A

PROJECT REPORT

ON

“**Application Of Artificial Intelligence in Handwriting Recognition**”

SUBMITTED

To

CENTRE FOR ONLINE LEARNING

Dr. D.Y. PATIL VIDYAPEETH, PUNE



IN PARTIAL FULFILMENT OF DEGREE OF

MASTER OF BUSINESS ADMINISTRATION

BY

**Rakesh Sambhaji Patil**

**PRN:** 23050202736

**BATCH** 2023-2025



**Dr. D.Y. Patil Vidyapeeth’s**

**CENTRE FOR ONLINE LEARNING,**

**Sant Tukaram Nagar, Pune.**

**CERTIFICATE**

This is to certify that Mr. Rakesh Sambhaji Patil

PRN - 23050202736

has completed his/her internship at Synechron Private Limited

starting from\_\_\_\_\_\_\_\_\_\_\_ to \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

His project work was a part of the MBA (ONLINE LEARNING)

The project is on Application Of Artificial Intelligence in Handwriting Recognition

Which includes research as well as industry practices. He was very sincere and committed in all tasks.

Course Coordinator

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date -

**COMPANY LETTER**

**(TO BE PROVIDED BY THE COMPANY WHERE THE PROJECT WILL BE CARRIED OUT)**

**To whomsoever it may concern**

This is to certify that Mr./Ms. Rakesh Sambhaji Patil

PRN - 23050202736

has completed his/her internship at Synechron Private Limited.

starting from March - 2024 to till now .

His / Her project work was a part of the MBA (ONLINE LEARNING)

The project is on Application Of Artificial Intelligence in Handwriting Recognition

Which includes research as well as industry practices. He/ She was very sincere and committed in all tasks.

Signature & Seal of Industry Guide

**DECLARATION BY LEARNER**

This is to declare that I have carried out this project work myself in part fulfillment of the M.B.A Program of Centre for Online Learning of Dr..D.Y.Patil Vidyapeeth’s, Pune – 411018

The work is original, has not been copied from anywhere else, and has not been submitted to any other University / Institute for an award of any degree / diploma.

Date: - Signature: -

Place: Pune Name: Rakesh Sambhaji Patil

**ACKNOWLEDGEMENT**

I would like to thanks D Y Patil University for providing me an opportunity to enhance my knowledge in the field of Artificial Intelligence & Machine Learning while working. It may have been remained as dream if not given a chance to upskill myself in the field of machine learning & artificial intelligence.

I am extremely grateful to both of my mentors and guides. Mr. Mahantesh Sir (DPY Guide) & Mr. Vivek Malkar (Industry Guide). Thanks for helping me to drive this journey.

I will be failing in duty if I do not acknowledge with grateful thanks to the authors of the references and other literatures referred in this Project.

I express my thanks to all staff members and friends for all the help and co-ordination extended in bringing out this project successfully in time.

Finally, I am very much thankful to my family who helped me to achieve this in time.

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Item** | **Page No** |
| 1 | Executive Summary | 7 - 12 |
| 2 | Chapter 1: Introduction, Objective and Purpose of Study | 13 - 16 |
| 3 | Chapter 2: Literature Review | 17 - 19 |
| 4 | Chapter 3: Research methodology | 20 - 24 |
| 5 | Chapter 4: Algorithms | 25 - 35 |
| 6 | Chapter 5: Data Analysis | 36 - 39 |
| 7 | Chapter 6: Coding Implementation | 40 - 55 |
| 8 | Chapter 7: Findings, suggestions, recommendation | 56 - 57 |
| 9 | Chapter 8: Conclusion | 58 |
| 10 | Scope for future study | 59 |
| 11 | Reference (Website, company paper) | 60 |
| 12 | Bibliography (Books, Journals, research work) |  |

**EXECUTIVE SUMMARY**

**Introduction to Handwriting Recognition:**

This project is very helpful in classifying handwritten words and characters and converting them into digital format. This technology bridges gap between human-computer interaction, transforming written documents into digital data that can be stored, searched and analyzed. Handwriting recognition plays vital role in diverse industries including finance, healthcare, education and government. Enabling automated data processing and increasing accessibility to historical and handwritten data.

Historically, handwriting recognition has been a challenging task due to the wide variations in handwriting styles, layouts and quality of handwritten documents. Recent advancements in AI & ML have significantly improved the accuracy and efficiency of handwritten recognition systems, making it a promising technology for both commercial and research applications.

I have used a concept of machine learning known as Recurrent Convolutional Neural Networks (RNN) to recognize and classify words and characters in a given image.

This model will help to enhance the basic tasks performed in various areas like

* Data entry can be simplified if we just provide the image as input to our model and it will provide result as a digital text
* Maintaining old records will be an easy task and fetching them would become more easier, as its just type away rather than roaming through bunch of files.

I will be training the model using a pre available IAM data so that its predicting capabilities will be near to real world.

**What is handwriting recognition? And how is handwriting recognition different from traditional OCR?**

Traditional OCR algorithms and techniques assume we’re working with a fixed font of some sort. In the early 1900s, that could have been the font used by microfilms.

In the 1970s, specialized fonts were developed specifically for OCR algorithms, thereby making them more accurate.

By the 2000s, we could use the fonts that came pre-installed on our computers to automatically generate training data and use these fonts to train our OCR models.

Each of these fonts had something in common:

They were engineered in some manner. There was a predictable and assumed space between each character (thereby making segmentation easier).

The styles of the fonts were more conducive to OCR.

Essentially, engineered/computer-generated fonts make OCR far easier.

Handwriting recognition is an entirely different beast though. Consider the extreme number of variations and how characters often overlap. Everyone has their own unique writing style.

Characters can be elongated, swooped, slanted, stylized, crunched, connected, tiny, gigantic, etc. (and come in any of these combinations).

Digitizing handwriting recognition is extremely challenging and is still far from solved but deep learning is helping us improve our handwriting recognition accuracy.

**Key Components and Types of Handwriting Recognition:**

Handwriting recognition can be broadly classified into two types:

1. Online Handwriting Recognition: This involves recognizing text written on a digital surface in real-time, using devices such as tablets or styluses. Online handwriting recognition captures temporal information, such as stroke order and direction, which enhances recognition accuracy.

2. Offline Handwriting Recognition: This involves recognizing text from scanned or photographed images of handwritten documents. It is more complex due to the lack of temporal data and the high variability in handwriting style.

