A PROJECT REPORT

on

"HANDWRITTEN OPTICAL CHARACTER RECOGNITION "

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN
INFORMATION TECHNOLOGY

BY

VISHAL GARG	2106084
ARGHAJIT DAS	2106100
MAYANK RAJPUT	2106122
BIKASH KUMAR MAHANTA	2106202
PIYUSH JAISWAL	2106232

UNDER THE GUIDANCE OF
RAGHUNATH DEY

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA - 751024

April 2024

A PROJECT REPORT

on

"HANDWRITTEN OPTICAL CHARACTER RECOGNITION"

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN INFORMATION TECHNOLOGY

BY

VISHAL GARG	2106084
ARGHAJIT DAS	2106100
MAYANK RAJPUT	2106122
BIKASH KUMAR MAHANTA	2106202
PIYUSH JAISWAL	2106232

UNDER THE GUIDANCE OF

RAGHUNATH DEY

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA -751024

April 2024

School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is certify that the project entitled

"HANDWRITTEN OPTICAL CHARACTER RECOGNITION"

submitted by

VISHAL GARG	2106084
ARGHAJIT DAS	2106100
MAYANK RAJPUT	2106122
BIKASH KUMAR MAHANTA	2106202
PIYUSH JAISWAL	2106232

is a record of bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering OR Information Technology) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2023-2024, under our guidance.

Date: 12 / 04 / 2024

(Raghunath Dey)
Project Guide

Acknowledgements

We are profoundly grateful to **Raghunath Dey** of **Affiliation** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion.

VISHAL GARG
ARGHAJIT DAS
MAYANK RAJPUT
BIKASH KUMAR MAHANTA
PIYUSH JAISWAL

ABSTRACT

Handwritten Optical Character Recognition (OCR) has witnessed advancements through various deep learning models, including Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, transformer-based models, attention-based models, capsule networks, and graph neural networks. This report explores the efficacy of these models in handwritten OCR tasks, evaluating their performance on benchmark datasets. Through extensive experimentation, A hybrid model, combining Conditional Random Fields (CRF) and a beam search algorithm, attained an accuracy of 78.72% on the IBM_UB_1 dataset. Seam carving, yielded a remarkable 98% accuracy on the ICDAR2007 Handwriting Segmentation Contest dataset. Utilizing Artificial Neural Networks (ANN) on handwritten English sentences, an overall recognition accuracy of nearly 92% was achieved. A fusion of Local Binary Patterns (LBP) and Support Vector Machine (SVM) realized a commendable 96.5% accuracy for character recognition. Deep Neural Networks achieved 88.8% accuracy, while CNN models demonstrated effectiveness with accuracies of up to 94% and 98.94% on different datasets, including the A-Z Handwritten Alphabet dataset. Recognizing handwritten English text using Recurrent Neural Networks (RNN) with Long Short-Term Memory (LSTM) units, coupled with Connectionist Temporal Classification (CTC) layers, resulted in a high accuracy level of 90%. Employing CNNs and RNNs together, an overall accuracy of 93.7% was obtained. Support Vector Machine (SVM) combined with Deep Convolutional Neural Networks (DCNN) achieved an average accuracy of 94.53%. Conversely, SVM outperformed K-Nearest Neighbours (K-NN) with a word recognition rate of 75%. These findings provide valuable insights into the strengths and weaknesses of different deep learning models in handwritten OCR, offering guidance for further research and practical implementations in real-world scenarios.

Henceforth the above results prove that CNN (Convolutional Neural Networks) is the most effective algorithm for predicting and identifying the handwritten characters with higher accuracy and precision.

Keywords:
Convolutional Neural Networks (CNN), Support Vector Machine (SVM), Handwritten Optical Character Recognition (OCR), Recurrent Neural Networks (RNNs), K-Nearest

Contents

List of Figures

Chapter 1
Introduction

In today's digital age, the conversion of handwritten text into digital formats has become increasingly vital across various industries and applications. Handwritten Optical Character Recognition (OCR) plays a crucial role in this process by enabling the automated extraction of information from handwritten documents. However, despite significant advancements in OCR technology, several gaps persist in current solutions, necessitating the development of more robust and accurate systems.

The importance of this project lies in its potential to address these existing gaps and enhance the efficiency and effectiveness of handwritten OCR. Firstly, current OCR systems often struggle with accurately recognizing handwritten text due to the inherent variability and complexity of handwriting styles. As a result, there is a pressing need for improved algorithms and models capable of deciphering diverse handwriting patterns with higher accuracy.

Furthermore, existing OCR solutions may lack scalability and adaptability to different languages, writing styles, and document formats. This limitation hampers their applicability across various domains and necessitates the development of more versatile and adaptable OCR systems.

Moreover, the reliance on traditional OCR techniques may lead to suboptimal results in scenarios with degraded or poor-quality handwritten documents, such as historical manuscripts or archival materials. These documents often contain faded ink, smudges, or irregularities, posing significant challenges for OCR systems.

The structure of this report reflects a comprehensive exploration of the current landscape of handwritten OCR, including an analysis of the shortcomings and limitations of existing solutions. By identifying these gaps, the report aims to delineate the specific areas where improvements are needed and provide a roadmap for the development of more effective OCR systems.

Through the subsequent sections, the report will delve into the methodology employed in this project, highlighting the use of Convolutional Neural Network (CNN) models as a promising approach to address the identified gaps. Additionally, the report will present the results and findings obtained through extensive experimentation, including comparisons with alternative models and evaluations on benchmark datasets.

Overall, this project seeks to contribute to the advancement of handwritten OCR technology by bridging the existing gaps in current solutions and developing more accurate, scalable, and adaptable systems. By elucidating the importance of this endeavor and outlining the structure of the report, this section sets the stage for the subsequent discussions and analyses.

Chapter 2
Basic Concepts/ Literature Review

2.1 What is OCR?

OCR, short for Optical Character Recognition, is a technology used to convert scanned documents or images containing text into editable and searchable digital text. It identifies characters within these documents using algorithms and techniques like image preprocessing, pattern recognition, and machine learning. OCR has various applications, including document digitization, data entry automation, and accessibility solutions for visually impaired individuals.

2.2 ML models:

1. **Convolutional Neural Networks (CNNs):**
   Overview: CNNs are a type of deep learning architecture designed for image-based tasks. They use convolutional layers to automatically learn hierarchical features from pixel data.
   Application: CNNs are widely employed in handwritten character recognition due to their ability to capture spatial patterns and features in images. They excel in recognizing shapes, strokes, and variations in handwritten characters.
   Accuracy: CNNs have achieved high accuracy rates exceeding 90% and even reaching 95% or higher on well-curated datasets.

2. **Recurrent Neural Networks (RNNs):**
   Overview: RNNs are neural networks designed to handle sequential data by maintaining hidden states that capture information about previous inputs in a sequence.
   Application: RNNs are useful for tasks where the context of previous characters or strokes is crucial, making them suitable for cursive handwriting recognition or predicting the next character in a sequence.
   Accuracy: RNNs can achieve competitive accuracy rates, often ranging from 85% to 95% or higher on well-curated datasets.

3. **Support Vector Machines (SVMs):**
   Overview: SVMs are a type of supervised machine learning algorithm used for classification tasks. They aim to find a hyperplane that best

separates data points into distinct classes.
Application: SVMs can be applied to handwritten character recognition by training the model to distinguish between different characters based on extracted features. They are effective when dealing with high-dimensional data, making them suitable for image classification tasks.
Accuracy: Accuracy rates for SVMs can range from 80% to 95%, depending on factors such as feature selection, kernel choice, and the quality of training data.

4. **Hidden Markov Models (HMMs):**
   Overview: HMMs are statistical models that represent systems with hidden states and observable outputs. They are often used in sequence modeling and pattern recognition.
   Application: HMMs can be applied to recognize patterns and structures in handwritten characters by modeling the temporal dependencies in the writing process. They are particularly useful for recognizing dynamic aspects of handwriting, such as pen movements.
   Accuracy: Their accuracy can be competitive, often ranging between 80% and 95%, depending on the complexity of the writing styles and the modeling approach.

