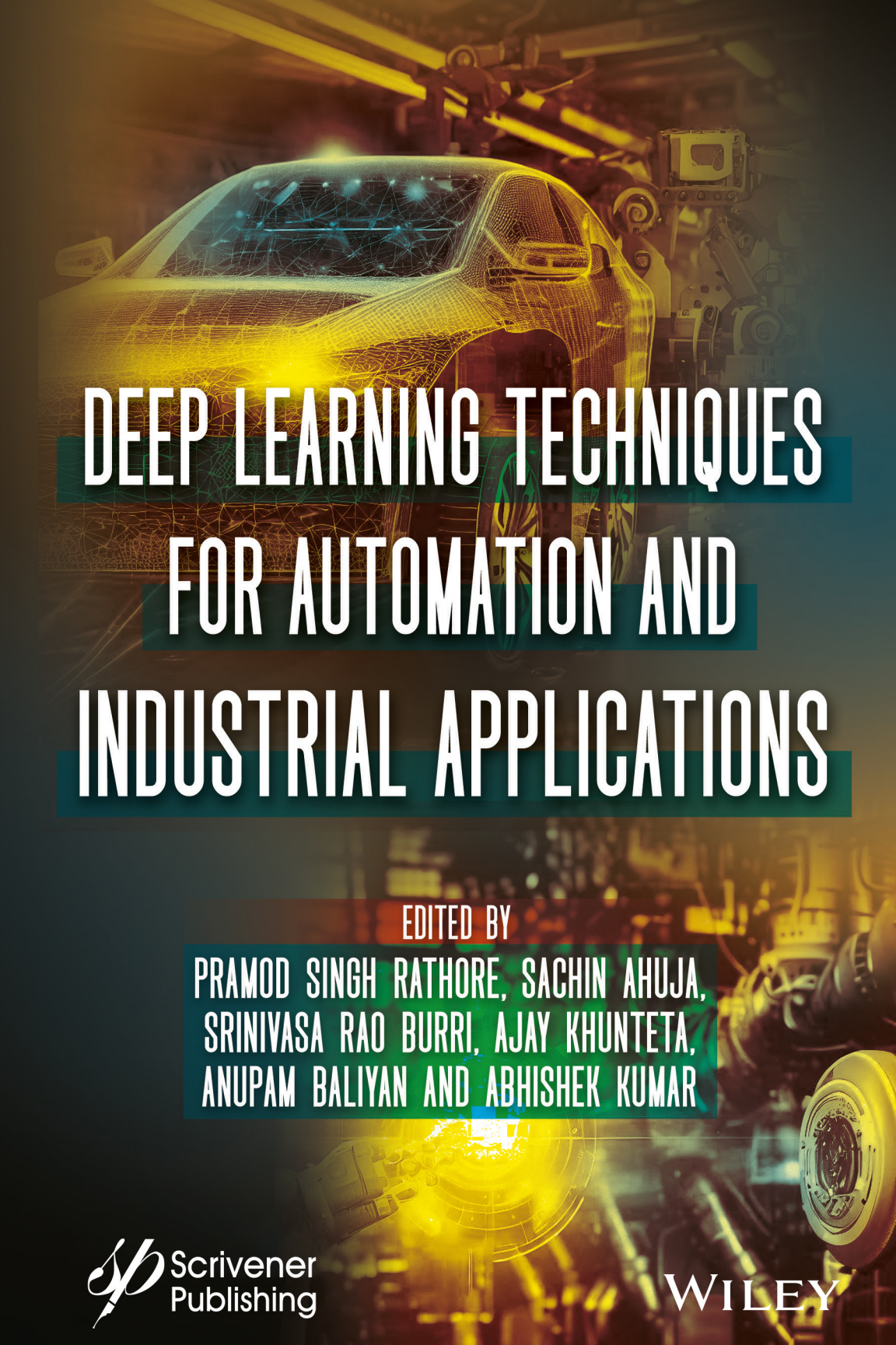
DEEP LEARNING TECHNIQUES
FOR AUTOMATION AND
INDUSTRIAL APPLICATIONS
EDITED BY
PRAMOD SINGH RATHORE, SACHIN AHUJA,
SRINIVASA RAO BURRI, AJAY KHUNTETA,
ANUPAM BALIYAN AND ABHISHEK KUMAR
Scrivener
Publishing
WILEY

Deep Learning Techniques for Automation and Industrial Applications

Scrivener Publishing
100 Cummings Center, Suite 541J
Beverly, MA 01915-6106

Publishers at Scrivener
Martin Scrivener (martin@scrivenerpublishing.com)
Phillip Carmical (pcarmical@scrivenerpublishing.com)

Deep Learning Techniques for Automation and Industrial Applications

Edited by
Pramod Singh Rathore
Sachin Ahuja
Srinivasa Rao Burri
Ajay Khunteta
Anupam Baliyan
and
Abhishek Kumar

WILEY

This edition first published 2024 by John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, USA and Scrivener Publishing LLC, 100 Cummings Center, Suite 541J, Beverly, MA 01915, USA

For more information about Scrivener publications please visit www.scrivenerpublishing.com.

