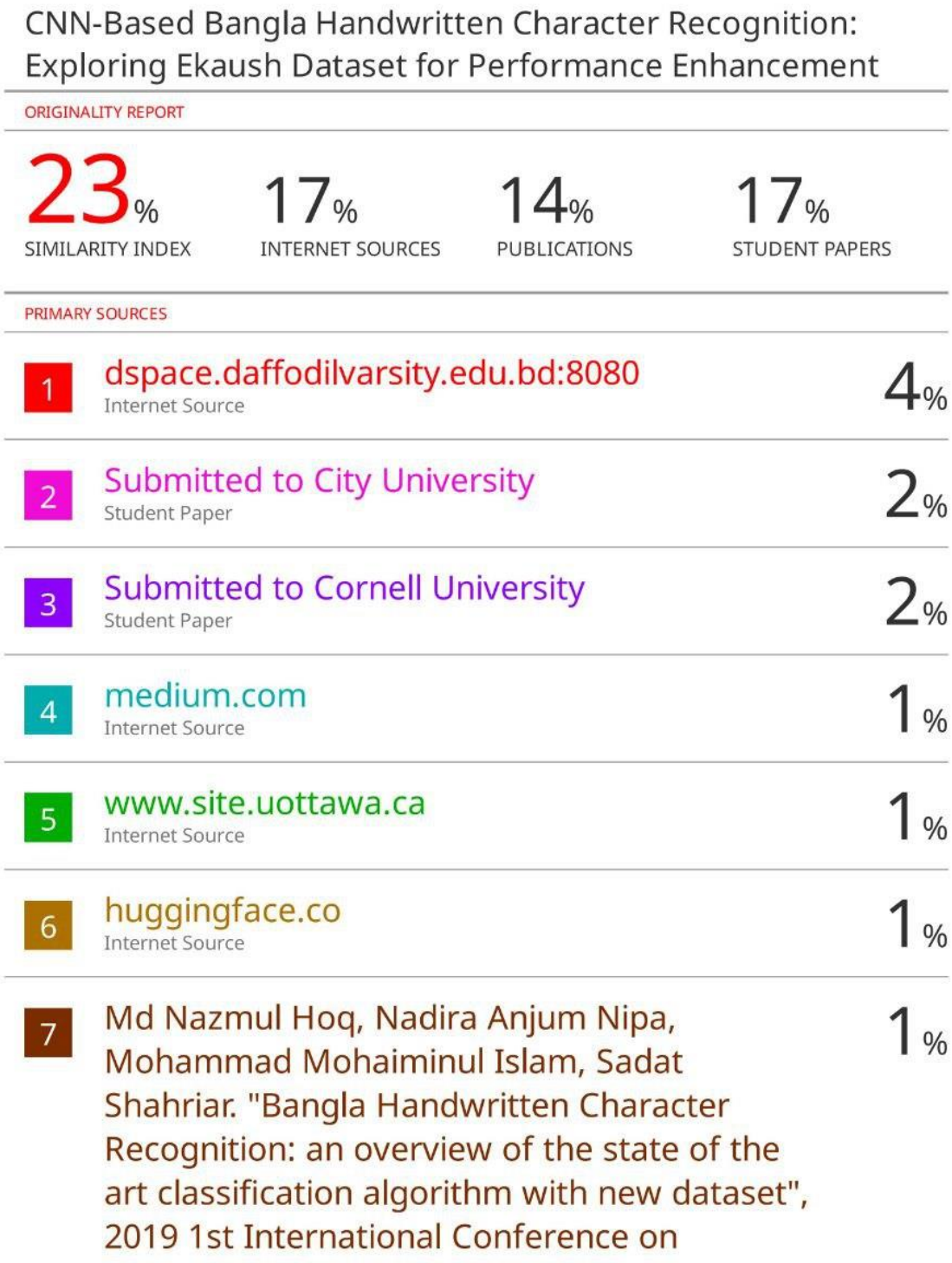
1. **Data Augmentation and Exploration:**
    a. Increase the size and diversity of the dataset: Work with organizations and people to collect more varied handwritten samples for testing and training, which will improve the generalizability of the model.
    b. Make use of sophisticated data augmentation methods: To further increase model robustness, experiment with methods such as Generative Adversarial Networks (GANs) to generate synthetic data that resembles real-world variations.
2. **Architecture and Algorithm Exploration:**
    a. Examine transfer learning: Using pre-trained CNNs on larger datasets such as ImageNet, investigate the possibilities of transfer learning and refine them for Bangla character recognition.
    b. Experiment with hybrid models: To capture sequential patterns in handwritten strokes and possibly increase accuracy, combine CNNs with other methods such as recurrent neural networks (RNNs).
3. **Algorithm-Specific Explorations:**
    a. Random Forest, Extra Trees, and Bagging Classifier: Investigate methods for managing high-dimensional data and choosing features, as these algorithms may be affected by the complexity of the feature space.
    b. SVM: To maximize performance for classes that are not linearly separable, investigate various kernel functions and parameter tuning techniques.

c. CustomCNN: Investigate cutting-edge architectures customized for Bangla character features, possibly with the addition of capsule networks or attention mechanisms for enhanced feature representation.