5. **Feature-based Methods:**
   Overview*:* Feature-based methods involve extracting relevant features from handwritten characters and using these features for classification.
   Application: Features such as edges, loops, endpoints, and curvature can be extracted from handwritten characters. Traditional machine learning algorithms or rule-based systems can then be employed for classification based on these features. While less common in comparison to deep learning methods, feature-based approaches are still used in certain scenarios, especially with limited data.
   Accuracy: Accuracy rates typically range from 70% to 90%, depending on the quality of features and the effectiveness of the classification algorithm.

6. **Dilated Convolutional Neural Networks (DCNN):**
   Overview: Dilated Convolutional Neural Networks (DCNNs), also known as atrous convolutional networks, are a type of convolutional neural network (CNN) architecture that incorporates dilated convolutions. Unlike

traditional convolutional layers where filters slide over input data with a fixed stride, dilated convolutions introduce gaps between filter elements, allowing for an expanded receptive field without increasing the number of parameters. This dilation factor controls the spacing between elements in the filter, enabling DCNNs to capture multi-scale features efficiently.

Application: DCNNs have gained prominence in various computer vision tasks, including image classification, semantic segmentation, and object detection. In handwritten character recognition tasks, DCNNs are applied to extract hierarchical features from input images and accurately classify individual characters. By leveraging dilated convolutions, DCNNs can capture contextual information across different scales, enhancing their ability to recognize characters in handwritten samples with varying styles and complexities.

Accuracy: DCNNs have demonstrated state-of-the-art performance in handwritten character recognition, often surpassing traditional machine learning approaches and earlier CNN architectures. By efficiently capturing multi-scale features, DCNNs can achieve high accuracy rates on standard handwritten character recognition datasets. With proper training and optimization, DCNN-based models can consistently achieve accuracies exceeding 95% on benchmark datasets, making them a reliable choice for handwritten character recognition tasks. Moreover, advancements in DCNN architectures, regularization techniques, and data augmentation strategies continue to enhance their accuracy and generalization capabilities in challenging handwriting recognition scenarios.

7. **Fully Convolutional Networks (FCNs):**

Overview: Fully Convolutional Networks (FCNs) are a variant of convolutional neural networks designed specifically for semantic segmentation tasks, where the goal is to classify each pixel in an image into a particular category.

Application: FCNs have been applied to handwritten character recognition tasks, particularly for segmenting individual characters within handwriting samples. By processing the entire
image at once and preserving spatial information through convolutional operations, FCNs can
effectively recognize and segment handwritten characters.

Accuracy: FCNs have demonstrated competitive accuracy rates in handwritten character recognition, often achieving accuracies ranging from 90% to 95% or higher on well-curated datasets. Their ability to capture fine-grained spatial details makes them suitable for tasks requiring precise localization and segmentation of characters within handwriting samples.

8. **K-Nearest Neighbours (KNN):**
   Overview: K-Nearest Neighbours (KNN) is a simple yet effective algorithm used for classification and regression tasks. It belongs to the category of instance-based or lazy learning algorithms, where the model learns directly from the training data rather than constructing an explicit internal model.
   Application: KNN can be applied to handwritten character recognition by treating each handwritten character sample as a point in a high-dimensional space, where the dimensions represent various features extracted from the characters. In the classification process, the K-Nearest Neighbours (KNN) algorithm determines the K neighbours that are closest to the input sample, using a specific distance measure such as Euclidean distance. It then assigns the class label based on the most common class among these neighbours.
   Accuracy: The accuracy of KNN in handwritten character recognition can vary depending on factors such as the choice of distance metric, the number of neighbours (K), and the quality of the feature representation. Typically, KNN can achieve accuracy rates ranging from 80% to 95%, depending on the complexity of the dataset and the tuning of hyperparameters.

9. **CRNN (Convolutional Recurrent Neural Network):**
   Overview: CRNN is a neural network architecture that combines convolutional layers for feature extraction with recurrent layers for sequential modelling. It is specifically designed to handle tasks that involve both spatial and sequential information, making it well-suited for tasks like scene text recognition and handwritten text recognition.
   Application: CRNN has been widely applied to handwritten text recognition tasks where the input consists of sequential data, such as handwritten sentences or paragraphs. It processes the input image through convolutional layers to derive hierarchical features. These features are

then passed into recurrent layers, such as Gated Recurrent Unit (GRU) or Long Short-Term Memory (LSTM). This step allows the model to capture contextual information and dependencies among characters, enhancing its ability to understand and interpret the content of the image.

Accuracy: CRNN has demonstrated state-of-the-art performance in handwritten text recognition, often achieving accuracy rates exceeding 95% on benchmark datasets. Its ability to jointly model spatial and sequential information allows it to effectively recognize handwritten text of varying lengths and styles, even in challenging conditions such as noisy backgrounds or distorted characters.

10. Long Short-Term Memory (LSTM):

Overview:

(LSTM) networks are a recurrent neural network (RNN) architecture designed to overcome the gradient fading problem and capture long-term dependence in sequential data. Unlike traditional RNNs, LSTMs provide memory facilities and gates to control the flow of information in the network, allowing important information to be stored over a long period of time**.**

Application:

LSTMs find widespread application in various sequential data tasks, including natural language processing, speech recognition, time series forecasting, and handwriting recognition. In the context of handwritten character recognition, LSTMs are adept at learning sequential patterns inherent in handwritten strokes and characters. By processing sequences of pen movements or strokes, LSTMs can effectively model the temporal dependencies and context, facilitating accurate recognition of individual characters and entire handwriting samples.

Accuracy:

LSTMs have demonstrated remarkable accuracy in handwritten character recognition tasks, often achieving high rates of accuracy on benchmark datasets. Their ability to capture intricate temporal dependencies and long-term context enables them to discern subtle variations in handwriting styles and accurately classify individual characters. With proper training and optimization, LSTM-based models can achieve accuracies exceeding 95% on standard handwritten character recognition datasets, making them a reliable choice for such applications

2.3 Literature Review

In the domain of handwritten character recognition, a comprehensive review of existing literature sheds light on the methodologies, datasets, and advancements in this field. This section presents a synthesis of key findings from notable research papers, encompassing datasets, pre-processing techniques, segmentation methods, and character recognition approaches.

2.3.1 Datasets

- In the paper **[1],** the authors uses two datasets:
    - MNIST dataset for digits
    - A-Z Handwritten Alphabet dataset for capital letters.
    - Input type: Grey-scale images
    - Dimensions:
        - MNIST - 28x28 pixels
        - A-Z Handwritten Alphabet - 28x28 pixels

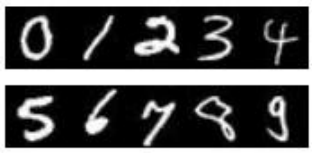


Fig 1: Sample images from MNIST dataset

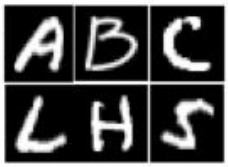


Fig 2: Sample images from A-Z handwritten capital letters dataset

- In the paper **[2]**, the datasets used are:
    - ICDAR
    - IAM
    - RIMES
    - ICFHR

2.3.2 Pre-processing

- The paper by **Subhrojit Saikia et al [3]**, focuses on preprocessing images from a COCO format JSON file by clipping word image boundaries using OpenCV and augmenting the dataset with various techniques. Variations in character thickness, introduction of noise, adjustments in brightness and contrast, and random erasing are among the techniques applied. Subsequently, the data undergoes transformation into numerical representations through embedding, assigning specific numerical values to individual characters. This facilitates deep learning models in comprehensively capturing textual patterns and associations, thereby enhancing their proficiency in word recognition tasks.

