

# IDENTIFYING HANDWRITTEN LETTER USING CONVOLUTIONAL NEURAL NETWORK

By

K. PAVAN SAI KIRAN ( Reg.No - 40130102 )

G.TEJA ( Reg.No – 40130072 )



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING  
SCHOOL OF ELECTRICAL AND ELECTRONICS

## 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 - 600119

OCTOBER - 2023



# **SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

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

[www.sathyabama.ac.in](http://www.sathyabama.ac.in)

## **DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

### **BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of K.PAVAN SAI KIRAN (40130102) and G.TEJA (40130072) who carried out the project entitled "IDENTIFYING HANDWRITTEN LETTERS USING CONVOLUTIONAL NEURAL NETWORK" under our supervision from November 2023 to April 2024.

**Internal Guide**

**Dr. M.R. EBENEZAR JEBARANI, ME., Ph.D.**

**Professor**

**Head of the Department**

**Dr. T. RAVI, M.E., Ph.D.**

**Professor**

Submitted for Viva voce Examination held on

17/4/24

**Internal Examiner**

**External Examiner**

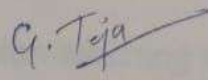
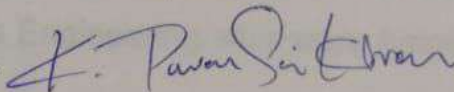
## DECLARATION

We, K.PAVAN SAI KIRAN and G.TEJA hereby declare that the project Report entitled "IDENTIFYING HANDWRITTEN LETTERS USING CONVOLUTIONAL NEURAL NETWORK" was done by us under the guidance of Dr. M.R. EBENEZAR JEBARANI, Professor, Dept. of Electronics and Communication Engineering, SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY, CHENNAI, is submitted in partial fulfillment of the requirements for the award of a Bachelor of Engineering degree in Electronics and Communication Engineering.

DATE: 17/04/2024

PLACE: Chennai

SIGNATURE OF THE CANDIDATES

1. 
2. 

## ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to **the Board of Management of SATHYABAMA** for their kind encouragement in doing this **project and** completing it successfully. We are grateful to them.

We convey our thanks to **Dr. N.M.Nanditha M.E., Ph. D., Prof & Dean,** School of Electrical and Electronics, **Dr. T. Ravi, M.E., Ph.D., Prof & Head of the Department** of Electronics and Communication Engineering for providing us with necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr.Ebenezar Jebarani, M.E, Ph.D,Professor,** for her valuable guidance, suggestions, and constant encouragement that paved the way for the successful completion of our phase-1 project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the **Department of Electronics and Communication Engineering** who were helpful in many ways for the completion of the project.

## ABSTRACT

Two crucial technological areas that are uniting the world are machine translation and natural language processing. The barrier to communication with others who speak a different language is removed via machine translation. Finding the word's native language and then translating it into the target language are the foundations of machine translation. If each letter in the word can be recognized, determining the language of the word may be simpler. We now arrive at "Character Recognition." This thesis focuses on evaluating the available techniques for handwritten character identification using deep learning and those techniques to recognize handwritten characters. The intriguing thing about machine learning is that the computers that are programmed to interpret, say, Chinese, can also be programmed to translate, say, Hindi or Telugu. This is a key characteristic that the thesis seeks to investigate because it is so potent. Beyond just detecting the characters, modern technology can also translate such characters between different languages. Depending on the character, the deep learning model created for this thesis can accurately identify the character or alphabets with a 70–90% rate of success. All data is saved and documented in datasets for further sharing and project usage. A large database of handwritten numbers is needed to train different methods for analyzing images the MNIST database, which is frequently used. The database is commonly used for developing and evaluating machine learning algorithms. We used our dataset collected from available online resources and implemented our model using CNN algorithm. We applied different data pre-processing techniques like thresholding and normalization in order to filter the data. We applied all the layers such as softmax, ReLU of CNN in the process of training our model.

## TABLE OF CONTENTS

| Chapter<br>No. | TITLE                                                     | Pg. No |
|----------------|-----------------------------------------------------------|--------|
|                | <b>LIST OF FIGURES</b>                                    | viii   |
|                | <b>LIST OF ABBREVIATIONS</b>                              | ix     |
|                | <b>INTRODUCTION</b>                                       |        |
|                | 1.1 GENERAL                                               | 1      |
| 1              | 1.2 PROBLEM STATEMENT                                     | 4      |
|                | 1.3 PROPOSED SOLUTION                                     | 4      |
|                | 1.4 PROGRAMMING LANGUAGE & LIBRARIES                      | 5      |
|                | <b>LITERATURE SURVEY</b>                                  |        |
|                | 2.1 LITERATURE REVIEW                                     | 8      |
| 2              | 2.2 OVERVIEW OF EXISTING RESEARCH                         | 9      |
|                | 2.3 SUMMARY OF CONCEPTS & THEORIES ABOUT THE PROJECT      | 10     |
|                | 2.4 JUSTIFICATION OF PROJECT'S IMPORTANCE                 | 11     |
| 3              | <b>DATA COLLECTION &amp; DATA PRE-PROCESSING</b>          |        |
|                | 3.1 DATA COLLECTION PROCESSING                            | 12     |
|                | 3.2 DATA PRE-PROCESSING METHODS                           | 14     |
| 4              | <b>SYSTEM DESIGN</b>                                      |        |
|                | 4.1 GENERAL                                               | 18     |
|                | 4.2 DESCRIPTION OF PROPOSED MODEL                         | 19     |
|                | 4.3 DETAILING HYPER PARAMETERS & OPTIMIZING THE ALGORITHM | 21     |
|                | 4.4 DISCUSSING TRAINING PROCESS & EVALUATION METRICS      | 23     |