**Key components of handwriting recognition include:**

* Preprocessing: Enhancing image quality by removing noise and normalizing text to improve the performance of recognition algorithms.
* Segmentation: Breaking down the handwritten text into individual characters or words to facilitate recognition.
* Feature Extraction: Identifying unique characteristics such as edges, shapes, and contours, which serve as inputs for recognition algorithms.
* Recognition: Using algorithms to map extracted features to corresponding characters or words.
* Post-processing: Refining the recognized output by correcting errors and improving accuracy through context and grammar checks.

**Machine Learning Approaches in Handwriting Recognition**

Modern handwriting recognition has seen rapid advancements due to developments in machine learning, particularly deep learning. The major approaches include:

**1. Traditional Machine Learning Techniques:** Early handwriting recognition systems relied on techniques like k-nearest neighbours (KNN), support vector machines (SVMs), and hidden Markov models (HMMs). These methods use hand-crafted features and statistical models to classify characters, with limited scalability and adaptability to varied handwriting styles.

**2. Deep Learning Models:** Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have revolutionized handwriting recognition due to their capability to learn features automatically from data, without manual intervention. CNNs are effective in extracting spatial features from images, while RNNs, especially Long Short-Term Memory (LSTM) networks, excel in handling sequential data and capturing context in character sequences.

**3. Hybrid Approaches:** Combining CNNs and RNNs has proven to be highly effective for handwriting recognition. A CNN extracts spatial features from the image, which are then passed to an RNN that models sequential dependencies. Hybrid architectures like the CNN-LSTM have achieved state-of-the-art results in handwriting recognition by effectively handling both spatial and temporal patterns.

**Current Applications and Use Cases**

**1. Digitization of Historical Documents:** Handwriting recognition is instrumental in digitizing archives, such as historical manuscripts and legal documents. Libraries and government agencies employ handwriting recognition to convert these records into searchable digital formats, preserving cultural heritage.

**2. Banking and Finance:** Automated check processing and verification systems rely on handwriting recognition to read handwritten account details, signatures, and check amounts, improving efficiency in transaction processing.

**3. Healthcare:** In medical settings, handwriting recognition helps digitize handwritten prescriptions and medical notes, reducing transcription errors, and making patient records easily accessible.

**4. Education:** Handwriting recognition assists in automatic grading of handwritten assignments, making it easier for educators to assess and record student performance.

**5. Personal Devices and Applications:** Mobile devices increasingly support handwriting recognition for tasks like notetaking, text entry, and personal reminders, enhancing user interaction with devices.

**Key Challenges and Limitations**

**1. Handwriting Variability:** Handwriting styles vary significantly among individuals and cultures. This variability, coupled with differences in letter shapes, sizes, and orientations, makes consistent recognition challenging.

**2. Noise and Low-Quality Input:** Handwriting recognition systems must be robust to noise and image quality issues, such as smudges, faded ink, or image compression artifacts, which are common in scanned documents.

**3. Multilingual Recognition:** Recognizing multiple languages and scripts, including cursive and non-Latin characters, is complex, as each language presents unique challenges in structure and features.

**4. Data Scarcity:** Large, labelled datasets are essential for training accurate handwriting recognition models. However, labelled data for historical or rare languages can be difficult to obtain.

**5. Computational Requirements:** Deep learning models, especially those that are highly accurate, require significant computational resources for training and deployment, which can be a barrier for smaller organizations.

**CHAPTER 1**

**INTRODUCTION, OBJECTIVES & PURPOSE OF STUDY**

**1.1 Introduction to Machine Learning**

Machine Learning is a part of Artificial Intelligence (AI). It uses various techniques to provide machines the ability to learn with the help of large amounts of data. While Machine Learning has reduced the burden on programmers to explicitly program the computers, it has touched the areas which have never been explored before. This has led to various technological advances. There are 3 types of Machine Learning algorithms:

● Supervised Learning Algorithms

● Unsupervised Learning Algorithms

● Reinforcement Learning Algorithms

A diagram of machine learning

Description automatically generated

In Supervised learning, you train the machine using data which is well "checked." It suggests some data is starting to be marked with the correct answer. It might be stood out from acknowledging which occurs inside seeing a chief or an instructor.

A regulated taking in computation gains from named getting ready data, urges you to foresee results for unforeseen data. Adequately building, scaling, and passing on exact oversaw AI Data science models requires some genuine energy and particular capacity from a gathering of astoundingly skilled data scientists. Furthermore, Data scientists must revamp models to guarantee the encounters given remain substantial until its data changes.

Independent learning is an AI technique, where you don't need to coordinate the model. Taking everything into account, you need to allow the model to manage its own to discover information. It essentially deals with the unlabelled data.

Independent learning estimations license you to perform all the more confounding taking care of tasks diverged from coordinated learning. But, independent learning can be more eccentric differentiated and other typical learning significant learning and stronghold learning methods.

**Importance of Machine Learning:**

Through cutting edge processing advancements, AI isn't what it resembled previously. It was conceived from design acknowledgment and the hypothesis that PCs can learn without being modified to play out specific assignments. Researchers intrigued by AI needed to check whether PCs could gain from information. The iterative part of AI is significant on the grounds that as models are presented to new information, they can autonomously adjust. They gain from past calculations to deliver dependable, repeatable choices and results.

**Simple Programming Model:** Machine Learning models can be programmed using Python which is one of the easiest yet the most productive programming languages. Also Python has one of the biggest developer communities who have provided us with huge libraries which make machine learning algorithms easy to implement.

**Cost Effective:** Machine Learning with Python guarantees that it doesn't consume a gap in your pocket with regards to overseeing humongous measures of data. This has been an issue with ancestor programming which has been cost restrictive. Numerous organizations have needed to erase and downsize data with the end goal to diminish their expenses.

**1.2 Introduction to Handwriting Recognition:**

It's the ability of the computer to interpret and recognize handwritten input. It’s sometimes known as HTR (handwritten text recognition). This could be a scanned handwritten document or a photo of a handwritten note, for instance. The growth and proliferation of touch screens add another way to input handwriting.

The goal of handwriting recognition has been around since the 80s — and has suffered from accuracy issues from the beginning. There are two types of handwriting recognition. First is the older of the two, known as offline handwriting recognition. This is where the handwritten input is scanned or photographed and given to the computer.

The second is online, which is where the writing is input through a stylus/touchscreen. This offers the computer more clues about what’s being written. (For instance, stroke direction and pen weight)

**1.3** **Objective of Study:**

Primary objective of this study is to accurately and efficiently interpret, digitize and convert handwritten text into a digital format.

Goal is achieved by developing algorithms capable of identifying and classifying handwritten characters, words and sentences across various styles and languages, often from noisy or low-quality sources such as scanned images or photographs.