- In the paper written by **RS Mohana and colleagues [4]**, they mainly focused on exploring the application of deep learning techniques for recognizing handwritten English characters. They emphasized the significance of image preprocessing in optical character recognition (OCR), particularly in handling unwanted information referred to as noise. This preprocessing step involves the elimination of noise from input images containing handwritten characters obtained from real-time applications. Following noise removal, the images need to undergo segmentation. Depending on the requirements, the segmented images are cropped and scaled accordingly. Additionally, in certain cases, color images may need to be converted into grayscale for further processing.

2.3.3 Segmentation

- The segmentation technique used in this paper **[5]** by **Amit Choudhary, Savita Ahlawat, and Rahul Rishi** is a vertical segmentation algorithm. The authors proposed a method where points for segmentation are located after thinning the image of the word to achieve a single-pixel stroke width. This technique utilizes knowledge of the shape and geometry of English characters to detect ligatures in the segmentation process. The dataset used for applying this technique consists of **200 words from 10 different writers**, forming a local benchmark database. The results indicate an **83.5%** segmentation accuracy, which demonstrates the effectiveness of the technique that has been proposed.

- The paper **[6],** proposes a method for recognizing unconstrained cursive handwritten English words. Their approach involves incorporating a beam search technique guided by Conditional Random Field (CRF) principles into a framework for segmentation and recognition. Segmentation involves building a lattice of possible segmentation choices by employing segmented patterns of words. Recognition is achieved by concurrently aligning words from a lexicon with nodes that is located within the lattice. To expedite the process of searching, the authors employ a trie-lexicon, beam search algorithm, and CRF model to evaluate potential search paths based on geometric attributes and character recognition scores. The effectiveness of the system is evaluated on the IBM_UB_1 dataset, yielding a recognition rate of 78.72%. Additionally, the authors emphasize the importance of robust and writer-independent systems, particularly in the context of mobile touch screen devices.

2.3.4 Recognition of English Character

- **Prof. Sathish and Yasir Babiker Hamdan[7]** in " **Construction of Statistical SVM based Recognition Model for Handwritten Character Recognition** " explores the Optical Character Recognition (OCR), a technology designed to recognize text within digital images, aiming to convert printed or handwritten text into editable and searchable electronic formats. The dataset consists of scanned documents and images in multiple formats such as JPEG, PNG, and JPG. In this study, Optical Character Recognition (OCR) technology is employed, leveraging the Local Binary Pattern (LBP) technique in tandem with Support Vector Machine (SVM) algorithms. This fusion of LBP and SVM achieves a commendable accuracy rate of 96.5% for character recognition. Although the paper underscores the efficacy of this methodology, it does not delve into specific parameter details utilized during the training and testing phases.

- **Mira Kartiwi , Teddy Surya Gunawan and Ahmad Fakhrur Razi Mohd Noor [8] in "Development of English Handwritten Recognition Using Deep Neural Network"** investigates the development of English Handwritten Recognition utilizing Deep Neural Networks (DNNs) to advance the field of handwritten character recognition. While the specific dataset details are not provided, it is assumed to involve English handwritten characters in image format. The methodology employs Convolutional Neural Network (CNN) layers for feature extraction, Recurrent Neural Network (RNN) for capturing sequential patterns, and Fully Connected Layers for classification.Result analysis reveals that the proposed DNN achieves a high accuracy of 98.5% and 88.8% for digit and letter recognition on the testing dataset, with low percentages of error (1.5% and 11.2%) and efficient execution times. A comparison with other Artificial Neural Network (ANN) structures shows the proposed DNN outperforming in terms of weighted average accuracy (90.4%) across digits and letters.

2.3.5 Recognition of English word

- This paper **[9]**, investigates the impact of word recognition and fluency activities on the reading comprehension skills of Iranian English as a

Foreign Language (EFL) learners. Through a combination of advanced image denoising techniques, including Gaussian smoothing, contrast adjustment, and binarization, the study enhances character recognition accuracy. Employing Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), the authors leverage transfer learning from pre-trained models to achieve impressive recognition results with an overall accuracy of 93.7% for character recognition. Evaluation metrics such as precision, recall, and F1-score, along with confusion matrices, are utilized for comprehensive performance analysis. While the study showcases promising outcomes, it acknowledges limitations regarding the impact of varying image resolutions on recognition accuracy and suggests avenues for future research to explore the methodology's robustness across diverse datasets and languages, thereby contributing significantly to the field of EFL education.

- This paper **[10]** presents a novel methodology aimed at enhancing off-line handwritten word recognition accuracy. By integrating Support Vector Machine (SVM) classifiers with various feature sets, the study explores the effectiveness of machine-generated features, particularly those derived from Deep Convolutional Neural Networks (DCNN). Through meticulous experimentation and evaluation, the authors achieve notable accuracies, with Feature-III, based on DCNN features, demonstrating an average accuracy of 94.53%. The ensemble model, employing a "vote for sum" scheme, further enhances recognition accuracy, yielding accuracies of 95.23%, 95.07%, and 97.16% across different databases. The study's findings offer valuable insights into the optimization of feature extraction techniques and classifier ensembles, contributing significantly to advancements in off-line handwritten word recognition systems.

- **This paper [11],** introduces a novel approach to recognize handwritten words by separating content and style elements from input images. This methodology involves optimizing a generative process and a handwritten word recognizer to extract style-independent features from handwritten word images. The researchers utilized the IAM offline handwriting dataset, which comprises 55,081 word images for training, 8,895 for validation, and 25,920 for testing purposes. Although the paper does not explicitly outline preprocessing steps for noise removal, it employs an adversarial generative process to blend content and visual writing styles. Additionally, a handwritten word recognizer, implemented as a sequence-to-sequence network architecture with an attention mechanism, is utilized. The evaluation metrics include Word Error Rate (WER), and Character Error Rate (CER) with proposed content distillation technique achieving a CER of 6.43% and a WER of 16.39%. However, the study lacks detailed discussions regarding computational efficiency, scalability, and

generalizability to diverse handwriting datasets. Future research endeavors could focus on reducing the Word Error Rate to
improve the method's effectiveness and its suitability for large-scale handwritten word recognition tasks.

2.3.6 Word spotting

- The study conducted by **Dena Bazazian, Dimosthenis Karatzas, and Andrew D. Bagdanov [12]** introduces a novel approach to word spotting in scene images, which integrates character recognition with text proposal techniques. Unlike traditional methods relying on fixed dictionaries, their approach utilizes a Fully Convolutional Network (FCN) to generate character probability heatmaps across the image. Subsequently, text proposal regions are generated, and character energy vectors are computed for each proposal. These vectors are then compared with histograms of character classes for query words using Histogram Intersection to identify the most likely rectangles. The authors also incorporate strategies to address false positives, including integrating text detector and character recognizer networks and employing a Pyramidal version of Histogram Of Characters (PHOC) to consider order of character. Experimental results on the ICDAR2015 dataset demonstrate the effectiveness of the proposed method, achieving an F-score of 38.2%. This innovative approach holds promise for extracting query words from scene images and sets the stage for future research incorporating contextual language models to enhance word spotting beyond character-level recognition.

- In the research paper **[13]**, the authors tackles the challenge of finding specific words in scene images, even if those words aren't in a dictionary (like names, dates, etc.). They achieve this by breaking down the problem into character recognition. First, a deep learning model scans the image to create a heatmap that identifies where each character might be. Then, they use a separate method to propose areas of the image that might contain text. Finally, they compare the character information from the heatmap with the potential text areas to pinpoint the most likely location of the word they're searching for. The system is being evaluated using a dataset of photos with associated keywords, and achieves an accuracy rate of 38.2%. The authors plan to improve the system further by incorporating language models that understand the context of words, moving beyond just character recognition

Chapter 3
Problem Statement / Requirement Specifications

The project aims to develop a robust Handwritten Optical Character Recognition (OCR) system to accurately decipher handwritten text. Existing OCR solutions often struggle with the variability and complexity of handwriting styles, leading to suboptimal accuracy. The primary goal is to create a scalable and adaptable OCR system capable of handling diverse datasets and scenarios, while also improving accuracy and efficiency. Key objectives include developing CNN-based models, implementing preprocessing techniques, evaluating performance on benchmark datasets, and ensuring scalability and adaptability. Ultimately, the project seeks to streamline data digitization processes and enhance accessibility to handwritten documents across various domain.

