

HANDWRITTEN HINDI CHARACTER RECOGNITION USING CNN

**Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering**

By

**Ansh Saxena (38110037)
Anurag Nishad (38110040)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND
TECHNOLOGY (DEEMED TO BE
UNIVERSITY)**

**Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by
AICTE**

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH-2022



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)



Accredited with Grade “A” by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI

CHENNAI– 600119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Ansh Saxena (Reg. No: 38110037)** and **Anurag Nishad (Reg. No: 38110040)** who carried out the project entitled “**Handwritten Hindi Character Recognition using CNN**” under my supervision from November 2021 to May 2022.

Guide – DR MURUGAN S M.E., P.HD

Head of the Department – Dr. L. LAKSHMANAN

Submitted for Viva voice Examination held on:_____

Internal Examiner

External Examiner

DECLARATION

I, **Ansh Saxena and Anurag Nishad** hereby declare that the project report entitled **“Handwritten Hindi Character Recognition using CNN”** done by me under the guidance of **Dr. Murugan S M.E., P.HD** at “Sathyabama Institute of Science and Technology (Deemed to be University), Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600119” is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering/Technology degree in Information Technology.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE:

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D, Dean, School of Computing, Dr. S. Vigneshwari, M.E., Ph.D. and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. Murugan S M.E., P.HD** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

The goal of this study is to employ a Convolutional Neural Network (CNN)-based Deep Learning model to recognize handwritten Hindi characters. The identified characters can then be saved digitally or used for other purposes. The data was taken from the Kaggle dataset, which has 92,000 photos separated into two sets: training (80 percent) and test (20 percent). It comprises a variety of handwritten Devanagari characters produced by various people that can be used to train and test handwritten text recognizers as well as conduct writer identification and verification tests. On top of a TensorFlow backend, the model is built with Keras libraries. For recognition, it has four CNN layers followed by three fully connected layers. As input, grayscale handwritten character pictures are employed. Filters are used to extract different features from the photos at each layer. The Convolution operation does this. Pooling and flattening are the other two major activities involved. The CNN layers' output is delivered to the fully connected layers. Finally, each character's chance or probability score is calculated, and the character with the highest probability score is displayed as the output. 92.4% recognition accuracy is achieved. Similar models exist for the purpose, but the new model outperformed and outperformed some of the older models in terms of performance and accuracy

TABLE OF CONTENTS

ABSTRACT	5
TABLE OF CONTENTS	6
LIST OF FIGURES	7
LIST OF TABLES	7
ABBREVIATIONS	8
REFERENCES	9
KEYWORDS	10
1. Introduction	11
1.1. Introduction	
1.2. Aim and Objective	
1.3. Motivation	
1.4. Software Requirements Specification	
2. Literature Review	14
2.1. Different approaches used	
2.2. Comparison of Existing methods with merits and demerits	
3. Methodology	16
3.1. Models Used	
3.2. Architecture diagram	
4. Implementation	19

4.1.	Data collection and analysis	
4.2.	Data Preprocessing	
4.3.	Tokenization	
4.4.	Feature Extraction	
4.5.	Splitting the dataset	
4.6.	Model Training and verification	
5.	Libraries Used	22
5.1.	Numpy	
5.2.	Pandas	
5.3.	Sklearn	
6.	Coding and Testing	25
7.	Conclusion and Enchantment	28

LIST OF FIGURES

● Architecture Diagram	18
------------------------	----

LIST OF TABLES

● Literature Review	14
● Accuracy Table	28

ABBREVIATIONS

ML- Machine Learning

np - Numpy

pd – Pandas

CNN – Convolutional Neural Network

Reference :