We will be using IAM database for the purpose of training and validation.

**1.4 Scope & Purpose of Study:**

The scope defines the boundaries and areas of focus. For handwritten recognition, it could include:

1. **Image Processing and Preprocessing:** Understanding methods to preprocess handwritten text images, such as resizing, noise reduction, thresholding, and normalization.
2. **Machine Learning Algorithms:** Exploring and comparing various machine learning techniques for handwritten recognition, particularly deep learning models like CNNs, RNNs, or combinations of both.
3. **Dataset Utilization:** Working with standard handwritten datasets, such as MNIST (for digit recognition) or IAM (for cursive writing).
4. **Evaluation Metrics:** Defining and using metrics like accuracy, precision, recall, and F1-score to evaluate model performance.
5. **Limitations and Challenges:** Identifying factors that affect recognition accuracy, like varied handwriting styles, quality of scanned images, and limitations of the chosen model.

**Purpose of Study**

The purpose describes why this study is important and what it aims to achieve:

1. **Advancing Handwritten Text Recognition**: To explore and demonstrate the effectiveness of AI in accurately recognizing and converting handwritten text into digital text.
2. **Automation and Efficiency**: To highlight how automated handwritten recognition can streamline processes in various fields like banking (check processing), postal services, and archival work, reducing the need for manual data entry.
3. **Improving Accessibility**: To help preserve historical documents or create digital archives by converting handwritten notes or documents into searchable text.
4. **Educational Contribution**: To offer insights into how AI and machine learning work in practical applications, contributing to educational understanding of ML models for image processing and recognition.
5. **Technological Development**: To contribute to developing better models for specific languages, scripts, or use cases, potentially offering new algorithms or approaches for future research in handwritten recognition.

**CHAPTER 2**

**LITERATURE REVIEW**

**2.1 The Recognition for Handwritten English Letters**

A Review Character affirmation is one of the most captivating and testing research areas in the field of Image taking care of. English character affirmation has been extensively amassed over the most recent 50 years. Nowadays different approaches are in use for character affirmation. File affirmation, progressed library, scrutinizing bank store slips, examining postal addresses, removing information from checks, data area, applications for charge cards, clinical inclusion, credits, charge records, etc are application locales of electronic report getting ready. This paper gives a survey of assessment turn out finished for affirmation of physically composed English letters. In Hand formed substance there is no necessity in the creating style. Interpreted letters are difficult to see because of various human handwriting styles, assortment in point, size and condition of letters. Various strategies of physically composed character affirmation are analyzed here close by their introduction.[1]

**2.2 Feature Extraction for Handwritten Alphabets Recognition System which is diagonal based Using Neural Network:**

A detached interpretation in successive request character affirmation systems using a multilayer feed forward neural association is depicted in the paper. Another strategy, called, corner to corner-based component extraction is introduced for eliminating the features of the physically composed letters all together. Fifty instructive records, each containing 26 letter sets created by various people, are used for setting up the neural association and 570 particulars translated all together characters are used for testing. The proposed affirmation system performs well indeed, yielding more raised degrees of affirmation 6 accuracy diverged from the structures using the conventional level and vertical procedures for incorporating extraction. This system will be sensible for changing over translated reports into essential substance structure and seeing physically composed names.[2]

**2.3 Using Neural Network for optical character recognition by Image**

**Preprocessing:**

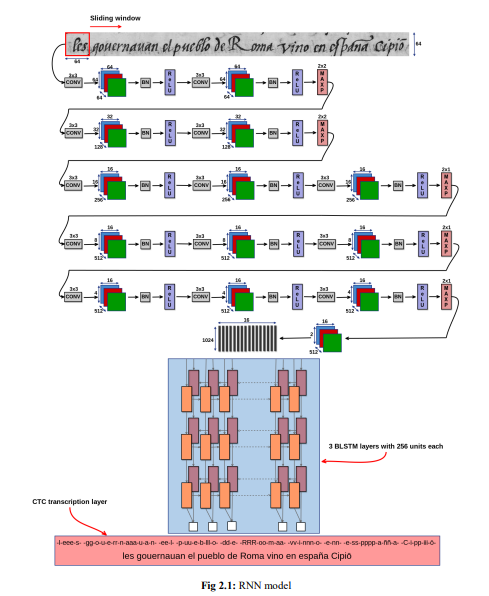
Essential errand of this postulation is to make a hypothetical and pragmatic premise of preprocessing of printed text for optical character acknowledgment utilizing forward-feed neural organizations. Show application was made and its boundaries were set by aftereffects of acknowledged investigations.[3]

**2.4 Handwriting Recognition of Historical Documents:**

The majority of current offline handwritten text recognition (HTR) algorithms operate at the line level, converting the text-line picture into a series of feature vectors. These characteristics are supplied into an optical model (for example, a recurrent model).

In order to distinguish handwritten characters, a neural network was used. Recent work on document-level text identification and localization and combined line segmentation and identification at the paragraph level has yielded encouraging results. The end outcome However, the finest outcomes in terms of recognition are still to be found. Systems that work at the line level are able to do this.

In this model thirteen stack convolutional layers and three bidirectional layers are used having 256 units in each layer. ReLU is also used to have non- linearity after each layer of CNN. Bidirectional LSTM is used with CTC [5] loss function for making the model end to end trainable. This model was found to be very accurate in predicting on the READ dataset and this model also won second place in ICDAR2017 competition



**CHAPTER 3**

**RESEARCH METHODOLOGY**

**3.1 Software Used:**

● Jupyter Notebook

● Google Colab

**3.1.1 Requirements of System:**

Cloud based google colab utility

**Jupyter Notebook:**

The Jupyter Notebook is an open-source web application that licenses you to make and share reports that contain live code, conditions, observations and record text. Uses include: data cleaning and change, numerical reenactment, quantifiable showing, data recognition, AI, and impressively more. Jupyter Notebooks are a fantastic technique to make and rehash on your Python code for data assessment.

**Why a Jupyter Notebook?**

They turn into latex reports easy. They also turn into slideshows easy. They also let you run blocks of code easily.

**Some other alternatives:** PyCharm, RStudio

**Google Colab:**

It is made by Google’s Research Department. It's used to execute and write Python code online in a browser. It is very useful and efficient for people who are trying to learn Machine Learning.