4. **Addressing Challenges and Real-World Applications:**
    a. Class imbalance: Create tactics to counteract it, like employing cost-sensitive learning to enhance the recognition of uncommon characters or oversampling minority classes.
    b. Variations and noise: By using regularization and data augmentation techniques, you can increase the model's resistance to variations in handwriting styles, noise, and distortions.
    c. Implementation in the real world: Take into account hardware limitations and computational efficiency when exploring integration into mobile apps, document scanning systems, and educational resources.
5. **Interdisciplinary Collaboration and Knowledge Sharing:**
    a. Encourage collaboration: Work with cognitive scientists, linguists, and educational researchers to better understand language-specific difficulties and human handwriting processes. This knowledge will help with the development and assessment of models.
    b. Make a contribution to open-source materials: Exchange code, datasets, and research results to spur additional study and advance the field of handwritten character recognition in Bangla.

REFERENCE

1. Basu, Subhadip, et al. "Handwritten Bangla digit recognition using classifier combination through DS technique." Pattern Recognition and Machine Intelligence: First International Conference, PReMI 2005, Kolkata, India, December 20-22, 2005. Proceedings 1. Springer Berlin Heidelberg, 2005.
2. Alom, Md Zahangir, et al. "Handwritten bangla character recognition using the state-of-the-art deep convolutional neural networks." Computational intelligence and neuroscience 2018 (2018).
3. DeFries, R. S., and Jonathan Cheung-Wai Chan. "Multiple criteria for evaluating machine learning algorithms for land cover classification from satellite data." Remote Sensing of Environment 74.3 (2000): 503-515.
4. Kamavisdar, Pooja, Sonam Saluja, and Sonu Agrawal. "A survey on image classification approaches and techniques." International Journal of Advanced Research in Computer and Communication Engineering 2.1 (2013): 1005-1009.
5. Sharma, Arun, and Vidushi Sharma. "An Empirical Study of Supervised Learning Techniques on Multispectral Dataset."
6. Shamim, S. M., et al. "Handwritten digit recognition using machine learning algorithms." Global Journal Of Computer Science And Technology 18.1 (2018): 17-23.
7. Al-Amin, Md, Md Saiful Islam, and Shapan Das Uzzal. "Sentiment analysis of Bengali comments with Word2Vec and sentiment information of words." 2017 international conference on electrical, computer and communication engineering (ECCE). IEEE, 2017.
8. Das, Nibaran, et al. "A genetic algorithm based region sampling for selection of local features in handwritten digit recognition application." Applied Soft Computing 12.5 (2012): 1592-1606.
9. Surinta, Olarik, et al. "Recognition of handwritten characters using local gradient feature descriptors." Engineering Applications of Artificial Intelligence 45 (2015): 405-414.
10. Das, Nibaran, et al. "Recognition of handwritten Bangla basic characters and digits using convex hull based feature set." arXiv preprint arXiv:1410.0478 (2014).
11. Liu, Cheng-Lin, and Ching Y. Suen. "A new benchmark on the recognition of handwritten Bangla and Farsi numeral characters." Pattern Recognition 42.12 (2009): 3287-3295.
12. Alom, Md Zahangir, et al. "Handwritten bangla digit recognition using deep learning." arXiv preprint arXiv:1705.02680 (2017).
13. Noor, Rouhan, Kazi Mejbaul Islam, and Md Jakaria Rahimi. "Handwritten Bangla numeral recognition using ensembling of convolutional neural network." 2018 21st international conference of computer and information technology (ICCIT). IEEE, 2018.
14. Shawon, Md Tanvir Rouf, Raihan Tanvir, and Md Golam Rabiul Alam. "Bengali Handwritten Digit Recognition using CNN with Explainable AI." 2022 4th International Conference on Sustainable Technologies for Industry 4.0 (STI). IEEE, 2022.



CNN-Based Bangla Handwritten Character Recognition: Exploring Ekaush Dataset for Performance Enhancement

ORIGINALITY REPORT

23%	17%	14%	17%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dspace.daffodilvarsity.edu.bd:8080 Internet Source	4%
2	Submitted to City University Student Paper	2%
3	Submitted to Cornell University Student Paper	2%
4	medium.com Internet Source	1%
5	www.site.uottawa.ca Internet Source	1%
6	huggingface.co Internet Source	1%
7	Md Nazmul Hoq, Nadira Anjum Nipa, Mohammad Mohaiminul Islam, Sadat Shahriar. "Bangla Handwritten Character Recognition: an overview of the state of the art classification algorithm with new dataset", 2019 1st International Conference on	1%