- [1] Apoorva Chaudhary, Raushan Lal Chhoker ,”Handwritten Hindi Numeric Character Recognition and comparison of Algorithms”, 7th International Conference on Cloud Computing, Data Science & Engineering – Conference (2017).
- [2] R.Parthiban,R.Ezhilarasi,D.Saravanan,”OpticalCharacter Recognition for English Handwritten Text Using Recurrent Neural Network”, University of Gothenburg (2020).
- [3] Ripal Patel, Palak Patel,” Handwritten Character Recognition using Neural Network”, International Journal of Scientific & Engineering Research Volume 2, Issue 3, March-2011.
- [4] Mahmoud M. Abu Ghosh, Ashraf Y. Maghari, ” handwritten digit recognition using neural networks research paper”, 2017 International Conference on Promising Electronic Technologies (2017).
- [5] Parshuram Kamble, Ravinda, Hegadi,” Handwritten Marathi character recognition using R-HOG Feature”, International Conference on Advanced Computing Technologies and Applications(2015).
- [6] M. A. Pragathi, S. Saveetha, K. Priyadarshini, A. Shavar Banu, K. O. Mohammed Aarif, ”Handwritten Tamil Character Recognition Using Deep Learning”,2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN) (2019).
- [7] Naila , Adnan,” Urdu Optical Character Recognition Systems: Present Contributions and Future Directions”,2018 IEEE.
- [8] Kaensar,” A Comparative Study on Handwritten Digit Recognition Classifier Using Neural Network”, 2019 IEEE.
- [9] Sujatha, Lalitha,” A Survey on Offline Handwritten Text Recognition of Popular Indian Scripts”, International Journal of Computer Science and Engineering 2019.
- [10] Wu, Chen.” Image Recognition Based on Deep Learning”, 2015 IEEE
- [11] Sheng ,” A Review of Gradient-Based and Edge-Based Feature Extraction Method for Object Detection”, 2011 IEEE

KEYWORDS

Handwritten Hindi Characters;
Devanagari;
Keras;
Convolutional Neural Network;

1.INTRODUCTION

1.1 INTRODUCTION:

In today's environment, handwritten text recognition technology is extremely useful and necessary. Physical data storage and retrieval are challenging, and there is a risk of harm or loss. Physical data generation is also prone to errors. If the text is in a digital format, however, error scanning methods and autocorrect tools can aid in the correct and efficient storage of data. This is one of the most difficult jobs in the field of pattern recognition. One of the most significant benefits of digitally preserving text is that it can be retrieved from any location. You do not need to be present at the location where the data is stored. Data storage, access, and analysis have all become considerably easier because of this technology. Optical character recognition deals with an important issue called the character classification and this is more challenging because of the similarities between them. Character classification is something that identifies the characters in Hindi that are trained initially and it can identify the characters written by different users. Hindi language is most popular and its techniques grammar are formulated for the recognition of a Hindi script is one of the foremost among the 16 major national languages spoken reading and writing by the North Indian people. Character recognition can be online or offline. In On-line character recognition system, the representation of two-dimensional coordinates of successive points is done. It is automatic conversion of text into a digital form. Off-line character recognition is used to convert written text into letter codes. Since there are many hidden layers in deep neural network the parameters for the training is very huge. To prevent over fitting we require a large set of examples. One such magnificence of deep mastering is the Convolution Neural Network. Convolution Neural Network is a special type of neural network used effectively for image recognition and classification. By implementing these techniques, we recognize the handwritten Hindi characters. Hindi is one of the ancient languages spoken by many people.

1.2 AIM AND OBJECTIVES

- The aim of a handwriting character recognition system is to convert handwritten character into machine readable formats.
- The main objective of this work is to ensure effective and reliable approaches for hand-written character recognition and making machine readability easier.
- Handwritten character recognition is meant for receiving and interpreting handwritten input in the form of pictures or paper documents.

1.3 MOTIVATION

Hand writing recognition of characters has been around since the 1980s. The task of handwritten character recognition, using a classifier, has great importance and use such as – online handwriting recognition on computer tablets, recognize zip codes on mail for postal mail sorting, processing bank check amounts, numeric entries in forms filled up by hand like tax forms, bank forms and so on. The general problem we predicted we would face in this digit classification problem was the similarity between the digits like 1 and 7, 5 and 6, 3 and 8, 9 and 8 etc. Also people write the same digit in many different ways - the digit '1' is written as '1', '1', '1' or '1'. Similarly 7 may be written as 7, 7, or 7. Finally the uniqueness and variety in the handwriting of different individuals also influences the formation and appearance of the digits.