**3.1.2 Minimum Hardware Requirement:**

12 GB RAM

TPU Processor is preferable for faster processing

Python 3 runtime

A screenshot of a computer

Description automatically generated

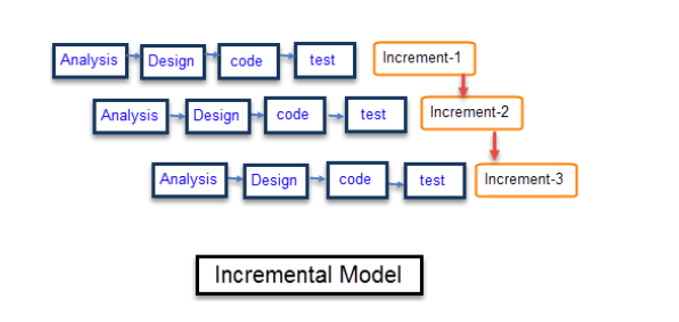
**3.2 Modelling**

We are using an incremental model in this project.

**What is an Incremental Model?**

Incremental Model is a system of programming improvement where prerequisites are broken into various autonomous modules of programming advancement cycle. Steady advancement is done in endeavors from investigation plan, execution, testing/confirmation, support. Each accentuation will encounter the necessities, structure, coding and testing stages. Each resulting appearance of the system adds ability to the last release until the point that all arranged value has been completed.

The structure is placed into age when the essential expansion is passed on. The primary growth is much of the time a middle thing where the crucial necessities are tended to, and worthwhile features are remembered for going with increments. At the point when the middle thing is broken somewhere around the customer, there is other plan improvement for the accompanying enlargement



Pictorial representation of Incremental model

**3.3 Designing**

The volume of information is extending every day that we can manage business trades, sensible data, pictures, chronicles and various others. In this way, we need a system that will be good for isolating the information open and that can normally make reports, viewpoints or overview of data for better use.

**There are three phases in designing a model:**

**Training Data:** It is the data on which the machine learning model learns and trains itself. Usually, it is large in comparison to test data.

**Validation Dataset:** Hyper-parameters of a classifier. It is sometimes also called the development set.

**Test Data:** It is the data on which testing is done, and the model is evaluated on the basis of results obtained from this dataset. Usually, it is small in comparison to training data.

A diagram of a product

Description automatically generated with medium confidence

**Fig 3.3**

**The 7 Steps of approaching a framework in Machine Learning:**

**Stage-1:** Data Collection.

**Stage-2:** Data Preparation.

**Stage-3:** Choose a Model.

**Stage-4:** Train the Model.

**Stage-5:** Evaluate the Model.

**Stage-6:** Parameter Tuning.

**Stage-7:** Make Prediction

A diagram of a model

Description automatically generated

**3.4** **ALGORITHMS**

**3.4.1 Algorithms for Classification Techniques**

**3.4.1.1 Logistic Regression Algorithm:**

Logistic Regression is an algorithm that is utilized for binary classification and is the most basic algorithm for classification techniques. It uses sigmoid function for predicting the output. The algorithm makes use of the decision boundary to predict the output.

**Tools:** Python, Jupyter notebook.

A diagram of a data flow

Description automatically generated

Fig 4.1.1

**3.4.1.2 Artificial Neural Networks:**

In ANN (Artificial Neural Networks) we manufacture a type of transient states, which allows the machine to learn in a more refined manner. The objective of this article is to draw out the arrangement of ANN figuring in relating to the value of the psyche. It is truly said that the working of ANN takes its hidden establishments from the neural association living in the human brain. ANN chips away at something suggested as Hidden State. These covered states resemble neurons. All of these covered states is a transient structure which has a probabilistic lead. A framework of such hid state goes probably as a platform between the data and the yield.

A diagram of a neural network

Description automatically generated

Fig 4.1.2

**3.4.1.3 Random Forest Algorithm:**

The estimation is incredibly standard in various competitions. The end yield of the model takes after a black box and from this time forward should be used wisely. Subjective woods look like a bootstrapping computation with a Decision tree model. In Random Forest, we create different trees rather than a lone tree in the CART model. To describe another article subject to attributes, each tree gives a portrayal, and we express the tree "votes" for that class. The forest area picks the course of action having the most votes (over all the trees in the forested areas) and if there ought to emerge an event of backslide, it takes the ordinary of yields by different trees.[4]

A diagram of a tree

Description automatically generated

Fig 4.1.3

A graph of a number of trees

Description automatically generated

Fig 4.1.4

**Two stages of Random Forest Algorithm:**

**Making of random forest code.**

Code for prediction using random forest.

4.2 Comparing Different Algorithms:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm** | **Technique** | **Scalability** | **Faster** |
| Logistic Regression | Apply sigmoid function and predict. | Depends on how large the dataset is. | It is very fast for binary classification. |
| Artificial Neural Networks | Activate the nodes of the next layer and then apply backpropagation. | Can take huge number of features and gets refined as no. of nodes increases | Fast for multiclass classification |
| Random Forest | Each tree votes for a class and the one with the most vote wins. | Highly scalable and is very efficient for multiclass classification. | Fast for multiclass classification |

**Fig 4.2.1: Comparison Table of Algorithm**

**3.4.3 Neural Network:**

A neural association is a movement of estimations that attempts to see concealed associations in a lot of data through a cycle that duplicates the way where the human brain works. Neural associations can conform to changing data, so the association makes the best result without hoping to redesign the yield models.

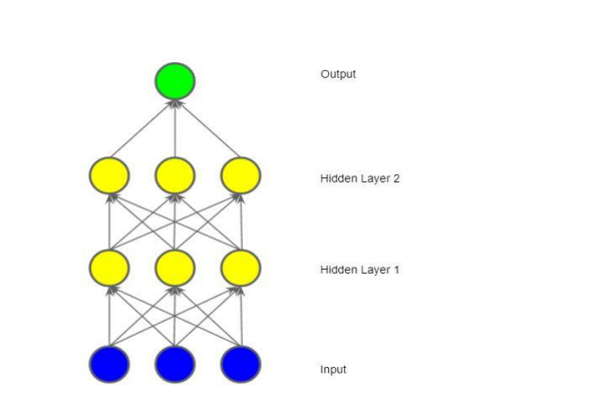


Fig 4.3 Neural Network with two hidden layers

**3.4.4 Convolutional Neural Network:**

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning estimation which can take in a data picture, consign centrality (learnable burdens and tendencies) to various points/objects in the image and have the choice to isolate one from the other. The pre-taking care required in a ConvNet is a ton lower when diverged from other portrayal figurines. While in unrefined methodologies channels are hand-planned, with enough getting ready, ConvNets can get comfortable with these channels/characteristics.