Wiley Global Headquarters
111 River Street, Hoboken, NJ 07030, USA

For details of our global editorial offices, customer services, and more information about Wiley products visit us at www.wiley.com.

Library of Congress Cataloging-in-Publication Data

ISBN 978-1-394-23424-0

Cover image: Pixabay.Com
Cover design by Russell Richardson

Set in size of 11pt and Minion Pro by Manila Typesetting Company, Makati, Philippines

Printed in the USA

10 9 8 7 6 5 4 3 2 1

Contents

Preface

Artificial intelligence learning is the fastest-growing field in computer science. Deep learning algorithms and techniques are found to be useful in different areas, such as automatic machine translation, automatic handwriting generation, visual recognition, fraud detection, and detecting developmental delays in children. *Deep Learning Techniques for Automation and Industrial Applications* presents a concise introduction to the recent advances in the field of artificial intelligence (AI). The broad-ranging discussion herein covers the algorithms and applications in the areas of AI, reasoning, machine learning, neural networks, reinforcement learning, and their applications in various domains like agriculture and healthcare. Applying deep learning techniques or algorithms successfully in these areas requires a concerted effort, fostering integrative research between experts ranging from diverse disciplines, from data science to visualization.

This book provides state-of-the-art approaches to deep learning in these areas. It covers detection and prediction, as well as future framework development, building service systems, and analytical aspects. For all these topics, various approaches to deep learning, such as artificial neural networks, fuzzy logic, genetic algorithms, and hybrid mechanisms, are explained.

The successful application of deep learning techniques to enable meaningful, cost-effective, personalized cloud security service is a primary and current goal. However, realizing this goal requires effective understanding, application, and amalgamation of deep learning and several other computing technologies to deploy such a system effectively. This book helps to clarify certain key mechanisms of technology to realize a successful system. It enables the processing of very large datasets to help with precise and comprehensive forecasts of risk and delivers recommended actions that improve outcomes for consumers. This is a novel application domain of deep learning and is of prime importance to all of human civilization.

Preparing both undergraduate and graduate students for advanced modeling and simulation courses, this book helps them to carry out effective simulation studies. In addition, graduate students will be able to

comprehend and conduct AI and data mining research after completing this book.

The book comprises fourteen chapters. In Chapter 1, images play a crucial role in describing, representing, and conveying information, which aids in human productivity, cost, analysis, and other areas. Text extraction is the method used to convert text to plain text. Text extraction is a very challenging problem because of the many changes in these texts' size, orientation, and alignment, as well as low-resolution/pixelated images, noisy backgrounds, etc. Using the Tesseract OCR engine, we aim to reduce these issues in this project. Tesseract is developed by Google and its an open-source optical character recognition (OCR) engine. OCR technology allows computers to recognize text in images, making it possible to convert images of text into machine-readable text. Tesseract has been trained in a wide variety of languages and scripts, including English, Chinese, Arabic, and more.

Chapter 2 addresses agriculture, which is the most important part of everyone's life. Crops and plants play a huge role in the making of life and taking care of those crops and plants is both important and tough. Thus, to detect the disease in plants, this chapter demonstrates how to tell whether a plant is healthy or not. The dataset consists of chilly plant leaves collected from one of the fields located in Andhra Pradesh, India. Many image classification models, as well as transfer learning models, are applied. Deep learning models CNN and transfer learning models, like InceptionV3 and VGG16, are applied with and without data augmentation.

In Chapter 3, researchers have applied various deep learning and transfer learning methods to accurately predict the disease of a damaged plant, so that we can cure the plant in its initial stage. The models are trained on the image dataset containing various categories of plants like mango and pomegranate. The results state that ResNet outperformed Inception, VGG19, and CNN by giving an accuracy of 88% and 87.5% percent for pomegranate and mango respectively.

In Chapter 4, researchers have compared different deep learning-based classification techniques on a remote sensing image dataset. The dataset has been taken from the UC Merced Land Use Dataset, which contains a total of 21 classes, with every class consisting of 100 images of size 256 x 256. The models used in this study are VGG, ResNet, Inception, Dense Net, and Efficient Net, which are deep convolutional network architectures for image classification with different numbers of layers. To make meaningful comparisons, all models were extended by adding three layers at the end to improve their performance. The performance of the VGG19 model

was found to superior. This model was able to classify almost all images belonging to 21 classes with an accuracy of 100% in training and 95.07% in testing data, followed by VGG16 with 93% and ResNet with 91% accuracy in testing data.