# CHAPTER 1

## INTRODUCTION

### 1.1 GENERAL

Intelligent image analysis is an appealing research area in Artificial Intelligence and crucial for a variety of present open research difficulties. Handwritten digits recognition is a well-researched subarea within the field that is concerned with learning models to distinguish pre-segmented handwritten digits. It is one of the most important issues in data mining, machine learning, pattern recognition along with many other disciplines of artificial intelligence. The main application of machine learning methods over the last decade has determined efficacious in conforming decisive systems that are competing to human performance and which accomplish far improved than manually written classical artificial intelligence systems used in the beginnings of optical character recognition technology. However, not all features of those specific models have been previously inspected. A great attempt of research worker in machine learning and data mining has been contrived to achieve efficient approaches for approximation of recognition from data. In twenty first Century handwritten digit communication has its own standard and most of the times in daily life are being used as means of conversation and recording the information to be shared with individuals. One of the challenges in handwritten characters recognition wholly lies in the variation and distortion of handwritten character set because distinct community may use diverse style of handwriting, and control to draw the similar pattern of the characters of their recognized script. To locate such regions different kind of region sampling techniques are used in pattern recognition. Hence, robust feature extraction is very important to improve the performance of a handwritten character recognition system.

Nowadays handwritten digit recognition has obtained lot of concentration around pattern recognition system sowing to its application in diverse fields. In next days, character recognition system might serve as a cornerstone to initiate paperless surroundings by digitizing and processing existing paper documents. Handwritten digit dataset is vague in nature because there may not always be sharp and perfectly straight lines. The main goal in digit recognition is feature extraction is to remove the redundancy from the data and gain a more effective embodiment of the word image through a set of numerical attributes. It deals with extracting most of the essential information from image raw data. In addition, the curves are not necessarily smooth like the printed characters. Furthermore, characters dataset can be drawn in different sizes and the orientation which are always supposed to be written on a guideline in an upright or downright point. Accordingly, an efficient handwritten recognition system can be developed by considering these limitations. It is quiet exhausting that sometimes to identify handwritten characters as most of the human beings can't even recognize their own written scripts. Hence, there exists constraint for a writer to write apparently for recognition of handwritten documents. Before revealing the method used in conducting this research, software engineering module is first presented. Pattern recognition along with Image processing plays compelling role around handwritten character recognition. The study describes numerous types of classification of feature extraction techniques like structural feature-based methods, statistical feature-based methods, and global transformation techniques. Statistical approaches are established on planning of how data are selected. It utilizes the information of the statistical distribution of pixels in the image. The paper provided SVM based offline handwritten digit recognition system. Authors claim that SVM outperforms in the experiment. Experiment is carried out on NIST SD19 standard dataset. The study provides the conversion of handwritten data into electronic data, nature of handwritten characters and the neural network approach to form machine competent of recognizing handwritten characters. The study addresses a comprehensive criterion of handwritten digit recognition with various state of the art approaches, feature representations, and datasets. However, the relationship of training set size versus accuracy/error and the dataset-independence of the trained models are analyzed. The paper presents convolution neural networks into the handwritten digit recognition research and describes a system to find any digit.

Handwriting recognition of characters has been around since the 1980s. The task of handwritten digit 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 (for example - tax forms) and so on. There are different challenges faced while attempting to solve this problem. The handwritten digits are not always of the same size, thickness, or orientation and position relative to the margins. My goal was to implement a pattern classification method to recognize the handwritten digits provided in the MNIST data set of images of handwritten digits (0-9). The data set used for our application is composed of 300 training images and 300 testing images and is a subset of the MNIST data set [1] (originally composed of 60,000 training images and 10,000 testing images). Each image is a 28 x 28 grayscale (0-255) labeled representation of an individual digit. The general problem we predicted we would face in this digit classification problem was the similarity between the letters.

Researchers have been investigating various methods for testing handwritten character recognition over the past few decades. Learning becomes harder as our assignments become more challenging. We employ deep neural networks, which is suitable. Make a neural network design with higher performance metrics and train it. Deep learning and neural networks enable us to train the model we independently constructed to learn for itself from our experiences.

Convolutional Neural Network is one of the methods of Deep Learning Algorithms. Characters present in the image are successfully recognized by the CNN classifier. Convolutional layers for feature extraction and fully linked layers followed by a softmax layer for classification make up the architecture of conventional CNN classifiers. One of the effective feature extractors is CNN. Data exploration is a methodology that integrates methods from machine learning, statistics, and computer systems to find patterns in enormous volumes of data. The neurons in CNN's architecture are arranged in layers. This design consists of an input layer, many hidden levels, and an output unit. Typically, networks with a lot of hidden layers are referred to as deep neural networks. The neurons in the hidden layers of CNN are connected to only a fraction of the input space produced by the preceding layer, as opposed to entirely important factor like Multi Laminated Perceptron (MLP)

networks. The inversion process produces an extracted feature when the convolution kernel passes across the input matrix for the layer, adding to the input of the following layer. Then, further layers like normalizing, pooling, and entirely linked levels are introduced. In conclusion, handwritten digit recognition is a significant, quickly developing topic with a wide range of real-world uses. We may anticipate significant advancements in precision and performance as deep learning and computer vision technologies keep evolving, making them even more beneficial in a variety of fields.

## **1.2 PROBLEM STATEMENT**

Handwritten letter recognition is the task of correctly identifying handwritten letters from a given dataset. Deep learning is a promising approach for solving this problem, as it has been shown to achieve state-of-the-art results on a variety of image recognition tasks. The problem statement for handwritten letter recognition using deep learning would involve developing a model that can accurately classify images of handwritten letters into their corresponding alphabets (A-Z). The model should be able to handle variability in writing styles, rotations, and scaling of the letters, while also generalizing well to new, unseen data. The goal is to achieve high accuracy on a test set of images, while also minimizing the model's complexity and training time. Additionally, the model should be able to distinguish between uppercase and lowercase letters, and handle recognition of both printed and cursive handwriting styles.