1.4 SOFTWARE AND HARDWARE REQUIREMENTS SPECIFICATION:

- Any modern Intel or AMD processor
- 4 GB RAM
- 50 GB or more storage
- Stable Internet Connection
- OS – Windows 7 or Above
- Python 3.7
- Anaconda Navigator

2. LITERATURE REVIEW

2.1 DIFFERENT APPROACHES USED:

AUTHOR	YEAR OF PUBLICATION	DESCRIPTIONS	PROS	CONS
Apoorva Chaudhary , Raushan Lal Chhoker	2017	Handwritten Hindi Numeric Character Recognition	Multiple algorithms used – HOG, PCA, KNN, ANN	Only works for hindi digit recognition
			Around 100 different handwritings used	Feature selection and Classification are separate
AUTHOR	YEAR OF PUBLICATION	DESCRIPTIONS	PROS	CONS
R. Parthiban, R. Ezhilarasi, D. Saravanan	2020	Optical Character Recognition for English Handwritten Text Using Recurrent Neural Network	Indexing time is low because of faster classification	Since OCR is used Accuracy and Reliability is never 100 percent
			Helps in keeping Structured Database Record	Search Time is higher compared to indexing

AUTHOR	YEAR OF PUBLICATION	DESCRIPTIONS	PROS	CONS
Mahmoud M. Abu Ghosh, Ashraf Y. Maghari	2017	A Comparative Study on Handwriting Digit Recognition Using Neural Networks	It automatically detects the important features without human supervision	Only gives Basic insights on Neural Network
			It uses three neural network algorithms to determine best result	Data set is very standard level
AUTHOR	YEAR OF PUBLICATION	DESCRIPTIONS	PROS	CONS
Parshuram M. Kamble, Ravinda S. Hegadi	2015	Handwritten Marathi Character Recognition using R-HOG Feature	HOG feature extraction and FFANN classification is more effective with processing speed and accuracy	CNN is comparatively better in every way
				CNN is more successful in human detection method

AUTHOR	YEAR OF PUBLICATION	DESCRIPTIONS	PROS	CONS
Naila Habib Khan, Awais Adnan	2018	Urdu Optical Character Recognition Systems	Indexing time is low because of faster classification	Since OCR is used Accuracy and Reliability is never 100 percent
			Helps in keeping Structured Database Record	Search Time is higher compared to indexing
AUTHOR	YEAR OF PUBLICATION	DESCRIPTIONS	PROS	CONS
M. A. Pragathi, S. Saveetha	2019	Handwritten Tamil Character Recognition Using Deep Learning	The Efficiency achieved is higher due to deep learning	Some hyperparameters could take longer time to converge
K. Priyadarshini, A. Shavar Banu			Deep learning performs better with huge amount of data	It is extremely expensive to train data
K. O. Mohammed Aarif				

3. METHODOLOGY

3.1 MODELS USED:

- **CONVOLUTIONAL NEURAL NETWORK**

A Convolutional Neural Network is a Deep Learning algorithm which can take in an input image, assign importance to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

1. Convolutional Layer

This layer is the first layer that is used to extract the various features from the input images. In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size $M \times M$. By sliding the filter over the input image, the dot product is taken between the filter and the parts of the input image with respect to the size of the filter ($M \times M$). The output is termed as the Feature map which gives us information about the image such as the corners and edges. Later, this feature map is fed to other layers to learn several other features of the input image.

2. Pooling Layer

In most cases, a Convolutional Layer is followed by a Pooling Layer. The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs. This is performed by decreasing the connections between layers and independently operates on each feature map. Depending upon method used, there are several types of Pooling operations. In Max Pooling, the largest element is taken from feature map. Average Pooling calculates the average of the elements in a predefined sized Image section. The total sum of the elements in the predefined section is computed in Sum Pooling. The Pooling Layer usually serves as a bridge between the Convolutional Layer and the FC Layer Must Read: Neural Network Project Ideas

3. Fully Connected Layer

The Fully Connected (FC) layer consists of the weights and biases along with the neurons and is used to connect the neurons between two different layers. These layers are usually placed before the output layer and form the last few layers of a CNN Architecture. In this, the input image from the previous layers are flattened and fed to the FC layer. The flattened vector then undergoes few more FC layers where

the mathematical functions operations usually take place. In this stage, the classification process begins to take place.