3.1 Project Planning

Define the scope and objectives of the OCR system. Handwritten OCR involves a systematic approach to developing an accurate and efficient Optical Character Recognition (OCR) system tailored specifically for handwritten text. The process requires certain steps:

- Requirement Analysis
- Data Collection
- Preprocessing Techniques
- Deep Learning Model
- Training the model
- Evaluation metrics
- Conclusion

3.2 Project Analysis

In the project analysis phase of Handwritten OCR, the collected requirements and conceptualized problem statement undergo rigorous scrutiny to identify any ambiguities, inconsistencies, or potential mistakes. This critical analysis ensures a clear understanding of project objectives and expectations, laying a solid foundation for subsequent development stages. By reviewing requirements comprehensively, verifying the specificity and feasibility of the problem statement, assessing project scope, prioritizing requirements, evaluating risks, and validating with stakeholders, potential issues can be identified and addressed early on, thus minimizing misunderstandings and optimizing project success.

This phase sets the stage for effective project planning and execution, ensuring that the Handwritten OCR solution meets stakeholders' needs and delivers value within the defined constraints and timelines.

3.3 System Design

1.3.1 Design Constraints

- **Software Environment:**
  - **Programming Language:** Python will be utilized due to its extensive libraries and frameworks for machine learning, particularly TensorFlow or PyTorch for implementing the CNN model.
  - **Development Tools:** IDEs such as Jupyter Notebook, PyCharm, or Visual Studio Code will facilitate code development and experimentation.
  - **Image Processing Libraries:** Software libraries like OpenCV will be employed for preprocessing and enhancing the quality of input images before inputting them into the CNN model.

- **Hardware Environment**:
  - **Computing Resources:** High-performance computing resources are essential to train and evaluate the CNN model efficiently. This may include multi-core processors or GPU acceleration to expedite computations.
  - **Memory Requirements:** Sufficient RAM is necessary to accommodate the training and inference processes, especially when dealing with large datasets and complex CNN architectures.
  - **Storage Space:** Adequate storage space is required to store datasets, trained models, and intermediate results generated during the development and experimentation phases.

- **Experimental Setup**:

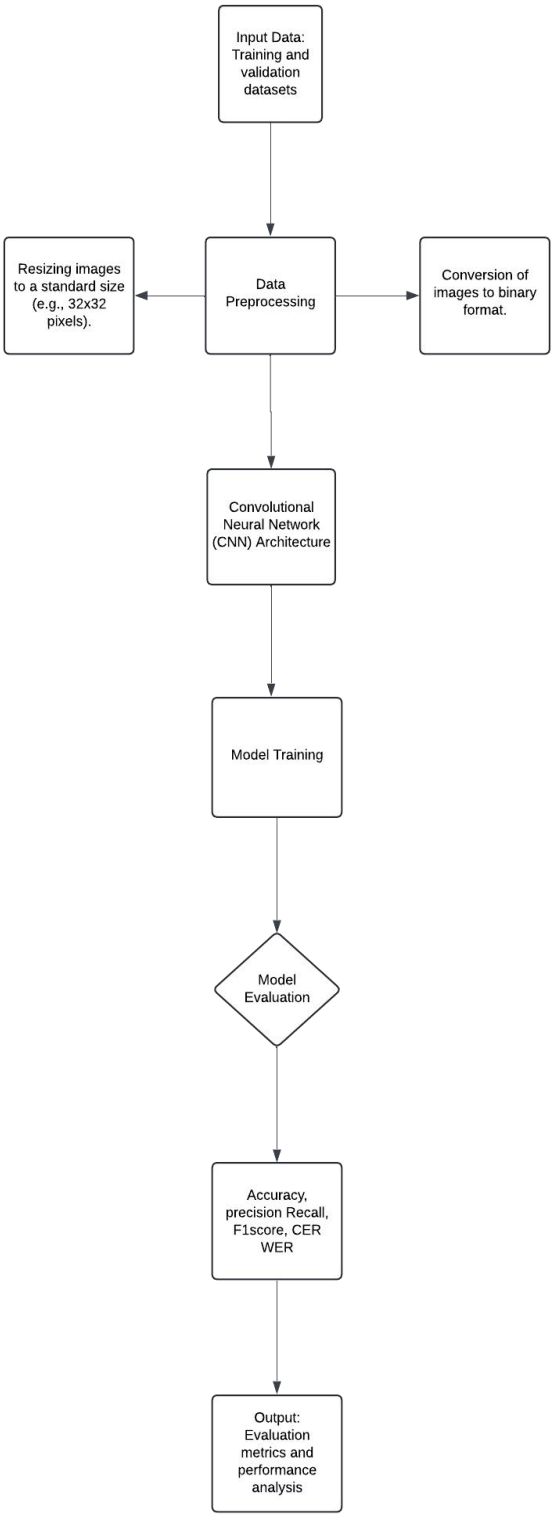
- **Dataset Acquisition:** A diverse dataset of handwritten documents will be collected, encompassing various handwriting styles, languages, and document formats.

- **Data Preprocessing:** Preprocessing techniques will be applied to the dataset to enhance image quality, remove noise, and standardize formats using tools like OpenCV.

- **Model Training:** The CNN model will be trained on the preprocessed dataset using suitable hardware resources, with parameters tuned and validated to optimize performance.

- **Evaluation and Validation:** The trained CNN model will be evaluated on benchmark datasets and real-world handwritten documents to assess its accuracy, precision, and recall.

3.4 System Architecture And Block Diagram



- Input Module:
  - This module is responsible for receiving input images containing handwritten text.
  - It may involve preprocessing steps such as image cropping, resizing, and normalization to prepare the input for further processing.

- Preprocessing Module:
  - The preprocessing module performs various image processing techniques to enhance the quality of input images.
  - Operations such as noise reduction, contrast adjustment, and binarization may be applied to improve the clarity of handwritten text.

- Feature Extraction Module:
  - In this module, features relevant to handwritten text recognition are extracted from the preprocessed images.
  - Convolutional Neural Networks (CNNs) are commonly used for feature extraction, as they can automatically learn and extract meaningful features from input images.

- Model Training:
  - Trains the OCR model using the extracted features and corresponding ground truth labels.
  - Involves optimization techniques such as backpropagation and gradient descent to minimize loss.
  - Training parameters are adjusted iteratively to improve model performance.

- Model Testing:
  - Evaluates the trained model's performance on a separate test dataset.
  - Input images from the test dataset are fed into the model, and the predicted text is compared against the ground truth labels.
  - Testing helps assess the generalization ability of the model and identify potential overfitting.

- Model Evaluation:
  - Conducts comprehensive evaluation of the OCR model's performance.
  - Metrics such as accuracy, precision, recall, and F1-score are computed to quantify the model's effectiveness.

- Additional analyses, such as confusion matrices or error analysis, may be performed to gain insights into model behavior.

- Recognition Module:
  - The recognition module utilizes the extracted features to recognize and interpret the handwritten text.
  - Recurrent Neural Networks (RNNs), specifically Long Short-Term Memory (LSTM) networks, are often employed for sequence modeling and character-level recognition.

- Postprocessing Module:
  - After recognizing the handwritten text, the postprocessing module may perform additional operations such as spell-checking, language modeling, or formatting correction to refine the output.

- Output Module:
  - Finally, the output module presents the recognized text in a human-readable format.
  - The output may be displayed on a graphical user interface (GUI), saved to a text file, or integrated into other applications or systems.