The designing of a ConvNet is like that of the accessibility illustration of Neurons in the Human Brain and was excited by the relationship of the Visual Cortex. Solitary neurons respond to upgrades simply in a kept territory of the visual field known as the Receptive Field. A collection of such fields covers the entire visual region.

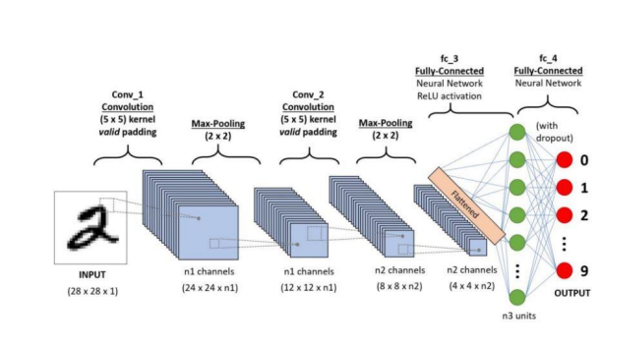


Fig 4.4

**3.4.5 Recurrent Neural Network:**

A recurrent neural network is a type of neural network having feedback, the output from the previous step is fed into the current step as an input. Unlike CNNs and ANNs they have a memory element where they can store the sequence of outputs. This feature makes them suitable for applications like speech recognition, handwriting recognition, and for making predictions.

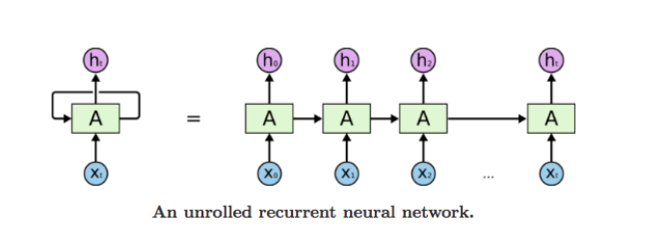


Fig 4.5 RNN

But there are some limitations of RNNs, and the 2 major limitations are:

* Exploding gradients
* Vanishing gradients

**3.4.4.1 Exploding gradients:**

In a neural network weights are updated constantly but if an error gradient is being used by the model to update the network. If an error gradient which is assigned a very large value is used to update the weights of the model, this could accumulate and become very huge with every iteration.

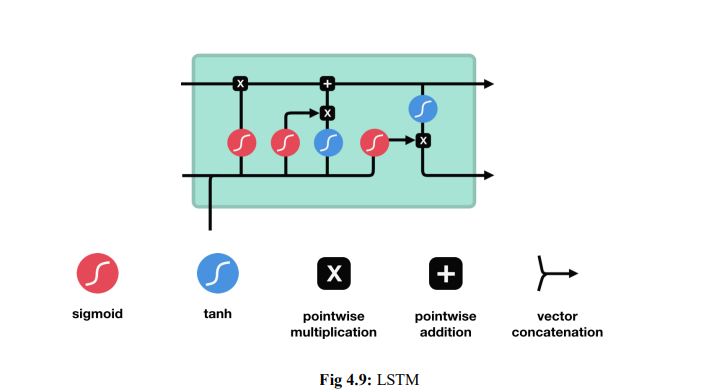
The problem of exploding gradients makes the network unstable and the loss of the network becomes very high. This problem can easily be solved by gradient clipping and squishing the gradients.

**3.4.4.2 Vanishing gradients:**

This is exactly the opposite of exploding gradients, here the values of gradients are very small. So the model stops learning then and skips ahead without learning them. This is very hard to solve as compared to exploding gradients. But this problem can be solved by the use of LSTMs.

**3.4.5 LSTM:**

Long short-term memory (LSTM) is a special kind of recurrent neural network which is capable of learning long term dependencies. LSTMs make it possible for RNNs to remember inputs for a longer period of time because of the presence of a memory element in LSTMs.



There are different operations happening inside the LSTM cell, these enable LSTM to remember the useful information and forget the information which will not help the model to improve further.

**3.4.6 CTC loss:**

Connectionist Temporal Classification Loss, or CTC Loss is very helpful in tasks like speech and handwriting recognition. It is help full where there are pauses or repeating words or alphabets.

Between a continuous (unsegmented) time series and a target sequence, it estimates a loss. It does this by accumulating the probabilities of alternative input-target alignments, yielding a loss value that is differentiable with respect to each input node. The input to target alignment is expected to be many-to-one.

A screenshot of a computer

Description automatically generated

Fig 4.6 CTC Functioning

**CHAPTER 4**

**DATA ANALYSIS**

The IAM dataset (Institute of Automation and Mathematics) is widely used for handwritten text recognition tasks and includes scanned images of handwritten English text, including both isolated words and full sentences. Let’s break down a basic data analysis approach for the IAM dataset, focusing on understanding and preparing the data for a handwriting recognition project.

**1. Loading and Understanding the Data**

* **Dataset Structure**: The IAM dataset includes several parts:
  + **Forms**: Full-page handwritten text samples.
  + **Lines**: Line-separated images from the full forms.
  + **Words**: Individual words extracted from lines.
  + **Transcriptions**: Text files containing the corresponding ground truth labels.
* **Metadata Files**: Often, the dataset includes metadata like writer ID, document type, and additional information that can be used for analysing writer variability or for separating training and test data.

**2. Exploratory Data Analysis (EDA)**

Performing EDA helps to understand the distribution, variety, and challenges within the dataset:

* **Data Distribution**:
  + **Word Lengths**: Calculate and visualize the distribution of word lengths. This can provide insights into how diverse the word samples are in terms of character count.
  + **Line and Form Lengths**: Look at the number of words per line and lines per form, as this can help in segmenting text data or designing multi-line recognition models.
  + **Character Frequencies**: Plot the frequency of characters (letters and special symbols) in the dataset. This will highlight any imbalance in character distribution, which can impact model training.
* **Handwriting Styles**:
  + **Writer Variability**: Check the number of unique writers and the number of samples per writer. Analyzing handwriting styles per writer can help understand how the model might generalize across different writing styles.
  + **Character Appearance**: Compare samples of the same character across different writers to assess variation.
* **Image Characteristics**:
  + **Image Sizes**: Check image dimensions to ensure consistent input sizes for the model. You may need to resize or pad images to a standard size for model training.
  + **Image Quality**: Look for samples with noise, blurred text, or poor contrast, as these might require preprocessing to enhance recognition accuracy.