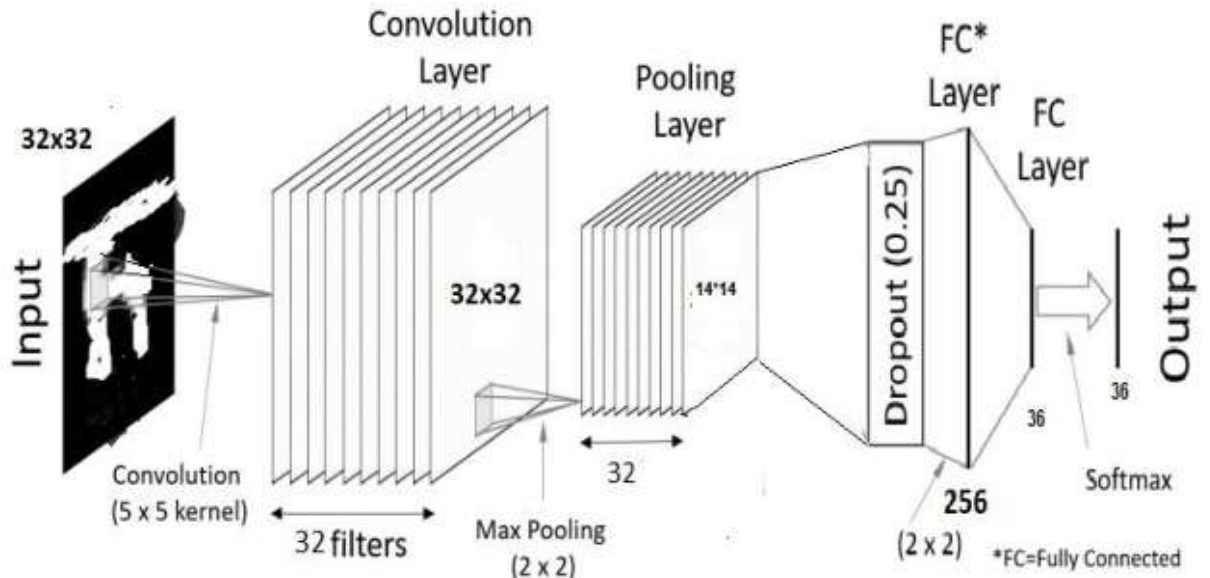
4. Dropout

Usually, when all the features are connected to the FC layer, it can cause overfitting in the training dataset. Overfitting occurs when a particular model works so well on the training data causing a negative impact in the model's performance when used on a new data. To overcome this problem, a dropout layer is utilised wherein a few neurons are dropped from the neural network during training process resulting in reduced size of the model. On passing a dropout of 0.3, 30% of the nodes are dropped out randomly from the neural network.

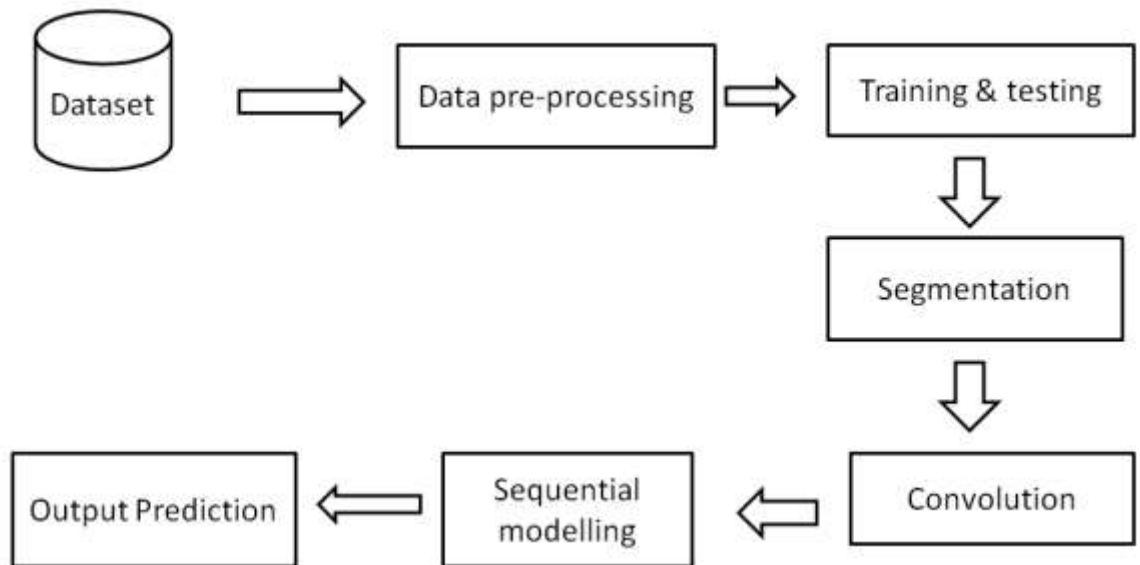
5. Activation Functions

Finally, one of the most important parameters of the CNN model is the activation function. They are used to learn and approximate any kind of continuous and complex relationship between variables of the network. In simple words, it decides which information of the model should fire in the forward direction and which ones should not at the end of the network.