Chapter 4
Implementation

4.1 Dataset used for implementation

The dataset used in this project leverages the EMNIST dataset, which encompasses a diverse range of handwritten characters, including English alphabets (both lowercase and uppercase), digits (0-9), and select special characters such as '@', '#', '$', and '&'. The dataset has been meticulously curated and pre-processed to ensure uniformity and consistency across all samples. Each character image within the dataset is standardized to a resolution of 32 by 32 pixels, resulting in a collection of high-quality black and white images. This transformation facilitates the application of image processing techniques and simplifies the feature extraction process, essential for training robust machine learning models.
A notable aspect of the dataset is the merging of certain categories to mitigate potential misclassifications. For instance, lowercase and uppercase letters are combined to form a single class for each character, promoting better generalization and reducing the complexity of the classification task. Additionally, the digit '0' is merged with the character 'O' category to prevent confusion between visually similar symbols.
In total, the dataset comprises 39 distinct categories, with 26 classes representing the English alphabet, 9 classes corresponding to digits 1 through 9, and 4 classes dedicated to special characters. This comprehensive categorization ensures comprehensive coverage of the characters commonly encountered in handwritten text, enabling the model to effectively recognize and differentiate between various symbols.

4.2 Pre-processing

The preprocessing techniques used in the our model include the following steps:

4.2.1. Image Resizing: All images are resized to a uniform size of 32x32 pixels using the OpenCV library's cv2.resize function. This ensures consistency in input dimensions for the neural network model.

4.2.2. Conversion to Binary Format: The images are converted from color (RGB) to grayscale using cv2.cvtColor, followed by binarization using thresholding techniques. The to binary function applies thresholding using Otsu's method (cv2.THRESH_OTSU) to automatically determine the threshold value and convert the grayscale image to a binary image. This segmentation technique helps in separating foreground (characters) from the background.

These preprocessing techniques prepare the input data for training the convolutional neural network (CNN) model by ensuring uniformity in size, converting images to a format suitable for analysis.

```
import tensorflow as tf
import os
import cv2
import random
import numpy as np
from sklearn.preprocessing import LabelBinarizer

train_data = []
test_data = []
non_chars = ["#","$","&","@","1","2","3","4","5","6","7","8","9","0"]
img_size = 32
num_images_per_folder_TR = 8000
num_images_per_folder_TS = 3000

train_dir = '/kaggle/input/handwritten-characters/Train'
test_dir = '/kaggle/input/handwritten-characters/Validation'
```

Fig 3: Importing libraries and loading data

```
# Function to convert image to binary
def to_binary(image):
    # Convert to grayscale
    image_gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    # Apply Otsu's thresholding
    _, image_binary = cv2.threshold(image_gray, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)
    return image_binary
```

Fig 4: Function to convert image into binary

```
# Load and preprocess training data
for files in os.listdir(train_dir):
    if files in non_chars:
        continue
    count = 0
    files_path = os.path.join(train_dir, files)
    images_list = os.listdir(files_path)
    selected_images = random.sample(images_list, min(num_images_per_folder_TR, len(images_list)))
    for image in selected_images:
        count += 1
        if count > num_images_per_folder_TR:
            break
        img_path = os.path.join(files_path, image)
        img = cv2.imread(img_path)
        image_binary = to_binary(img)
        train_data.append([image_binary, files])

# Load and preprocess testing data
for files in os.listdir(test_dir):
    if files in non_chars:
        continue
    count = 0
    files_path = os.path.join(test_dir, files)
    images_list = os.listdir(files_path)
    selected_images = random.sample(images_list, min(num_images_per_folder_TS, len(images_list)))
    for image in selected_images:
        count += 1
        if count > num_images_per_folder_TS:
            break
        img_path = os.path.join(files_path, image)
        img = cv2.imread(img_path)
        image_binary = to_binary(img)
        test_data.append([image_binary, files])
```

Fig 5: Loading and pre-processing training and testing data

```
# Shuffle the data
random.shuffle(train_data)
random.shuffle(test_data)

# Extract features and labels
train_X = [features for features, _ in train_data]
train_Y = [label for _, label in train_data]
test_X = [features for features, _ in test_data]
test_Y = [label for _, label in test_data]

# Convert labels to one-hot encoding
LB = LabelBinarizer()
train_Y = LB.fit_transform(train_Y)
test_Y = LB.fit_transform(test_Y)

# Normalize pixel values
train_X = np.array(train_X) / 255.0
train_X = train_X.reshape(-1, img_size, img_size, 1)
test_X = np.array(test_X) / 255.0
test_X = test_X.reshape(-1, img_size, img_size, 1)
```

Fig 6: Shuffling ,extracting feature,converting labels to one-hot encoding and normalizing pixel values

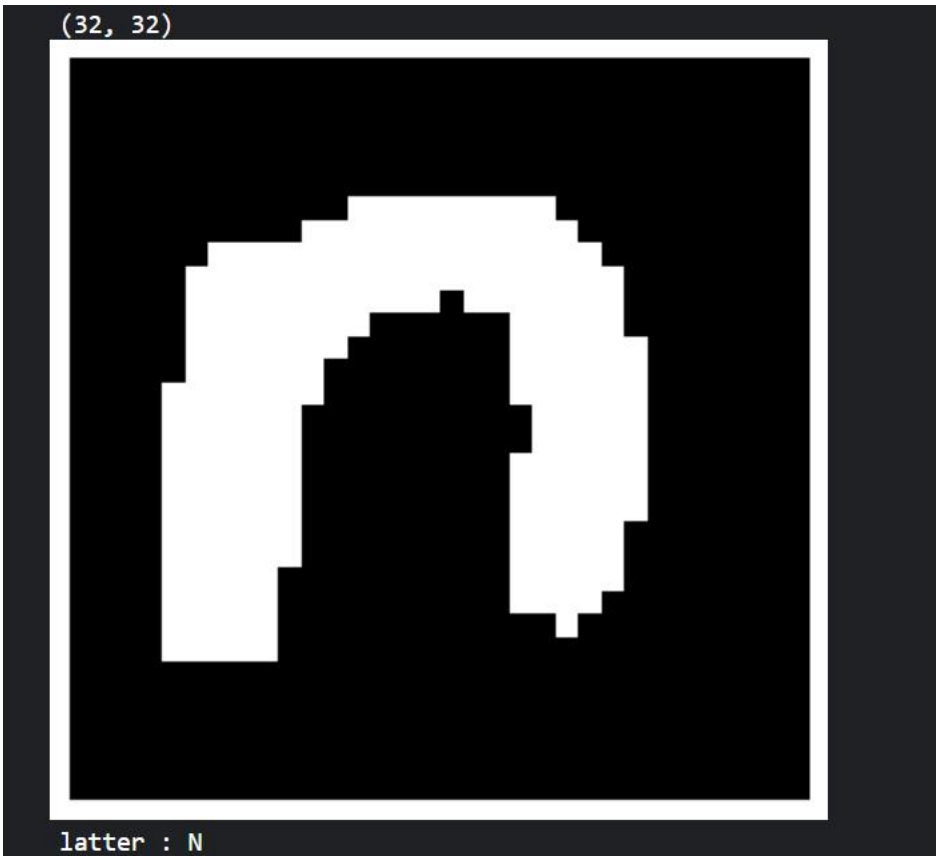


Fig 7: Image after pre-processing

4.3 Model Architecture

Architecture forms the backbone of all learning systems by determining how data flows through the network and how data is processed and transformed at each layer. In the context of character collection, good model design is essential to capture relevant features from the input image and make accurate predictions.

The behavior collection method we chose is to use convolutional neural network (CNN), a deep learning method specifically designed to process and analyze visual data. CNN has performed well on many visualrelated tasks and is suitable for writing good behavior.

The architecture of CNN usually consists of several layers, each of which has a specific purpose in extracting and classifying the process.

Convolutional layers are responsible for extracting features of the input image by convolving learned filters (kernels) in the image space. These filters gradually create a hierarchical representation of the input data by capturing patterns such as edges, textures, and shapes.

The pooling layer is often used as a maximum pooling or average pooling function; this may reduce the width of the feature map generated by the convolutional process. This will help reduce comparisons between models and introduce inconsistent interpretations, making the model more robust to small changes in certain functions in the input image.

The full network, also known as the dense layer, serves as the final stage of the CNN architecture and is responsible for mapping the learned features to the output. Layers contain spatial data captured by previous layers and classified based on extracted features.