## **1.3 PROPOSED SOLUTION**

There are several drawbacks of using traditional machine learning algorithms for handwritten letters recognition. Traditional machine learning algorithms require manual feature engineering, where domain experts manually select and engineer features that are relevant to the problem at hand. This can be time-consuming and often requires a deep understanding of the problem domain. These algorithms may struggle to handle variability in writing styles, rotations, and scaling of the letters, making it difficult to achieve high accuracy on a wide range of inputs. These algorithms may not generalize well to new, unseen data, especially if the data is significantly different from the training data. Many of the machine learning algorithms are often sensitive to noise in the data, which can negatively impact their accuracy

and performance. Traditional machine learning algorithms can struggle to handle large datasets, as the training process can be time-consuming and computationally expensive.

Since some deep learning approaches, such as convolutional neural networks (CNNs), have shown to overcome many of these drawbacks and achieve state-of-the-art results on handwritten letters recognition tasks, we are developing the deep learning model using CNN. We here are not using MNIST dataset ,since we feel it has not much more data and many of the algorithms are trained on this dataset. We have collected data from online resources we found and this project focuses not only on algorithm but also data processing which is the most important factor in any of the machine learning models.

## **1.4 PROGRAMMING LANGUAGE & LIBRARIES**

Python is a widely used general-purpose, high level programming language. It was created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code. Python is a programming language that lets you work quickly and integrate systems more efficiently.

### *1.4.1 Features of Python*

Python is a popular language for machine learning due to its simplicity, flexibility, and extensive libraries such as Scikit-learn, Tensorflow, and PyTorch, which provide powerful tools for data processing, modeling, and visualization.

#### *1.4.1.1 Interpreted*

There are no separate compilation and execution steps like C and C++.Directly run the program from the source code. Internally, Python converts the source code into an intermediate form called bytecodes which is then translated into native language of specific computer to run it. No need to worry about linking and loading with libraries, etc.

#### *1.4.1.2 Platform Independent*

Python programs can be developed and executed on multiple operating system platforms like Linux, Windows, Mac etc.

#### *1.4.1.3 High level Language*

In Python, no need to take care about low-level details such as managing the memory used by the program.

#### *1.4.1.3 Simple*

Closer to English language and easy to Learn. More emphasis on the solution to the problem rather than the syntax.

#### *1.4.1.4 Embeddable*

Python can be used within C/C++ program to give scripting capabilities for the program's users.

#### *1.4.1.5 Rich Library Support*

The Python Standard Library is very vast. Known as the “batteries included” philosophy of Python. It can help do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, email, XML, HTML, WAV files, cryptography, GUI and many more. Besides the standard library, there are various other high-quality libraries such as the Python Imaging Library which is an amazingly simple image manipulation library.

#### *1.4.1.6 Advantages*

- Presence of third-party modules.
- Extensive support libraries (NumPy for numerical calculations, Pandas for data analytics etc).
- Open source and community development.
- Versatile, Easy to read, learn and write.
- User-friendly data structures
- High-level language.
- Object-oriented language
- Portable and Interactive
- Ideal for prototypes – provide more functionality with less coding.

- (IoT)Internet of Things Opportunities.
- Interpreted Language.
- Portable across Operating systems

#### 1.4.1.7 Applications

- GUI based desktop applications.
- Graphic design, image processing applications, Games, and Scientific/computational Applications
- Web frameworks and applications
- Enterprise and Business applications
- Operating Systems
- Education
- Database Access
- Language Development
- Prototyping

In conclusion, python is widely used in the field of machine learning (ML) due to its simplicity, flexibility, and powerful libraries. ML frameworks such as TensorFlow, Keras, and PyTorch are written in Python and provide a high-level interface for building and training ML models. Python's extensive scientific computing libraries such as NumPy, Pandas, and SciPy make it easy to manipulate and analyze data, a key component of ML. Python's popularity and large community support have led to the development of many specialized ML libraries and tools, including scikit-learn, XGBoost, and NLTK. Moreover, Python's ability to integrate with other technologies and languages makes it a powerful tool for building end-to-end ML solutions. Overall, Python's importance in ML cannot be overstated, as it has helped make the development and deployment of ML models accessible to a wider audience of developers and researchers.

## CHAPTER 2

### LITERATURE SURVEY

#### 2.1 GENERAL

*2.1.1-Title:* Recognition of Handwritten Digit using Convolutional Neural Network in Python with Tensor flow and Comparison of Performance for Various Hidden Layers.[3]

Author: Fathima Siddique; Shadman Sakib

In this paper, to find the performance of CNN hidden layers, CNN as well as MNIST dataset real time input. have used. The network is trained using stochastic gradient descent and the backpropagation algorithm.

Scope of this paper is to adding feature like GUI to take real time input.

*2.1.2-Title:* Analogizing time Complexity of KNN and CNN in Recognizing Handwritten Digits.[9]

Author: Tanya makkar; Ashwani kumar Dubey

In this paper comparison of accuracy of KNN as well as CNN have done. In this paper MNIST data set has used for providing samples to the system.

Scope of this paper is to choosing CNN because it produces high accuracy then KNN and adding GUI for real time input.

*2.1.3-Title:* Telugu handwritten character recognition using adaptive and static zoning methods. [10]

Author: Sangula Durga Prasad; Yashwanth Kanduri

In this paper they presented two methods for this purpose. First method is based on Genetic Algorithm and uses adaptive zoning topology with extracted geometric features. In second method, zoning is done in static way and uses distance, density based features. In both the contexts, they used K-Nearest Neighbor (KNN) algorithm for classification purpose.

## **2.2 OVERVIEW OF EXISTING RESEARCH**

Handwritten letter recognition is a rapidly evolving field with a long history of research. The study involves the development of computer algorithms that can automatically identify and interpret handwritten characters. The primary challenge in this field is the high variability in handwriting styles, which makes it difficult for algorithms to distinguish between different characters accurately.