**3. Visual Analysis**

* **Random Samples**: Display random samples from the dataset, including individual words, lines, and forms. This gives a quick visual understanding of the variety in handwriting styles, word sizes, and any potential noise or artifacts.
* **Sample Distributions**: Plot histograms or box plots of word lengths, line counts, and other features to visually assess the dataset’s composition.

**4. Statistical Analysis**

* **Text Length Statistics**: Calculate mean, median, and standard deviation of word lengths and line lengths. This information can help decide the model's input handling strategy.
* **Frequency Analysis of Characters**: Identify the most and least common characters to understand any class imbalance. This can help in determining if data augmentation or weighting strategies are needed for rare characters.

**5. Preprocessing Analysis**

* **Noise Analysis**: Identify samples that may contain noise or artifacts (e.g., background textures or handwritten corrections). Noise reduction methods like thresholding, blurring, or sharpening can be assessed here.
* **Text Alignment and Segmentation**: Evaluate the alignment of text (e.g., horizontal alignment, spacing) and determine if further segmentation might be necessary, particularly for line and form images.

**6. Augmentation Needs Assessment**

* Based on variability in writing style, check if augmenting the data with transformations (rotation, scaling, etc.) could improve model robustness. This is especially useful if there is a high variance in character appearances or word orientations.

**7. Insights and Summary**

Summarize the key findings, such as:

* Common challenges in the dataset (e.g., variations in handwriting, noise).
* Strategies for handling imbalances in character frequency.
* Required preprocessing steps, such as resizing, contrast adjustment, or noise reduction, before model training.

**Datasets Used:**

The **IAM** Handwriting Database contains forms of handwritten English text, which can be used to train and test handwriting recognition models. The database contains forms of unconstrained handwritten text, which were saved as PNG images with 256 gray levels. Figure 2 provides samples of a completed form, a text line, and some extracted words. Here is a [link](http://www.fki.inf.unibe.ch/databases/iam-handwriting-database) to the IAM dataset.

The IAM Handwriting Database is structured as follows:

* 657 writers contributed samples of their handwriting
* 1,539 pages of scanned text
* 5,685 isolated and labelled sentences
* 13,353 isolated and labelled text lines
* 115,320 isolated and labelled words

We will only be using the labelled words images and *words.txt* (ASCII) files. After placing the downloaded files inside the data directory, we map each image to its label from the *words.txt* file for further processing.

A close-up of a letter

Description automatically generated

**CHAPTER 5**

**CODING IMPLEMENTATION**

**6.1 Tools and Techniques:**

We have used Python for implementing our project.

**Why Python?**

Python is very easy to understand and is a beginner friendly high-level language.

Python's simplicity allows us to write reliable systems. Python is more mechanical and models are quickly trained for machine learning.

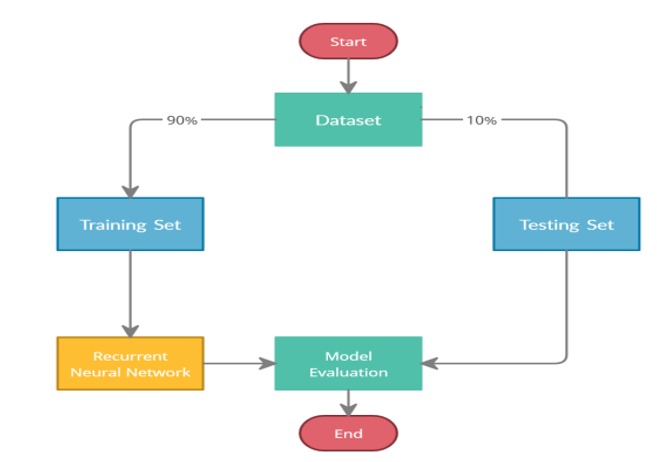


Fig 6.1 Block Diagram

The data from the dataset is divided into training and validation sets which are in the ratio of 10:1. Training set contains 38000 images whereas the testing set contains 3800 images. The RNN is trained on the images from the training set and then its accuracy is checked by using the model on the validation set.

**5.3 Libraries Used:**

We have used following libraries in the implementation code:

● Pandas

● Numpy

● Matplotlib

● Tensor-Flow

● Keras

**5.4 Specification of virtual machine used:**

CPU : Intel(R) Xeon(R) CPU @ 2.20GHz

GPU : Tesla P100 with 16GB VRAM

RAM : 14 GB DDR4

HDD : 73 GB

**Codebase Link:**

https://github.com/RAKAMSTR/MBA\_PROJECT/blob/master/Handwriting\_Recongnition.ipynb

**Little bit about Libraries Used**

● **OS:** Provides operating system functionalities, potentially useful for tasks like:

* + File path manipulation during data loading from the file system.
  + Saving and managing model checkpoints during training.

● **Python:** Python is high level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

● **Pandas**: is a [Python](https://www.python.org/) package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, **real-world** data analysis in Python.

● **Numpy:** Numerical Python is an open source Python library that’s widely used in science and engineering. The NumPy library contains multidimensional array data structures, such as the homogeneous, N-dimensional ndarray, and a large library of functions that operate efficiently on these data structures.

● **Matplotlib:** Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

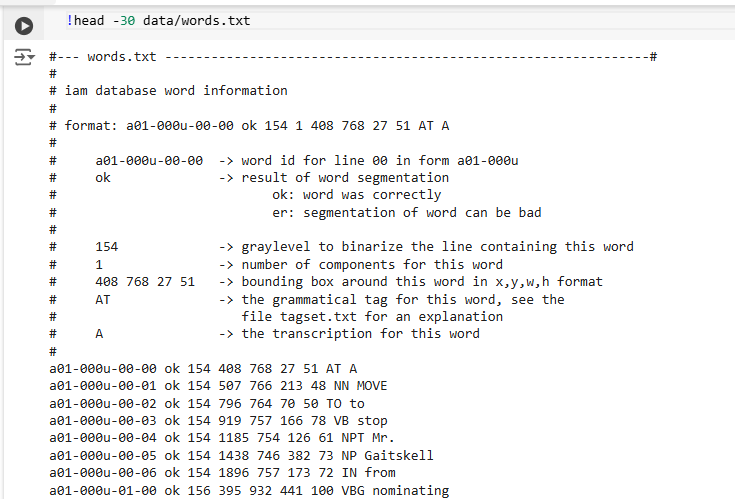
● **Tensor-Flow:**TensorBoard is a tool for providing the measurements and visualizations needed during the machine learning workflow. It enables tracking experiment metrics like loss and accuracy, visualizing the model graph, projecting embeddings to a lower dimensional space, and much more.