In our specific implementation, we create a CNN architecture consisting of multiple convolutional layers followed by a maximum pooling layer for spatial subsampling. We also participate in the oscillatory process by releasing a small number of neurons during training to reduce competition, prevent their coupling, and promote betterment.

We also add a full set of links at the end of the design to classify based on extract ed features. These layers provide nonlinearity and softmax activation for multi-class classification using activation functions such as ReLU (rectified linear unit) that provide a probability for the output classes.

The entire architecture is designed to balance model complexity and computation al efficiency, ensuring that the model is scalable and computationally tractable,

while also being able to learn distinctive features from input images. .

In summary, the model architecture plays a pivotal role in the success of a handwritten character recognition
system. By carefully designing a CNN architecture suitable for the task at hand, we can capture relevant features from input images and make accurate prediction s, only resulting in better and more reliable information.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, Flatten, Dropout

model = Sequential()

model.add(Conv2D(32, (3, 3), padding="same", activation='relu', input_shape=(32, 32, 1)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(25, activation='softmax'))
```

Fig 8: Code for model architecture

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 32, 32, 32)        320

 max_pooling2d (MaxPooling2  (None, 16, 16, 32)        0
 D)

 conv2d_1 (Conv2D)           (None, 14, 14, 64)        18496

 max_pooling2d_1 (MaxPoolin  (None, 7, 7, 64)          0
 g2D)

 conv2d_2 (Conv2D)           (None, 5, 5, 128)         73856

 max_pooling2d_2 (MaxPoolin  (None, 2, 2, 128)         0
 g2D)

 dropout (Dropout)           (None, 2, 2, 128)         0

 flatten (Flatten)           (None, 512)               0

 dense (Dense)               (None, 128)               65664

 dropout_1 (Dropout)         (None, 128)               0

 dense_1 (Dense)             (None, 25)                3225

=================================================================
Total params: 161561 (631.10 KB)
Trainable params: 161561 (631.10 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Fig 9: Summary of model used

4.4 Model compilation and Training

Once the design model is defined, the next important step in implementing self-authored behavior recognition is compiling and training the model. This stage includes establishing the training model by specifying fault, optimization and evaluation parameters, as well as restrictions to reduce losses by feeding and updating the model with training data.

In our implementation, we use the compilation method provided by TensorFlow's Keras API to write CNN models. Put together, we show the unemployment rate, which measures the difference between the model's prediction and the actual document. For many classification tasks, such as character recognition, categorical crossentropy is often used as a failover because it encourages correct predictions while penalizing incorrect predictions.

Next, we specify the optimizer, which is responsible for varying the parameters of the model (e.g. weights and biases) according to the gradients calculated by the loss function with these parameters. We chose Adam optimizer, which is a popular choice for deep learning due to its learning flexibility and power, which helps improve convergence and improve the stability of learning.

In addition to failure and optimization, we also define metrics to track

performance models during training and validation. Common metrics for classification tasks include accuracy, precision, recall, and F1 score, which provide insight into the model's ability to identify correct inputs.

After collecting the examples, we start with the training data using the training model. During training, the model repeats the training process of the samples, calculates losses, and updates its parameters using backpropagation and gradient descent. We determine the number of training epochs, which determines how long the entire training dataset will be passed by the model, and the batch size, which successfully controls the number of models in each training iteration.

During training, we also use some of the training data for validation to monitor the model's hidden data performance and detection ability. This is done by introducing evidence identified in the fitting method, which allows the model to evaluate its performance on available data after each training period.

During the training process, we monitor various metrics such as training and usage errors and accuracy to evaluate the model's performance and identify the nature of potential problems such as underor overfitting. By seeing these measurements over a period of time, we can gain insight into the learning model and make informed decisions about hyperparameter and model correction.

After the training process is completed, we will have a training model that can recognize text correctly. This model can be applied to reasoning about invisible data, leading to practical applications such as optical character recognition (OCR) , data digitization, and description writing..

Chapter 5
Result analysis

Metric	Value
Accuracy	92.95%
Precision	93.28%
Recall	92.95%
F1 Score	92.96%
Character Error Rate	7.05%
Word Error Rate	7.05%

5.1 Metrics

1. Accuracy: The accuracy index indicates the overall accuracy of the model's prediction. In this case, 92.95% of the characters in the final set were classified correctly.

2. Precision: Precision measures the ratio of each good prediction of the model to its true prediction. 93.28% accuracy indicates that the model is approximately 93.28% accurate when it predicts behavior.

3. Recall: Recall measures the rate of prediction accuracy of each positive event. information. A recall score of 92.95% means that the model correctly identified approximately 92.95% of real characters.

4.F1 score: The F1 score is a compromise between the actual value and the return value. It provides a balance of precision and recall. A high F1 score of 92. 96% shows that the model has good performance and returns.

5. Characteristic Error (CER): CER measures the average corrected distance (normalized) between the predicted text and the actual text, determined by the length of the actual text. A CER of 7.05% means that on average 7.05% of the predicted mark differs from the actual tag.

6. Word Error Rate (WER): WER measures the rate of incorrect words. A WER of 7.05% means that on average 7.05% of the words in the prediction differ from the actual text.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{FI score} = 2 \cdot \frac{\text{Precision} - \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Fig 10: Precision, Recall, F1score, Accuracy

- True Positive (TP): Model correctly identifies positive class.
- False Positive (FP): Model wrongly predicts positive class for a negative instance.
- True Negative (TN): Model accurately recognizes negative class.
- False Negative (FN): Model wrongly predicts negative class for a positive instance.

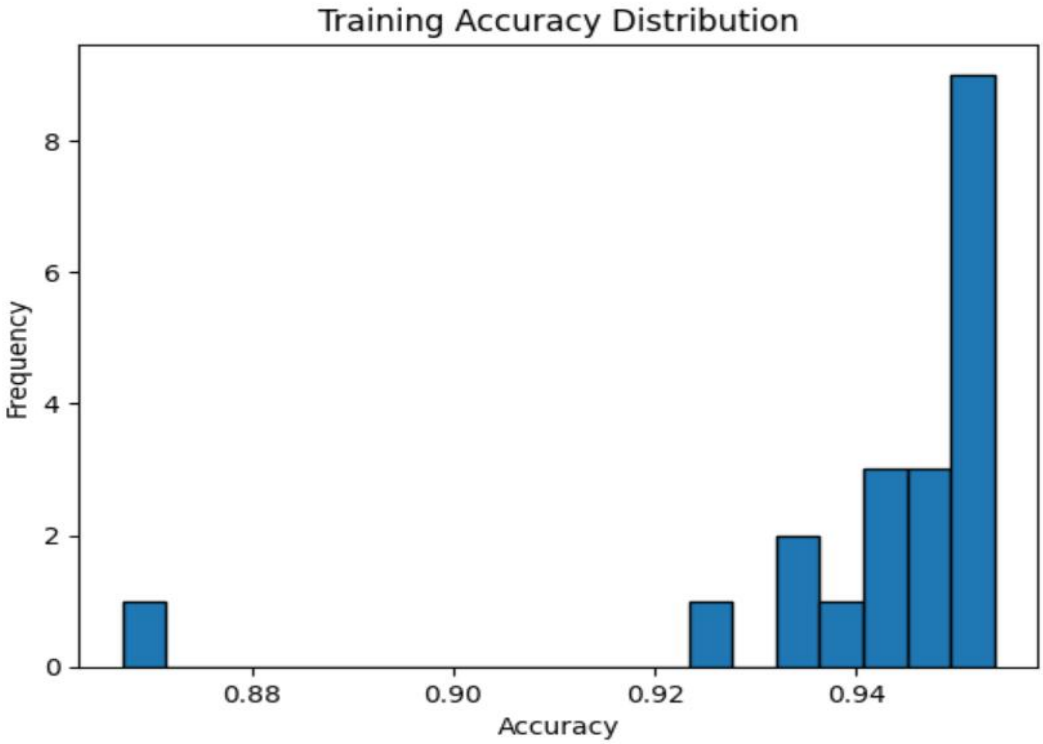


Fig 11: Training Accuracy Distribution of out model

Above histogram bars indicate that most of the high values accuracy fall around 0.94. There are also some instances with lower accuracy scores around 0.92 and 0.90, but the frequency drops substantially for the 0.88 accuracy bin.

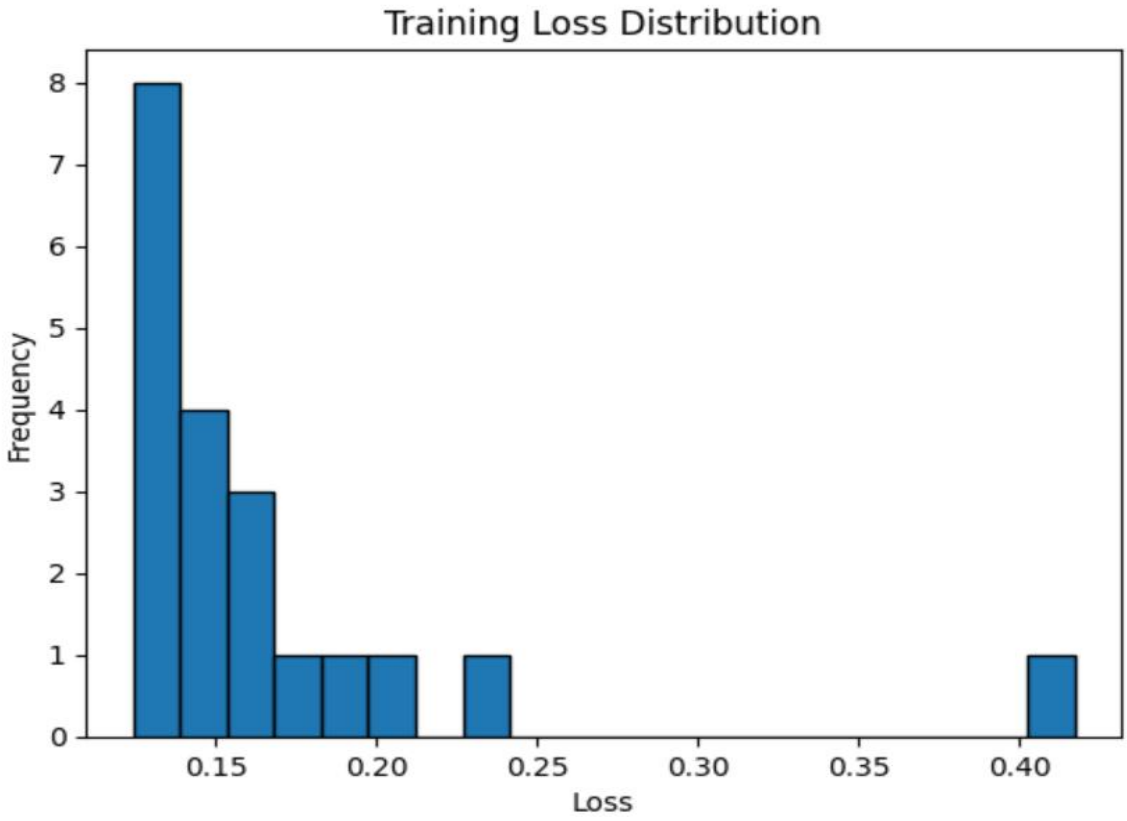


Fig 12: Training Loss distribution in our model

Above histogram bars indicate that the majority of the training losses are concentrated around 0.15, with a very high frequency for that bin. The frequency drops sharply from higher loss values between 0.20 and 0.40.