Early approaches to handwritten letter recognition were rule-based or template matching techniques. However, these methods were limited by their inability to handle variability in handwriting. In recent years, deep learning techniques have shown significant promise in improving the accuracy of handwritten letter recognition. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have been particularly successful in this regard.

One of the major issues in handwritten letter recognition is the lack of high-quality training data. While there are several datasets available for this purpose, they are often small and may not represent the variability in handwriting styles. Several researchers have attempted to address this issue by creating synthetic datasets or using generative adversarial networks (GANs) to generate new training data.

Another area of research in handwritten letter recognition is the development of new features and preprocessing techniques that can improve the accuracy of recognition algorithms. For example, some researchers have investigated the use of stroke-based features, which capture the structure and direction of each stroke in a handwritten character.

There have also been efforts to develop new evaluation metrics for assessing the performance of handwritten letter recognition algorithms. One such metric is the writer-independent error rate (WER), which measures the error rate when recognizing characters written by individuals who are not in the training set.

Despite the progress made in this field, there are still several challenges that need to be addressed. For example, the recognition of cursive handwriting is still a

major challenge for algorithms. Additionally, there is a need for more robust algorithms that can handle noise and other forms of degradation in handwritten documents. In conclusion, handwritten letter recognition is a field that has seen significant advancements in recent years, with the introduction of deep learning techniques and the development of new evaluation metrics and preprocessing techniques. However, there are still several challenges that need to be addressed before this technology can be widely deployed in practical applications.

## **2.3 SUMMARY OF CONCEPTS & THEORIES ABOUT THE PROJECT**

Firstly I examined a research paper regarding the Handwritten letters recognition. Handwritten Character Recognition using Star-Layered Histogram Features Stephen Karungaru, Kenji Terada and Minoru Fukumi, In this method, a character recognition method using features extracted from a star layered histogram is presented and trained using neural networks. After several image preprocessing steps, the character region is extracted. Its contour is then used to determine the center of gravity (COG). This CoG point is used as the origin to create a histogram using equally spaced lines extending from the CoG to the contour. The first point the line touches the character represents the first layer of the histogram. If the line extension has not reached the region boundary, the next hit represents the second layer of the histogram. This process is repeated until the line touches the boundary of the character's region. After normalization, these features are used to train a neural network.

Throughout the project we implemented the model using CNN. The core concept of CNNs is the use of convolutional layers that learn and detect local features from the input images. In CNNs, the input image is processed through a series of convolutional and pooling layers. The convolutional layers apply a set of filters to the input image to learn and detect features such as edges, corners, and textures. The pooling layers then down sample the feature maps generated by the convolutional layers, reducing their spatial size while retaining important information. This process is repeated multiple times, resulting in a hierarchy of features that represent more complex and abstract information about the input image.

In addition to convolutional and pooling layers, CNNs also typically include fully connected layers that learn to classify the input image based on the learned features. The final layer of the network typically outputs a probability distribution over the possible classes, allowing the network to make predictions about the input image. CNNs have demonstrated state-of-the-art performance in a wide range of computer vision tasks, due in part to their ability to learn and detect relevant features from the input images. The use of GPUs and other hardware accelerators has also enabled the efficient training of large-scale CNNs on massive datasets.

## **2.4 JUSTIFICATION OF PROJECT'S IMPORTANCE**

Handwritten recognition using CNN models is needed because of the widespread use of handwritten documents and the need for efficient and accurate automated processing of these documents. Handwritten text is still widely used in various domains such as banking, legal, education, and healthcare, and the ability to automatically recognize and transcribe this text can improve the efficiency and accuracy of many processes.

Traditional methods of handwritten recognition involve manually designing and extracting features from the handwritten images, which is a time-consuming and error-prone process. CNN models, on the other hand, can automatically learn and extract relevant features from the input images, eliminating the need for manual feature engineering. Moreover, CNN models have shown significant improvements in the accuracy of handwritten recognition compared to traditional methods. CNN models can handle the variability in handwriting styles and can recognize characters even with noise, distortion, and other forms of degradation.

In summary, implementing handwritten recognition using CNN models is needed to improve the accuracy and efficiency of automated processing of handwritten documents, eliminate the need for manual feature engineering, and enable various applications in different domains.

## CHAPTER 3

### DATA COLLECTION & PRE-PROCESSING

#### 3.1 DATA COLLECTION PROCESS

Data collection is a crucial step in the machine learning process as it provides the necessary information for training and testing models. The data collection process involves the following steps:

- Determine the problem statement: Define the problem to be solved and identify the relevant data needed to solve it.
- Identify the data sources: Identify potential sources of data, such as existing datasets, APIs, or manual collection methods.
- Collect the data: Collect data from the identified sources, either manually or through automated means.
- Preprocess the data: Clean the data by removing irrelevant or duplicated data, filling in missing values, and transforming the data into a suitable format for analysis.
- Label the data: Label the data with relevant categories or outcomes, such as class labels or numerical values.
- Split the data: Split the data into training, validation, and test sets to enable model development and evaluation.
- Store the data: Store the data in a suitable format, such as CSV, JSON, or database format, for easy access and analysis.
- Ensure data privacy and security: Ensure that the collected data is secure and that the privacy of individuals is maintained by anonymizing the data or obtaining consent where necessary.
- Effective data collection is critical for the success of machine learning models, as it enables the models to learn from real-world data and make accurate predictions or classifications.