3.2 ARCHITECTURE DIAGRAM AND WORKFLOW



ARCHITECTURE DIAGRAM



4. IMPLEMENTATION

4.1 DATA COLLECTION AND ANALYSIS

This is a dataset of Devanagari Script Characters. It comprises of 92000 images [32 x 32 px] corresponding to 64 characters, consonants “ka” to “gya”, and the digits 0 to 9. The data’s dimension is of 92000 * 1025. There are 1024 input features of pixel values in greyscale from 0 to 255. The column “character” represents the Devanagari Character Name corresponding to each image. There are few publicly available datasets for Devanagari Characters on Kaggle.

4.2 PRE-PROCESSING DATA

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. The main aim of Data Cleaning is to identify and remove errors and duplicate data, tokenization, removing stop words and stemming words to the root word in order to create a reliable dataset.

4.3 SEGMENTATION

In this stage, the image is converted into sub-images of individual character. The pre-processed image is segmented to isolated characters by assigning a number to each sub-image using labels. These labels provide information about the number of characters in the sub-image. Each character is uniformly sized accordingly into pixels. There are many differences in the sizes of the character, so we need to normalize the size.

4.5 FEATURE EXTRACTION

CNN is a neural network that extracts input image features and another neural network classifies the image features. The input image is used by the feature extraction network. The extracted feature signals are utilized by the neural network for classification

5.LIBRARIES USED: NUMPY , PANDAS, SCIKIT-LEARN (SKLEARN)

5.1 NUMPY :

Numpy is an array processing library that may be used to do a wide range of tasks. It comes with a high-performance multidimensional array object and tools to manipulate it. For scientific computing, it is the most important Python module. In addition to its obvious scientific applications, Numpy can be used as a multi-dimensional container of common data.

Some of the following features of Numpy are:

- Numpy Array is a fast and space-efficient multidimensional array with vectorized arithmetic operations and advanced broadcasting features.
- Standard mathematical functions can be utilised to conduct rapid operations on large arrays of data without the need for loops.
- Reading/writing array data to disc and working with memory-mapped files
- Toolkits for integrating C, C++, and Fortran code Capabilities in linear algebra, random number generation, and the Fourier transform

5.2 PANDAS :

Pandas is a Python package that allows you to work with large data sets. It offers tools for data analysis, cleansing, exploration, and manipulation. Wes McKinney came up with the name "Pandas" in 2008, which refers to both "Panel Data" and "Python Data Analysis."

Pandas provides you with data-related responses. Like:

- Is there a link between two or more of the columns?
- What is the average value of something?
- What is the maximum value?
- What is the minimum value?

Pandas can also delete rows that are no longer relevant or contain incorrect values, such as empty or NULL values. This is referred to as data cleaning.

5.3 SCIKIT-LEARN (SKLEARN)

In Python, Scikit-learn (Sklearn) is the most usable and robust machine learning library. It uses a Python consistency interface to give a set of efficient tools for machine learning and statistical modelling, such as classification, regression, clustering, and dimensionality reduction. NumPy, SciPy, and Matplotlib are the foundations of this package, which is mostly written in Python.

Almost all common supervised learning algorithms, such as Linear Regression, Support Vector Machine (SVM), Decision Tree, and others, are included in scikit-learn.

On the other hand, it includes all of the popular unsupervised learning algorithms, including clustering, factor analysis, PCA (Principal Component Analysis), and unsupervised neural networks.

Some of the important features of Sklearn are:

Clustering is a model for grouping data that hasn't been labelled.

Dimensionality reduction is a technique for lowering the amount of qualities in data so that it can be summarised, visualised, and features selected.

Ensemble approaches combine the predictions of numerous supervised models, as the name implies.