● **Keras:** Keras is the high-level API of the TensorFlow platform. It provides an approachable, highly-productive interface for solving machine learning (ML) problems, with a focus on modern deep learning. Keras covers every step of the machine learning workflow, from data processing to hyperparameter tuning to deployment. It was developed with a focus on enabling fast experimentation.

**\**

**Dataset:**

We are using IAM Dataset which is available at location [***https://git.io/J0fjL***](https://git.io/J0fjL)



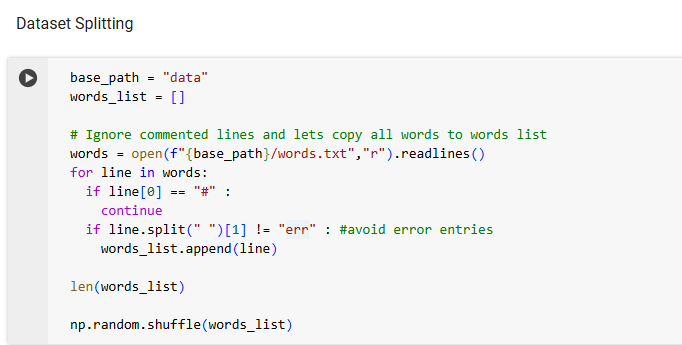
Here goes the imports,

A screenshot of a computer

Description automatically generated

Now let’s focus on splitting data into segments so that it can be used for training, testing and validation.

Proportion can be anything but will keep the ratio of 90:5:5. Here 90% data would be used for training and rest all 5-5 % would be used for validation and testing. On basis of which we will decide the accuracy of the model.

****

**Resize images without distortion**

In order to process images, we need to make them of uniform size. So all images needs to be resized accordingly and this should happen without affecting their aspect ratio and content off course.

**A computer screen shot of a computer code

Description automatically generated**

Our dataset contains different type of data which we refer as modality of data. Keras provide various set of preprocessing layers in order to process modality of data.

Few of them are listed below,

* Text preprocessing:- TextVectorization layer
* Numerical features preprocessing layers:- Normalization layer, Spectral Normalization layer
* Categorical features preprocessing layers:- CategoryEncoding layer, HashedCrossing layer, *StringLookup layer*
* Image preprocessing layers:- Resizing layer

We are using StringLookup layer in order to maps strings to (possibly encoded) indices. This layer translates a set of arbitrary strings into integer output via a table-based vocabulary lookup. This layer will perform no splitting or transformation of input strings.

This means that if we have two labels e.g. horse & zebra then our character vocabulary should be {h,o,r,s,e,z,b,a} ( Without a special tokens )

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

Here are few examples from the dataset samples, if its were stretched then it would have broken in pixels.

A close-up of words

Description automatically generated

Let’s prepare dataset for training, validation and test purposes.

A black text on a white background

Description automatically generated

A computer code with text

Description automatically generated

Visualize few samples,

A computer screen shot of a program

Description automatically generated

A group of black squares with words

Description automatically generated

Building a model,

A screenshot of a computer

Description automatically generated

**Evaluation Metric**

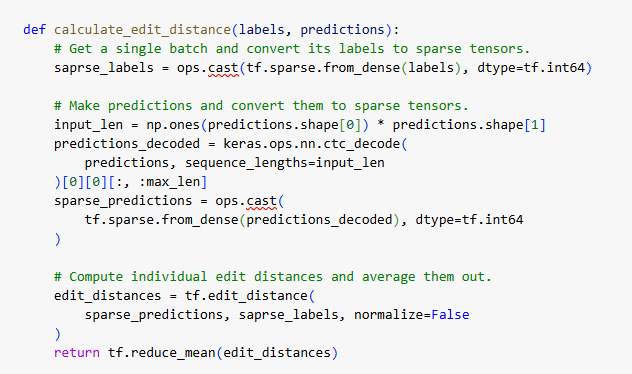
**EDIT DISTANCE --** In [computational linguistics](https://en.wikipedia.org/wiki/Computational_linguistics) and [computer science](https://en.wikipedia.org/wiki/Computer_science), edit distance is a [string metric](https://en.wikipedia.org/wiki/String_metric), i.e. a way of quantifying how dissimilar two [strings](https://en.wikipedia.org/wiki/String_(computing)) (e.g., words) are to one another, that is measured by counting the minimum number of operations required to transform one string into the other. Edit distances find applications in [natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing), where automatic [spelling correction](https://en.wikipedia.org/wiki/Spell_checker) can determine candidate corrections for a misspelled word by selecting words from a dictionary that have a low distance to the word in question. In [bioinformatics](https://en.wikipedia.org/wiki/Bioinformatics), it can be used to quantify the similarity of [DNA](https://en.wikipedia.org/wiki/DNA) sequences, which can be viewed as strings of the letters A, C, G and T. (Source- *Wikipedia*)

We segregate the validation images and their label for convenience,

**A computer code with black text

Description automatically generated**

Now, we create a callback to monitor the edit distances.

****

**A screen shot of a computer code

Description automatically generated**

Let’s kick off training,

A computer screen shot of a code

Description automatically generated

**Final Prediction Results**

As you can see the prediction is done accurately for few models and for few its unable to proceed properly.

It’s because dataset was limited in our case, to predict the result.

**EPOCHS**: This integer represents the number of times the entire training dataset will be passed through the model during training. In this case, the model will be trained for 40 epochs. This hyperparameter fine-tunes the model's learning process and influences its ability to learn patterns from the data.

Training for models started and completed for below epochs,

**1) First Training Iteration**

Training for **10 epochs**

A screenshot of a computer

Description automatically generated

**Predictions** done on test data set

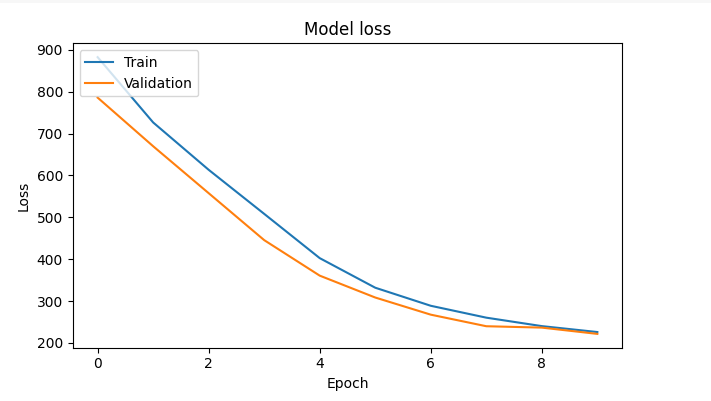
A close-up of several letters

Description automatically generated

A close-up of a list of words

Description automatically generated

**Model loss** when trained on **10** epochs



As the loss decreases accuracy increases,

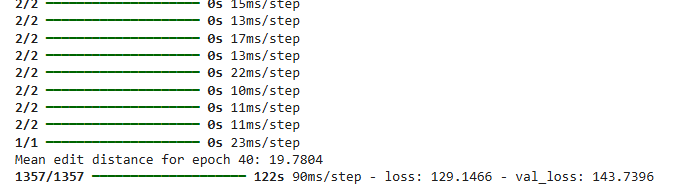
**Accuracy** Result for **10** epochs,

A close up of numbers

Description automatically generated

**2) Second Training Iteration**

Training for **40 epochs –** Loss has significantly been reduced.