Datasets used for handwritten letters recognition typically consist of a large number of handwritten letter images, along with corresponding labels that identify the correct letter for each image. Some of the commonly used datasets for handwritten letters recognition are:

- MNIST: This is a widely used dataset that consists of 70,000 images of handwritten digits from 0 to 9. Each image is a 28x28 grayscale image.
- EMNIST: This dataset is an extension of MNIST and consists of 240,000 images of handwritten letters and digits. The letters are from A to Z, both uppercase and lowercase, and the digits are from 0 to 9.
- NIST: This dataset consists of images of handwritten letters and digits collected by the National Institute of Standards and Technology (NIST) for research purposes.
- CEDAR: The CEDAR dataset consists of over 6,000 images of handwritten English letters, including both uppercase and lowercase letters.
- IAM: The IAM dataset consists of handwritten English words and sentences, including both cursive and printed writing styles.
- Chars74K: This dataset consists of 74,000 images of handwritten characters from various languages, including English, Arabic, and Chinese.

These datasets are widely used for training and testing machine learning models for handwritten letters recognition. They provide a diverse set of images with varying writing styles, noise, and distortions, enabling the development of robust models that can handle real-world scenarios. Since, all the above mentioned datasets are used by many in their previous works we implemented our dataset using references from them and contains more than 3 lakh images which we thought is really a good count for making the model more accurate.

Data preprocessing refers to the cleaning, transformation, and preparation of raw data before it is used for machine learning applications. The main goal of data preprocessing is to improve the quality and usability of the data by removing errors, filling in missing values, and transforming the data into a format that is suitable for analysis. Data preprocessing is an important step in machine learning because the quality and accuracy of the resulting model are directly influenced by the quality of the input data. Poor quality data can lead to inaccurate or unreliable predictions, and in some cases, may even render the model unusable.

## **3.2 DATA PRE-PROCESSING**

In machine learning, training data and testing data are two key components used to develop and evaluate machine learning models.

Training data refers to a set of input data and corresponding output labels that are used to train a machine learning model. During the training process, the model learns the underlying patterns and relationships between the input data and output labels. The goal of training is to optimize the model's parameters or weights, so it can accurately predict the correct output label for new, unseen data.

Testing data, on the other hand, is a set of input data and output labels that are used to evaluate the performance of a trained machine learning model. The testing data is separate from the training data, and the model has not seen these data points during the training process. The goal of testing is to determine how well the model generalizes to new, unseen data.

To ensure that the model performs well on new data, the testing data should be representative of the real-world data that the model will encounter. The testing data should also be large enough to provide a reliable estimate of the model's performance. In order to prevent overfitting, it's important to use separate sets of training and testing data. Overfitting occurs when a model learns the training data too well and becomes too specific to that data. This can lead to poor performance on new, unseen data. By using separate sets of training and testing data, we can evaluate the model's ability to generalize to new data and prevent overfitting. In summary, training data is used to teach the model how to make accurate predictions, while testing data is used to evaluate the model's performance and ensure that it generalizes well to new data.

### **3.2.1 Thresholding**

Thresholding is a common technique used in image processing and computer vision, including in handwritten letters recognition, to segment an image into foreground and background. The basic idea of thresholding is to convert a grayscale image into a binary image by setting a threshold value, and any pixel value above that threshold is considered part of the foreground, and any value below the threshold is considered part of the background.

### 4.3 DETAILING HYPER PARAMETERS & OPTIMIZING THE ALGORITHM

Hyperparameters in a convolutional neural network (CNN) are values that are set before training the model and can have a significant impact on the performance of the model. Here are some common hyperparameters in a CNN:

- **Number of filters:** This is the number of feature maps used in each convolutional layer. More filters can capture more complex features but also require more computational resources.
- **Filter/kernel size:** This is the size of the filter/kernel used in each convolutional layer. Larger filters can capture larger features but can also lead to more computation.
- **Stride:** This is the number of pixels by which the filter/kernel is moved across the input image at each step. Larger strides reduce the output size and computational cost but may lead to information loss.
- **Padding:** This is the number of pixels added to the input image to preserve its size during convolution. Padding can help to avoid information loss and ensure that the output size is the same as the input size.
- **Pooling:** This is the operation of down-sampling the output of a convolutional layer. Common pooling methods are max pooling and average pooling. Pooling can help to reduce the spatial dimension of the feature maps and can help to avoid overfitting.
- **Learning rate:** This is the step size used to update the weights of the neural network during backpropagation. It controls how much the weights are adjusted during training.
- **Batch size:** This is the number of training examples used in each iteration of training. A larger batch size can reduce the variance of the gradient estimate but can also require more memory.
- **Dropout rate:** This is the probability of dropping out a neuron during training. Dropout can help to prevent overfitting by randomly dropping out neurons and forcing the network to learn more robust features.

These are just a few examples of the hyperparameters that can be tuned in a CNN. The choice of hyperparameters will depend on the specific problem and dataset.

In the context of handwritten letters recognition, thresholding can be used to separate the handwritten letters from the background noise, making it easier to extract features and classify the letters. For example, after applying thresholding to an image of a handwritten letter, we can count the number of pixels in the foreground to get an estimate of the letter's size, or we can compute the horizontal and vertical projections of the letter to obtain information about its shape and orientation. There are several methods for thresholding images, including global thresholding, adaptive thresholding, and Otsu's method. Global thresholding uses a fixed threshold value for the entire image, while adaptive thresholding adjusts the threshold value for different parts of the image based on local characteristics. Otsu's method is a widely used thresholding technique that automatically computes an optimal threshold value based on the distribution of pixel intensities in the image. Thresholding can be a useful preprocessing step in handwritten letters recognition, particularly when dealing with noisy or complex images. However, the effectiveness of thresholding depends on the quality of the input image and the specific characteristics of the handwritten letters being analyzed. For our project the function `threshold()` from `cv2` library is used for this preprocessing technique.

### **3.2.2 One-hot Encoding**

Thresholding can be a useful preprocessing step in handwritten letters recognition, particularly when dealing with noisy or complex images. However, the effectiveness of thresholding depends on the quality of the input image and the specific characteristics of the handwritten letters being analyzed.