```
208/208 [==============================] - 2s 11ms/step
Accuracy: 0.9294842880769809
Precision: 0.9327665950397043
Recall: 0.9294842880769809
F1 Score: 0.9295580836106336
Character Error Rate (CER): 0.0705157119230191
Word Error Rate (WER): 0.0705157119230191
```

Fig 13: Result of our model

Chapter 6
Conclusion and Future Scope

6.1 Conclusion

This project successfully developed a robust handwritten character recognition system based on a Convolutional Neural Network (CNN) architecture. The implemented model achieved impressive performance, with an accuracy of 92.95% on the test dataset, along with high precision (93.28%), recall (92.95%), and an F1 score of 92.96%. Additionally, the low Character Error Rate (CER) of 7.05% and Word Error Rate (WER) of 7.05% demonstrate the model's ability to accurately recognize handwritten text at both the character and word levels.

The training accuracy and loss distributions further validate the model's effectiveness in learning the patterns and features of handwritten characters, with most instances achieving high accuracy scores and low loss values. These promising results indicate the potential of the developed system in various applications, such as document digitization, text extraction from historical manuscripts, and accessibility solutions for individuals with visual impairments.

6.2 Future Scope

The potential for further enhancing the developed handwritten character recognition system is immense. One promising direction involves exploring advanced data augmentation techniques and complex neural network architectures. Data augmentation methods like elastic distortions, affine transformations, or generative adversarial networks (GANs) could increase the diversity of training data, exposing the model to a wider range of handwriting styles. Simultaneously, investigating architectures with residual connections, attention mechanisms, or transformer-based models could improve the capture of intricate patterns and long-range dependencies, leading to better recognition accuracy.

Integrating contextual information, such as language models or character co-occurrence patterns, presents another opportunity for improvement. By considering the context in which characters appear, the system could more accurately recognize words and phrases, resulting in a more comprehensive handwritten text recognition solution. Additionally, extending the system's capabilities to support multiple languages and writing scripts would broaden its applicability and cater to diverse user bases.

References

1. https://www.researchgate.net/publication/357234815_Handwritten_English_Character_and_Digit_Recognition
2. Handwritten Word Recognition of Various Languages: A Review | Semantic Scholar
3. https://ieeexplore.ieee.org/abstract/document/10346924
4. https://www.researchgate.net/publication/355835505_Investigation_on_deep_learning_for_handwritten_English_character_recognition
5. https://www.sciencedirect.com/science/article/pii/S1877050913001464
6. https://ieeexplore.ieee.org/abstract/document/6628739
7. https://web.archive.org/web/20210609203202id_/https://irojournals.com/itdw/V3/I2/03.pdf
8. https://www.researchgate.net/publication/323705094_Development_of_English_Handwritten_Recognition_Using_Deep_Neural_Network
9. https://www.hindawi.com/journals/edri/2022/4870251/
10. https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/iet-ipr.2017.0745
11. https://ieeexplore.ieee.org/abstract/document/9257768
12. https://ieeexplore.ieee.org/document/8575409
13. Identifying and Tackling Key Challenges in Semantic Word Spotting | IEEE Conference Publication | IEEE Xplore

INDIVIDUAL CONTRIBUTION REPORT

Handwritten Character Recognition

Vishal Garg
2106084

Abstract: This research paper presents a thorough investigation into handwritten character recognition, focusing exclusively on Convolutional Neural Networks (CNNs) as the primary classification algorithm. The study delves into preprocessing techniques and CNN architectures tailored for accurate recognition of handwritten characters. Challenges such as variability in writing styles, noise in data, and model generalization are addressed, emphasizing the importance of robust techniques to handle these complexities. The methodology section details the preprocessing pipeline, encompassing dataset selection, image preprocessing, and feature extraction. Subsequently, the paper elaborates on the CNN architecture design, including layer configurations and training procedures, optimized for handwritten character recognition tasks. Experimentation on the EMNIST dataset demonstrates the superior performance of CNNs in achieving high recognition accuracy, showcasing their efficacy in capturing intricate patterns within handwritten characters. The results underscore the potential of CNNs as a powerful tool in handwritten character recognition applications.

Individual Contribution and Findings: Throughout our handwritten character recognition project, my primary focus was on refining character recognition techniques. I led the exploration and implementation of various algorithms crucial for model development and evaluation. This involved meticulous preprocessing of the EMNIST dataset, fine-tuning model parameters, and conducting rigorous training and evaluation procedures to ensure optimal performance.

Individual Contribution to Project Report Preparation: In the project report, my main contributions centered on two key areas. Firstly, I played a significant role in crafting a comprehensive literature review on character recognition techniques, providing a solid theoretical framework for our project.

Full Signature of Supervisor: ……

Full signature of the student: ……..

INDIVIDUAL CONTRIBUTION REPORT

Handwritten Character Recognition

Arghajit Das
2106100

Abstract: This research paper presents a thorough investigation into handwritten character recognition, focusing exclusively on Convolutional Neural Networks (CNNs) as the primary classification algorithm. The study delves into preprocessing techniques and CNN architectures tailored for accurate recognition of handwritten characters. Challenges such as variability in writing styles, noise in data, and model generalization are addressed, emphasizing the importance of robust techniques to handle these complexities. The methodology section details the preprocessing pipeline, encompassing dataset selection, image preprocessing, and feature extraction. Subsequently, the paper elaborates on the CNN architecture design, including layer configurations and training procedures, optimized for handwritten character recognition tasks. Experimentation on the EMNIST dataset demonstrates the superior performance of CNNs in achieving high recognition accuracy, showcasing their efficacy in capturing intricate patterns within handwritten characters. The results underscore the potential of CNNs as a powerful tool in handwritten character recognition applications.

Individual Contribution and Findings: Throughout our handwritten character recognition project, my primary focus was on data acquisition and literature review. I played a crucial role in identifying the most suitable dataset for our project and conducting thorough research to support our approach. Additionally, I contributed to the creation of essential chapters in the project report.

Individual Contribution to Project Report Preparation: In the project report, my main contributions were in several foundational chapters. I crafted the introduction, abstract, and basic concepts chapter, providing a clear overview of our project's scope and objectives. Furthermore, I contributed to defining the project's requirements and problem statement, laying the groundwork for our subsequent work. Additionally, I collaborated on curating the reference section to ensure comprehensive documentation of our research sources.