Chapter 5 deals with sarcasm. Sarcasm is a sardonic or bitter remark, intended to express disrespect or ridicule. It is used in Hindi language, originating from many of Hindi idioms and proverbs, and often uses indirect sarcasm, as in saying, "अन्धों में काना राजा" with the meaning, "मूर्ख मण्डली में थोड़ा पढ़ा-लिखा भी विद्वान् और ज्ञानी माना जाता है." Sentiment categorization is easier in comparison to sarcasm detection, as we see in the above-written idiom, which contains a negative sentiment. The intention of the sentence is to call an uneducated person knowledgeable among a group of fools. In the present scenario, people on social media platforms like Twitter, Facebook, and WhatsApp succeed in recognizing sarcasm despite interacting with strangers across the world. Sarcasm detection is a challenging task in Natural Language Processing due to the richness of morphology. Detecting sarcasm in Hindi language tweets is a prime task for Natural Language processing to avoid misconstruing sarcastic statements as original literal statements.

Chapter 6 explains how the image is the main source for all image processing fields, like surveillance, detection, recognition, satellite, etc. Good visibility of images captured by sensors becomes crucial for all computer vision tasks. Sometimes the scene quality is degraded by bad weather conditions like haze, fog, or smoke, therefore making it difficult for the computer vision area to obtain actual information. Haze can be removed from a single-input scene by using dehazing methods. Synthetic haze can be created by a haze generator, and, currently, most image dehazing techniques are applied for synthetic haze. Various single-image dehazing techniques are being developed and tested on real-world scenes that are captured in hazy environments using cameras.

Chapter 7 demonstrates how the framework needs accurate and real-time performance to count how many people are present at a particular moment in a particular frame. So, our counting framework automatically detects each person's face and makes instantaneous decisions to count the number of persons in front of the camera or within a set of images. The work of individual counting can be done in two broad ways: the first is the detection of faces; the second is the counting approach used to track and count people within a frame.

Chapter 8 describes how CNN networks can show a resemblance to traditional steganalysis by using filters for feature extraction. Due to the use

of content-adaptive steganographic methods, the stego message is hidden more often in the complex areas of the image and thus cannot be detected with a simple statistical analysis of the image. The stego information in these steganographic methods affect the dependencies between the pixels introduced through various kinds of noise present in the image. Thus, the difference between the cover and stego image is identified through the noise part rather than the image content. Different researchers have used various preprocessing filters for calculating the noise residuals and passing them to the CNN network instead of images directly. This work employs a content adaptive steganography method, Highly Undetectable Steganography (HUGO), for creating stego images. Furthermore, this work provides a comparative analysis of one of the variants of CNN models specific for steganalysis and various pre-trained models of computer vision that apply to steganalysis.

Chapter 9 discusses how groundwater abstraction beyond the safe limit is causing a rapid groundwater table depletion at the rate of 1–2 m/year in many districts. Uncontained and unplanned usage may affect food production by 20 percent. Due to the significant impact of this imperceptible resource on various aspects of life, the economy, the environment, and society, there is a pressing need to enhance the scientific comprehension, estimation, and administration of groundwater management. A scientific framework for the demarcation of its potential storage and recharge zonal maps, i.e., GWPSZ and GWRZ, can be instrumental in this regard for urban and rural water committees to objectively manage the resources at the regional level.

Chapter 10 explains the process of using pre-trained models to classify different types of fruit leaves accurately. We also discuss the advantages of transfer learning for industrial applications, including improved accuracy, reduced training time, and better utilization of resources. We provide code examples and practical guidance for implementing transfer learning using popular deep learning frameworks like TensorFlow. By the end of this chapter, readers will have a good understanding of how to use transfer learning for fruit leaf classification and how it can be applied in industrial settings.

Chapter 11 reveals that the carbon footprint of these calculations proves to be rather high. Deep learning research can be challenging for academics, students, and researchers, especially those from emerging economies, due to the financial expense of calculations. In addition to accuracy and related metrics, productivity is included as an evaluation criterion in this chapter's proposed practical solution. Our objective is to create green AI

that significantly outperforms red AI in terms of receiver performance and to increase green AI by reducing its environmental impact.

Chapter 12 explains how networking services have modified the methods and scale of cyberspace communication. In the past decade, the social network has gained significant attention. The Internet and Web 2.0 applications are becoming more affordable for accessing social network sites like Twitter, Facebook, LinkedIn, and Google+. People are becoming more interested in information news and opinion on a wide range of issues and therefore more reliant on social networks.