One hot encoding is commonly used in machine learning algorithms that require numeric input, such as neural networks and decision trees. It allows us to represent categorical variables as numerical data, making it easier for algorithms to process the data and learn relationships between the variables. However, one hot encoding can result in high-dimensional data, particularly for variables with many categories. In such cases, it may be necessary to reduce the dimensionality of the data through techniques such as feature selection or dimensionality reduction. In this project, we will use `to_categorical()` function imported from `keras` library.

### **3.2.3 Gaussian Blur**

Gaussian blur is a commonly used technique in deep learning for image processing, particularly in computer vision applications. The basic idea of Gaussian blur is to smooth an image by convolving it with a Gaussian filter, which is a bell-shaped curve that assigns weights to each pixel in the image based on its distance from the center pixel.

The Gaussian blur operation is performed by sliding a kernel (i.e., the Gaussian filter) over the image and computing the weighted average of the pixel values within the kernel. The resulting smoothed image has reduced noise and sharp edges, which can be useful for improving the accuracy of object detection and recognition algorithms. In deep learning, Gaussian blur is often used as a preprocessing step for input images before they are fed into a neural network. This can help to reduce the impact of noise and other artifacts in the image and improve the overall performance of the network. Gaussian blur can also be used as a data augmentation technique in deep learning, where it is applied to randomly selected images in the training dataset. By adding random variations to the input data, data augmentation can help to prevent overfitting and improve the generalization ability of the network. Overall, Gaussian blur is a useful technique in deep learning for image processing and data augmentation, and can help to improve the accuracy and robustness of neural network models. By using the `GaussianBlur()` function from `cv2` library we successfully implemented this preprocessing step.

### **3.2.4 Normalization**

Normalization is a common technique used in machine learning to rescale input data to a standard range. The main idea behind normalization is to transform the input data so that it has a mean of zero and a standard deviation of one, or to scale it to a specific range of values, such as between 0 and 1 or -1 and 1.

The purpose of normalization is to ensure that the input data has consistent and comparable scales, which can help to improve the performance of machine learning algorithms. For example, if one feature has a much larger range of values than another feature, it can dominate the learning process and lead to biased or inaccurate results.

There are several common normalization techniques used in machine learning, includes Z-score normalization, Min-max normalization, Decimal scaling

normalization. Normalization can be applied to different types of data, including continuous and categorical variables, and can be used as a preprocessing step before training a machine learning model. However, it is important to note that normalization can also have drawbacks, such as increasing the computational complexity of the model and reducing the interpretability of the results.

## CHAPTER 4

### SYSTEM DESIGN

#### 4.1 ARCHITECTURE DIAGRAM

CNN works as backend and GUI works as frontend. We started with data collection, next pre-processed the set of handwritten letters. After that we split the data for training & testing. We then implemented the CNN model by applying softmax, ReLU ,pooling layers. We measured the performance of our model with the help of in-built accuracy & loss functions available. Last step is to test the model by passing the input data to it. We implemented a GUI for passing the input and displaying the result.

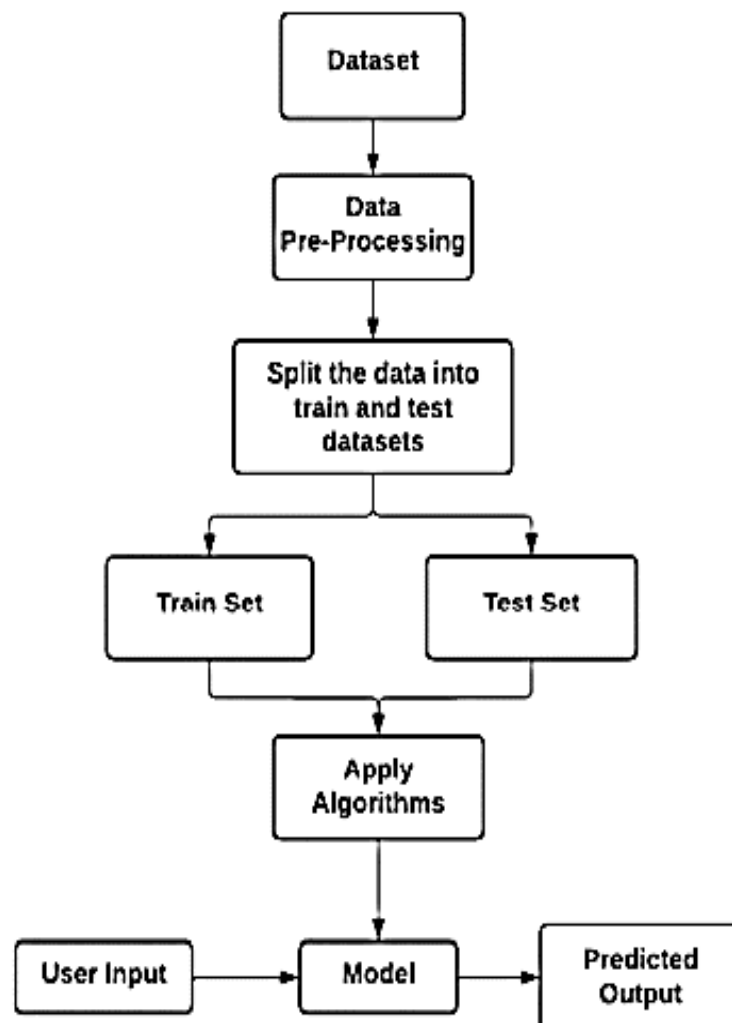


Fig 4.1 – Architecture diagram

## 4.2 DESCRIPTION OF PROPOSED MODEL

A Convolutional Neural Network (CNN) is a type of neural network that is commonly used for image processing and computer vision applications. The basic architecture of a CNN consists of multiple layers, including convolutional layers, pooling layers, and fully connected layers.

The convolutional layer is the core building block of a CNN, and it is responsible for learning feature maps that capture the patterns and structures in the input image. In this layer, a set of filters (or kernels) are applied to the input image to produce a set of feature maps. Each filter learns to detect a specific feature, such as edges, corners, or shapes, by convolving the filter with the input image and computing a dot product.