Feature extraction is a technique for extracting data features and defining properties in picture and text data.

Feature selection is a technique for identifying interesting qualities that can be used to build supervised models.

6. CODING AND TESTING:

Step 1 : For data manipulation, visualization and deep learning

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import random
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from keras.layers import Conv2D, BatchNormalization, MaxPool2D,
GlobalMaxPool2D, TimeDistributed, GRU, Dense, Dropout
```

Step 2 : Reading the data into a dataframe

```
df = pd.read_csv("data.csv")
df.head()
```

Step 3 : Splitting data into training and testing sets

```
train_df, test_df = train_test_split(df, test_size=0.2)
train_df = train_df.reset_index(drop=True)
test_df = test_df.reset_index(drop=True)
x_train, x_test = train_df.drop('character', axis=1),
test_df.drop('character', axis=1)
y_train, y_test = train_df['character'], test_df['character']
```

Step 4 : All output classes

```
all_chars = df['character'].unique()
```

```
print(f'All characters: {all_chars}')  
print(f'Number of characters: {len(all_chars)}')
```

Step 5 : Converting DataFrame having pixel values into numpy arrays

```
x_train, x_test = x_train.values.reshape(x_train.values.shape[0], 32,  
32), x_test.values.reshape(x_test.values.shape[0], 32, 32)  
print(f'Shape of Training Image array: {x_train.shape}')  
print(f'Shape of Testing Image array: {x_test.shape}')
```

Step 6 : Visualizing some training images along with their labels

```
fig, ax = plt.subplots(nrows=3, ncols=6, figsize=(12,6))  
idxs = y_train.index  
for row in ax:  
    for col in row:  
        idx = random.choice(idxs)  
        col.axis('off')  
        col.set_title(y_train[idx])  
        col.imshow(x_train[idx], cmap=plt.get_cmap('gray'))
```

OUTPUT SCREENSHOT

```
In [2]: # Loading Data
df = pd.read_csv("data.csv")
df.head()

Out[2]:
```

	pixel_0000	pixel_0001	pixel_0002	pixel_0003	pixel_0004	pixel_0005	pixel_0006	pixel_0007	pixel_0008	pixel_0009	...	pixel_1015	pixel_1016	pixel_1017
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0

5 rows × 1025 columns

```
In [4]: # All output classes
all_chars = df['character'].unique()

print(f'All characters: {all_chars}')
print(f'Number of characters: {len(all_chars)}')

All characters: ['character_01_ka' 'character_02_kha' 'character_03_ga' 'character_04_gha'
'character_05_kna' 'character_06_cha' 'character_07_chha'
'character_08_ja' 'character_09_jha' 'character_10_yna'
'character_11_taamata' 'character_12_thaa' 'character_13_daa'
'character_14_dhaa' 'character_15_adna' 'character_16_taba'
'character_17_tha' 'character_18_da' 'character_19_dha' 'character_20_na'
'character_21_pa' 'character_22_pha' 'character_23_ba' 'character_24_bha'
'character_25_ma' 'character_26_yaw' 'character_27_ra' 'character_28_la'
'character_29_waw' 'character_30_motosaw' 'character_31_petchiryakha'
'character_32_patalosaw' 'character_33_ha' 'character_34_chhya'
'character_35_tra' 'character_36_gya' 'digit_0' 'digit_1' 'digit_2'
'digit_3' 'digit_4' 'digit_5' 'digit_6' 'digit_7' 'digit_8' 'digit_9']
Number of characters: 46
```



```
In [8]: # label encoding the output classes
labelencoder = LabelEncoder()

y_train = labelencoder.fit_transform(y_train)
X_train = x_train.reshape(x_train.shape[0], x_train.shape[1], x_train.shape[2], 1)

y_test = labelencoder.fit_transform(y_test)
X_test = x_test.reshape(x_test.shape[0], x_test.shape[1], x_test.shape[2], 1)
```

```
In [9]: # Configuring TPU if available
try:
    tpu = tf.distribute.cluster_resolver.TPUClusterResolver()
except ValueError:
    tpu = None
```