****

**Predictions** done on test data set

As you can see, models’ prediction accuracy has been increased.

**A close-up of several black squares

Description automatically generated**

**A close-up of several words

Description automatically generated**

**Model loss** when trained on **40** epochs

A graph with blue and orange lines

Description automatically generated

**Accuracy** Result for **40** epochs,

A black text on a white background

Description automatically generated

**CHAPTER 6**

FINDINGS, SUGGESTIONS, RECOMMENDATIONS

**Findings:**

1. Variability in Handwriting Styles:

* The dataset likely exhibits a high level of variability in handwriting styles, with different letter formations, sizes, and orientations due to different writers. This can lead to challenges in achieving consistent recognition accuracy.

1. **Class Imbalance:**

* Some characters or words may appear more frequently than others, creating an imbalance that can affect the model’s ability to recognize less common characters accurately.

1. **Image Quality and Noise:**

* Some samples may have artifacts, noise, or blurred text, particularly if they were scanned from physical documents. Low-quality images can hinder recognition and reduce model performance.

1. **Character Segmentation Challenges:**

* In images with words or lines of text, characters may be connected or have irregular spacing. This can make it difficult for the model to distinguish individual letters, especially if they are cursive.

1. **Impact of Data Preprocessing:**

* Data preprocessing, such as resizing, normalization, and noise reduction, improves model performance but may require careful fine-tuning to avoid losing important text details.

**Recommendations:**

**1. Use Data Augmentation to Improve Generalization:**

* Apply data augmentation techniques like rotation, scaling, or adding slight noise. This will help the model generalize better across different handwriting styles and orientations.

**2. Implement a Convolutional Neural Network (CNN)-Based Model:**

* CNNs are effective for feature extraction in image-based tasks. Consider starting with well-known architectures (e.g., LeNet, VGG, or ResNet) and fine-tuning them to handle the unique challenges of handwriting recognition.

**3. Address Class Imbalance with Weighted Loss Functions:**

* To manage the imbalance in character frequency, consider using a weighted loss function. This can help the model pay more attention to less common characters, improving recognition accuracy across all character types.

**4. Segment Text for Improved Recognition:**

* For more complex data (e.g., lines or forms), segment the text into smaller parts like individual words or characters. Character segmentation can significantly improve recognition accuracy, especially when dealing with cursive or highly variable writing.

1. **Employ Sequence Models for Longer Text Recognition:**

* For recognizing sequences (like whole words or sentences), consider adding a recurrent layer (e.g., LSTM or GRU) to the model. CNNs can extract features, while the recurrent layer can process sequential dependencies, particularly in cursive writing.

1. **Optimize Image Preprocessing:**

* Implement consistent preprocessing steps, such as resizing all images to a fixed dimension, normalizing pixel values, and applying thresholding techniques to remove background noise. Preprocessing can improve model robustness and speed up training.

**CHAPTER 7**

**CONCLUSION**

The accuracy of this model can be further improved by using more training data which will help this model to learn and generalize better. Also with more number of epochs, prediction accuracy will increase further.

Some of the images in the dataset are not of very good quality and the annotations of some images are also wrong. Removing such images will also help in model’s learning.

Handwriting recognition is an increasingly viable and transformative technology, driven by advancements in Artificial Intelligence & Machine Learning.

The integration of deep learning, hybrid CNN-RNN architectures, and attention mechanisms has improved recognition accuracy, enabling practical applications across industries.

Despite challenges such as handwriting variability, noise, and multilingual support, ongoing research and innovation are pushing the boundaries of what is achievable in handwriting recognition.

Future developments are expected to make handwriting recognition more accessible, accurate and adaptable. Opening new possibilities for digitization, automation and interaction across both professional and personal domains.

**SCOPE FOR FUTURE STUDY**

**1. Attention Mechanisms:**

Attention mechanisms in neural networks allow the model to focus on specific parts of the input image, which enhances accuracy in recognizing complex or cursive handwriting.

**2. Use pre-trained learning models for prediction:**

Transfer learning enables models pertained on large datasets to adapt to handwriting recognition tasks with limited data, reducing the need for extensive labeled data.

**3. End-to-End systems on cloud solutions:**

Cloud systems provide capabilities to perform task faster and efficiently. It enables user to worry about only business logic rather than infrastructure. Google cloud vision & Microsoft Azure OCR offers scalable solutions, making handwriting recognition more accessible and integrable for organizations.

**4. Multilingual and Multimodal models:**

Future handwriting recognition systems are likely to integrate multilingual recognition capabilities and multimodal data sources, such as combining handwriting with audio or context, to improve accuracy and versatility.

**5. Further increase in epochs (training) will enhance accuracy:**

By increasing numbers of epochs for training and increasing data to train the model will enhance accuracy of the model in predicting the correct result.

**REFERENCES**

[1] Nisha Sharma et al, “Recognition for handwritten English letters: A Review”International Journal of Engineering and Innovative Technology (IJEIT) Volume 2, Issue 7

[2] J.Pradeep et al,, “Diagonal based feature extraction for handwritten alphabets recognition System using neural network”International Journal of Computer Science and Information Technology (IJCSIT), Vol 3, No 1

[3] Miroslav NOHAJ, Rudolf JAKA, “Image preprocessing for optical character recognition using neural networks”Journal of Pattern Recognition Research

[4] Jehad Ali, Rehanullah Khan, Nasir Ahmad, Imran Maqsood " Random Forests and Decision Trees ".

https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-inpython/

[5] A. Graves, S. Fern´andez, F. Gomez, and J. Schmidhuber,“Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in Proceedings of the 23rd international conference on Machine learning. ACM, 2006, pp. 369–376

[6] IAM dataset is being used to train model (https://git.io/J0fjL)

[7] https://en.wikipedia.org/wiki/Edit\_distance