The pooling layer is used to downsample the feature maps and reduce the dimensionality of the input data. This layer typically uses a max pooling or average pooling operation to extract the most important features from each feature map and discard the rest.

The fully connected layer is responsible for classifying the input image based on the learned features. In this layer, the feature maps are flattened into a one-dimensional vector, and a series of fully connected neurons are used to predict the class label of the input image.

The overall training process of a CNN involves feeding a large dataset of labeled images into the network, adjusting the weights of the filters using backpropagation and gradient descent, and evaluating the performance of the model on a validation set. The main objective of the training process is to minimize the loss function, which measures the difference between the predicted output of the model and the true output. In addition to the basic architecture of a CNN, there are several advanced techniques that can be used to improve its performance, such as data augmentation, regularization, and transfer learning. Data augmentation involves applying random transformations to the input images, such as rotations, flips, and zooms, to increase the size and diversity of the training dataset. Regularization techniques, such as L1 and L2 regularization, are used to prevent overfitting and improve the generalization ability of the model. Transfer learning

involves using a pre-trained CNN model as a starting point for a new task, and fine-tuning the model on a smaller dataset to adapt it to the new task.

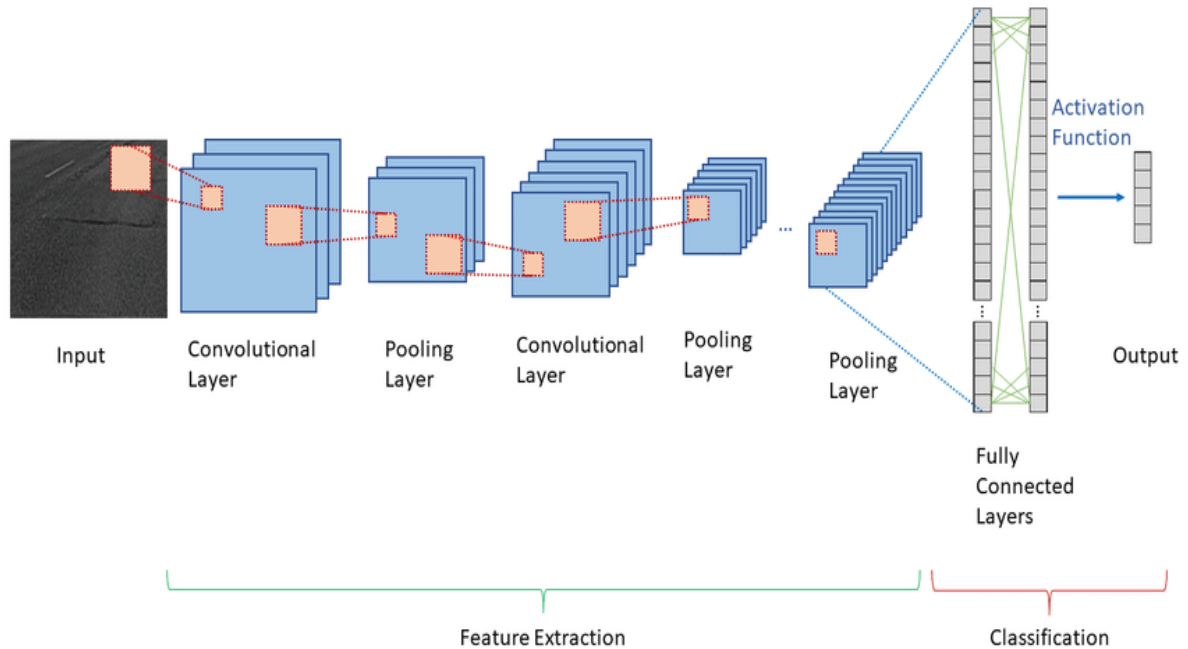


Fig 4.2 – CNN Architecture diagram

**Convolutional Layers:** The first layer in a CNN is typically a convolutional layer, which performs a convolution operation on the input image using a set of learnable filters. Each filter produces a feature map that highlights a particular aspect of the input image, such as edges or corners.

**Pooling Layers:** After each convolutional layer, a pooling layer is often used to downsample the feature maps and reduce their spatial size. This helps to reduce the number of parameters in the network and improve its computational efficiency.

**Activation Functions:** Nonlinear activation functions such as ReLU (Rectified Linear Unit) are applied to the output of each layer to introduce nonlinearity and enable the network to learn more complex patterns.

**Fully-Connected Layers:** The final layers in a CNN are typically fully-connected layers, which are used to map the high-level features learned by the previous layers to the output classes. These layers use a softmax activation function to produce a probability distribution over the possible classes.

Optimizing algorithms for a convolutional neural network (CNN) is an important task for achieving high performance. Here are some commonly used optimization algorithms for CNNs:

- Stochastic Gradient Descent (SGD): This is a widely used optimization algorithm for training neural networks. It updates the weights of the neural network in the direction of the negative gradient of the loss function with respect to the weights.
- Adam: This is a popular optimization algorithm that combines the advantages of both SGD and momentum. It uses adaptive learning rates for each parameter and adaptive momentum estimates to converge faster and more accurately.
- Adagrad: This is another optimization algorithm that adapts the learning rate to each parameter based on its historical gradient information. It is particularly effective for sparse data.
- RMSprop: This is an optimization algorithm that uses the moving average of squared gradients to adapt the learning rate. It is effective for handling non-stationary and noisy gradients.
- Adadelat: This is an optimization algorithm that uses the moving average of squared gradients and the moving average of squared parameter updates to adapt the learning rate. It has been shown to be particularly effective for large datasets and complex models.
- Nadam: This is an optimization algorithm that combines the advantages of both Adam and Nesterov momentum. It uses Nesterov momentum to compute the gradient and Adam to update the parameters.