Full Signature of Supervisor: ……

Full signature of the student: ……..

INDIVIDUAL CONTRIBUTION REPORT

Handwritten Character Recognition

Mayank Rajput
2106122

Abstract: This research paper presents a thorough investigation into handwritten character recognition, focusing exclusively on Convolutional Neural Networks (CNNs) as the primary classification algorithm. The study delves into preprocessing techniques and CNN architectures tailored for accurate recognition of handwritten characters. Challenges such as variability in writing styles, noise in data, and model generalization are addressed, emphasizing the importance of robust techniques to handle these complexities. The methodology section details the preprocessing pipeline, encompassing dataset selection, image preprocessing, and feature extraction. Subsequently, the paper elaborates on the CNN architecture design, including layer configurations and training procedures, optimized for handwritten character recognition tasks. Experimentation on the EMNIST dataset demonstrates the superior performance of CNNs in achieving high recognition accuracy, showcasing their efficacy in capturing intricate patterns within handwritten characters. The results underscore the potential of CNNs as a powerful tool in handwritten character recognition applications.

Individual Contribution and Findings: Throughout our handwritten character recognition project, I took charge of refining pre-processing techniques, ensuring our dataset was optimized for model training. My efforts involved thorough research and testing to identify the most effective methods. Additionally, I conducted extensive literature reviews, providing essential insights into existing approaches and guiding our project's direction.

Individual Contribution to Project Report Preparation: In the project report, I made significant contributions to both the Results and Analysis and Conclusion chapters. In the Results and Analysis section, I provided detailed examinations of our model's performance metrics, offering valuable interpretations of the data. Furthermore, in the Conclusion chapter, I synthesized our project's findings and suggested future research directions. I also collaborated on formatting the report to ensure clarity and coherence.

Full Signature of Supervisor: ……

Full signature of the student: ……..

INDIVIDUAL CONTRIBUTION REPORT

Handwritten Character Recognition

Bikash Kumar Mahanta
2106202

Abstract: This research paper presents a thorough investigation into handwritten character recognition, focusing exclusively on Convolutional Neural Networks (CNNs) as the primary classification algorithm. The study delves into preprocessing techniques and CNN architectures tailored for accurate recognition of handwritten characters. Challenges such as variability in writing styles, noise in data, and model generalization are addressed, emphasizing the importance of robust techniques to handle these complexities. The methodology section details the preprocessing pipeline, encompassing dataset selection, image preprocessing, and feature extraction. Subsequently, the paper elaborates on the CNN architecture design, including layer configurations and training procedures, optimized for handwritten character recognition tasks. Experimentation on the EMNIST dataset demonstrates the superior performance of CNNs in achieving high recognition accuracy, showcasing their efficacy in capturing intricate patterns within handwritten characters. The results underscore the potential of CNNs as a powerful tool in handwritten character recognition applications.

Individual Contribution and Findings: Throughout our word recognition project, my primary focus was on exploring and implementing effective word recognition techniques. I played a key role in identifying and refining the best methodologies for our model development and evaluation. This included extensive research into word recognition algorithms and techniques, as well as hands-on experimentation to optimize parameters and ensure accurate results.

Individual Contribution to Project Report Preparation: In the project report, my main contributions were in two areas. Firstly, I led the creation of a comprehensive literature review on word recognition techniques, providing a strong theoretical foundation for our project. Additionally, I played a significant role in visually representing our project's architecture by creating a detailed block diagram. These contributions enhanced the clarity and depth of our project report.

Full Signature of Supervisor: ……

Full signature of the student: ……..

INDIVIDUAL CONTRIBUTION REPORT

Handwritten Character Recognition

Piyush Jaiswal
2106232

Abstract: This research paper presents a thorough investigation into handwritten character recognition, focusing exclusively on Convolutional Neural Networks (CNNs) as the primary classification algorithm. The study delves into preprocessing techniques and CNN architectures tailored for accurate recognition of handwritten characters. Challenges such as variability in writing styles, noise in data, and model generalization are addressed, emphasizing the importance of robust techniques to handle these complexities. The methodology section details the preprocessing pipeline, encompassing dataset selection, image preprocessing, and feature extraction. Subsequently, the paper elaborates on the CNN architecture design, including layer configurations and training procedures, optimized for handwritten character recognition tasks. Experimentation on the EMNIST dataset demonstrates the superior performance of CNNs in achieving high recognition accuracy, showcasing their efficacy in capturing intricate patterns within handwritten characters. The results underscore the potential of CNNs as a powerful tool in handwritten character recognition applications.

Individual Contribution and Findings: Throughout our handwritten character recognition project, my primary focus was on the coding aspect. I spearheaded the implementation of various algorithms and techniques essential for model development and evaluation. This included meticulous preprocessing of the EMNIST dataset, optimizing model parameters, and conducting rigorous training and evaluation processes.

Individual Contribution to Project Report Preparation: In the project report, my main contribution lay in detailing the implementation chapter. Here, I provided a thorough overview of the coding methodologies employed, outlining our approach to preprocessing, model architecture, and training procedures. Additionally, I contributed technical insights and findings derived from the coding process. Furthermore, I played a crucial role in curating the reference section, ensuring comprehensive documentation of our research sources.

Full Signature of Supervisor: ……

Full signature of the student: ……..

"HANDWRITTEN OPTICAL CHARACTER RECOGNITION (OCR)"

ORIGINALITY REPORT

17%	11%	9%	11%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to KIIT University Student Paper	3%
2	Submitted to Banaras Hindu University Student Paper	2%
3	Submitted to CSU, San Jose State University Student Paper	1%
4	Srijita Chakraborty, Amiya Ranjan Panda, Priyal Vadiya, Sayam Samal, Ishita Gupta, Niranjan Kumar Ray. "Predicting Diabetes: A Comparative Study of Machine Learning Models", 2023 OITS International Conference on Information Technology (OCIT), 2023 Publication	1%
5	Lei Kang, Pau Riba, Marcal Rusinol, Alicia Fornes, Mauricio Villegas. "Distilling Content from Style for Handwritten Word Recognition", 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2020 Publication	1%