```
In [10]: # Select appropriate distribution strategy
if tpu:
    tf.config.experimental_connect_to_cluster(tpu)
    tf.tpu.experimental.initialize_tpu_system(tpu)
    strategy = tf.distribute.experimental.TPUStrategy(tpu)
    print('Running on TPU ', tpu.cluster_spec().as_dict()['worker'])
else:
    strategy = tf.distribute.get_strategy() # Default strategy that works on CPU and Single GPU
    print('Running on CPU instead')

print("Number of accelerators: ", strategy.num_replicas_in_sync)

Running on CPU instead
Number of accelerators: 1
```

```
In [14]: # Model with CNN
momentum = .9

cnn_model = models.Sequential()
cnn_model.add(Conv2D(64, (3,3), input_shape=(32, 32, 1), padding='same', activation='relu'))

cnn_model.add(Conv2D(64, (3,3), padding='same', activation='relu'))
cnn_model.add(BatchNormalization(momentum=momentum))

cnn_model.add(MaxPool2D())

cnn_model.add(Conv2D(128, (3,3), padding='same', activation='relu'))
cnn_model.add(Conv2D(128, (3,3), padding='same', activation='relu'))
cnn_model.add(BatchNormalization(momentum=momentum))

cnn_model.add(MaxPool2D())

cnn_model.add(Conv2D(256, (3,3), padding='same', activation='relu'))
cnn_model.add(Conv2D(256, (3,3), padding='same', activation='relu'))
cnn_model.add(BatchNormalization(momentum=momentum))

cnn_model.add(MaxPool2D())

cnn_model.add(Conv2D(512, (3,3), padding='same', activation='relu'))
cnn_model.add(Conv2D(512, (3,3), padding='same', activation='relu'))
cnn_model.add(BatchNormalization(momentum=momentum))

# flatten
cnn_model.add(GlobalMaxPool2D())

cnn_model.compile(optimizer='adam', loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True), metrics=['accuracy'])

# Display the architecture
cnn_model.summary()

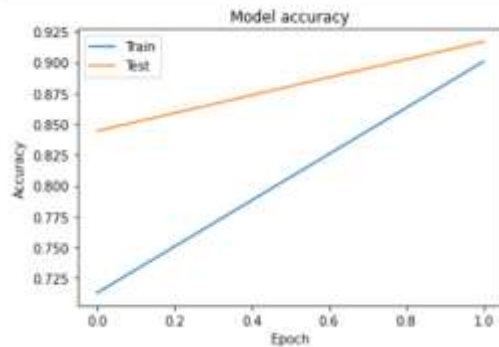
history = cnn_model.fit(X_train, y_train, epochs=2, validation_split=0.2, batch_size=400)
```

conv2d_14 (Conv2D)	(None, 4, 4, 512)	1180160
conv2d_15 (Conv2D)	(None, 4, 4, 512)	2359008
batch_normalization_7 (Batch Normalization)	(None, 4, 4, 512)	2048
global_max_pooling2d_1 (Global Max Pooling2D)	(None, 512)	0

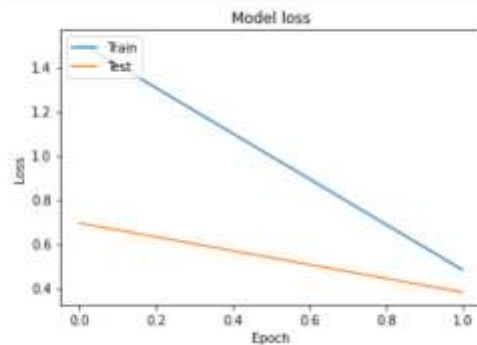
Total params: 4,688,064		
Trainable params: 4,686,144		
Non-trainable params: 1,920		

Epoch 1/2		
148/148 [-----] - 556s 4s/step - loss: 1.5105 - accuracy: 0.7131 - val_loss: 0.8936 - val_accuracy: 0.8444		
Epoch 2/2		
148/148 [-----] - 577s 4s/step - loss: 0.4818 - accuracy: 0.9006 - val_loss: 0.3809 - val_accuracy: 0.9168		

```
In [16]: #Plot of Model accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```



```
In [17]: # Plot training & validation Loss values
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```



7. CONCLUSION AND FUTURE ENCHANTMENT :

We created prediction model using Convolutional Neural Networks that have an accuracy of above 93.4% which predicts if the given character is in Hindi . We intend to create our own dataset, which will be updated as new information becomes available in the future.