The choice of optimization algorithm will depend on the specific problem and dataset. Generally, Adam is a popular choice due to its fast convergence and effectiveness on a wide range of problems. However, it is still important to experiment with different optimization algorithms to find the best one for the specific problem at hand.

#### 4.4 TRAINING PROCESS & EVALUATION METRICS

Training a convolutional neural network (CNN) involves a process of learning the weights of the model using a set of labeled training data. Here are the key steps involved in training a CNN:

- **Data preparation:** The first step is to prepare the training data. This involves splitting the data into training, validation, and test sets. The training set is used to update the weights of the model, the validation set is used to tune the hyperparameters of the model, and the test set is used to evaluate the final performance of the model.
- **Model architecture:** The next step is to choose the architecture of the CNN. This includes the number of convolutional layers, pooling layers, and fully connected layers. The architecture should be chosen based on the complexity of the problem and the size of the dataset.
- **Initialization:** The weights of the CNN are initialized randomly. It is important to use an initialization technique that allows for efficient learning of the model, such as the Glorot or He initialization.
- **Forward propagation:** During training, the input data is fed forward through the layers of the CNN, and a prediction is made for each example in the training set.
- **Loss calculation:** The difference between the predicted output and the true output is calculated using a loss function such as cross-entropy or mean squared error.
- **Backpropagation:** The error is propagated backward through the network, and the gradients of the loss function with respect to the weights of the network are calculated using the chain rule.
- **Weight update:** The weights of the network are updated using an optimization algorithm such as stochastic gradient descent (SGD), Adam, or Adagrad. The update rule adjusts the weights in the direction that minimizes the loss function.
- **Repeat:** Steps 4-7 are repeated until the weights of the model converge or a maximum number of epochs is reached.
- **Evaluation:** Once the weights are trained, the model is evaluated on the test set to obtain the final performance metrics.

It is important to note that training a CNN can be a time-consuming and computationally intensive task, especially for large datasets and complex models. Training on a GPU or using distributed training can significantly speed up the process. Additionally, careful tuning of hyperparameters and regularization techniques such as dropout and weight decay can improve the performance of the model.

The performance of the CNN for this task can be evaluated using several metrics. Here are some commonly used metrics for evaluating the performance of a CNN for handwritten letter recognition:

- **Accuracy:** This metric measures the proportion of correctly classified letters out of the total number of letters in the test set. It is the most commonly used metric for evaluating classification models.
- **Precision:** This metric measures the proportion of true positives (correctly identified letters) out of all the letters identified as positive by the model. Precision is a useful metric when the cost of false positives is high.
- **Recall:** This metric measures the proportion of true positives out of all the actual positive letters in the test set. Recall is a useful metric when the cost of false negatives is high.
- **F1 Score:** This is the harmonic mean of precision and recall, which combines the two metrics into a single value. It is useful for balancing precision and recall when both metrics are important.
- **Confusion matrix:** A confusion matrix is a table that summarizes the predicted and actual classes of the letters in the test set. It can be used to calculate the accuracy, precision, and recall, as well as to identify which classes are most frequently misclassified.
- **Receiver Operating Characteristic (ROC) curve:** The ROC curve plots the true positive rate (recall) against the false positive rate for different threshold values. It can be used to visualize the performance of the model across different classification thresholds.
- **Area Under the Curve (AUC):** This metric is calculated by integrating the ROC curve and measures the overall performance of the model across different threshold values.

The choice of metrics will depend on the specific requirements of the application. For example, if the goal is to minimize false positives, precision would be a more important metric than recall. Similarly, if the cost of false negatives is high, recall would be a more important metric. Overall, a combination of these metrics can provide a comprehensive evaluation of the performance of a CNN for handwritten letter recognition.

## REFERENCES

- Abu Ghosh, M. M., & Maghari, A. Y. (2017). A comparative study on handwriting digit recognition using neural networks. In International Conference on Promising Electronic Technologies (ICPET).
- Caifeng Shan, Shaogang Gong and Peter W. McOwan, "Facial expression recognition based on Localbinarypatterns: Acomprehensivestudy", ELSESEVIER Image and Vision Computing".
- F. Siddique, S. Sakib and M. A. B. Siddique, "Recognition of Handwritten Digit using Convolutional Neural Network in Python with Tensorflow and Comparison of Performance for Various Hidden Layers," 2019 5th International Conference on Advances in Electrical Engineering (ICAEE).
- Handwritten Character Recognition of South Indian Scripts: A Review by Jomy John, Pramod K. V, Kannan Balakrishnan, National Conference on Indian Language Computing.
- Handwritten Character Recognition for Telugu Scripts Using Multi - Layer Perceptrons (MLP). C. Vikram, C. Soba Bindhu, C. Sashikala.
- M. Yadav and R. K. Purwar, "Integrating Wavelet Coefficients and CNN for Recognizing Handwritten Characters," 2018 2nd IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES), Delhi, India, 2018, pp. 1160-1164.
- P. Voigtländer, P. Doetsch and H. Ney, "Handwriting Recognition with Large Multidimensional Long Short-Term Memory Recurrent Neural Networks," 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR), Shenzhen, China, 2016, pp. 228-233.
- Rudraswamimath, V.R., & Bhavanishankar, K. (2019). Handwritten digit recognition usingCNN. International Journal of Innovative Science and Research Technology, 4, 182–187.
- Shahid, A. R., & Talukder, S. Evaluating machine learning models for handwriting recognition-based systems under local differential privacy.
- T. Makkar, Y. Kumar, A. K. Dubey, Á. Rocha and A. Goyal, "Analogizing time complexity of KNN and CNN in recognizing handwritten digits," 2017 Fourth International Conference on Image Information Processing (ICIIP), Shimla, India, 2017.
- Telugu handwritten character recognition using adaptive and static zoning methods. Sanugula Durga Prasad, Yashwanth Kandhuri.