Chapter 13 engages in an in-depth analysis of the associated ideas and challenges that arise in the process of shopping app interface design. It further demonstrates the functional simplicity, ease of use, and effectiveness that is centered on the experience of the user and is built on an interactive model via the use of classical instances and practical design projects. After that, we use these theories, together with associated ideas in design psychology, behavior, design aesthetics, and ergonomics, to thoroughly study mobile shopping app interaction design. It presents mobile shopping apps via the process of "global-local- global" for repeated personalized interactive design, and it gives useful recommendations and ideas for the construction of mobile and vertical shopping apps. Additionally, the chapter includes analysis and data collecting of practical instances.

Chapter 14 presents an extensive review of the application of machine learning and deep learning methods in detecting COVID-19. It emphasizes the significance of early and accurate diagnosis in effectively managing the disease during the COVID-19 pandemic. The chapter encompasses various aspects of machine learning and deep learning techniques, including supervised and unsupervised learning, convolutional neural networks, recurrent neural networks, reinforcement learning, and a comparative analysis of machine learning and deep learning methods for COVID-19 detection.

We are deeply grateful to everyone who helped with this book and greatly appreciate the dedicated support and valuable assistance rendered by Martin Scrivener and the Scrivener Publishing team during its publication.

Pramod Singh Rathore
Department of Computer and Communication Engineering
Manipal University Jaipur
Rajasthan, India
Dr. Sachin Ahuja
Department of Computer Science
Chandigarh University, India

Srinivasa Rao Burri
Senior Software Engineering Manager
Western Union
Denver, CO
Dr. Ajay Khunteta
Department of Computer Science and Engineering,
Poornima University, Jaipur, India
Dr. Anupam Baliyan
Dean of Academic Planning and Research
Galgotias University, India
Dr. Abhishek Kumar
Faculty of Engineering, Manipal University, Jaipur, India

1

Text Extraction from Images Using Tesseract

Santosh Kumar*, Nilesh Kumar Sharma, Mridul Sharma and Nikita Agrawal

Department of Computer Science, Global Institute of Technology, Jaipur, India

Abstract

Images play a crucial role in describing, representing, and conveying information, which are essential in many businesses and organizations. Text extraction from images is the method used to convert text to plain text. Text extraction refers to the systematic procedure used for converting textual material into a simplified plain text format. The task at hand presents a significant difficulty as a result of the many changes in variables such as the size, orientation, and alignment of the text. Furthermore, the inclusion of low-resolution, pixelated pictures, coupled with the existence of noisy backgrounds, exacerbates the complexity associated with the process of text extraction. Using the Tesseract OCR engine, we aim to reduce these issues in this project.

Tesseract is developed by Google and is an open-source optical character recognition (OCR) engine. OCR technology allows computers to recognize text in images, making it possible to convert images of text into machine-readable text. Tesseract has been trained in a wide variety of languages and scripts, including English, Chinese, and Arabic.

Tesseract can process images that are rotated, tilted, or skewed and can recognize text written in different scripts, such as English and Arabic. It uses machine learning algorithms to improve its recognition accuracy over time, making it suitable for use in a wide range of applications, including document scanning, archiving, and indexing.

***Keywords*:** Text extraction, LSTM, image text, OCR, Tesseract

1.1 Introduction

Text extraction, also known as optical character recognition (OCR), is the process of automatically extracting text from an image or scanned

**Corresponding author*: sonu225914@gmail.com

Pramod Singh Rathore, Sachin Ahuja, Srinivasa Rao Burri, Ajay Khunteta, Anupam Baliyan and Abhishek Kumar (eds.) Deep Learning Techniques for Automation and Industrial Applications, (1–18)

document and converting it into machine-readable text that can be used for indexing, searching, editing, or storing the document.

OCR software uses advanced algorithms and machine learning techniques to identify and recognize characters and words within the document. The process of text extraction typically involves the following steps:

(a) Pre-processing: In this step, the image or scanned document is prepared for OCR by optimizing its quality. Techniques such as noise reduction, image enhancement, and contrast adjustment are used to improve image readability and reduce errors during OCR.
(b) Layout analysis: The OCR software analyzes the layout of the document to identify the text areas, headings, and other features.
(c) Optical character recognition: In this step, the OCR software reads and recognizes the characters in the document and translates them into machine-readable text.
(d) Post-processing: In this step, the recognized text is refined to correct errors and improve its accuracy. Techniques such as spell-checking, grammar correction, and formatting are used to improve the quality of the output [1, 2].

Utilizing OCR software, such as Google's open-source OCR engine Tesseract, the aforementioned steps are possible. Tesseract uses advanced machine learning techniques to improve its recognition accuracy over time and can recognize text in multiple languages and scripts.

The following are the steps involved in text extraction using OCR software:

(a) Scanning or importing the image or document to be processed
(b) Preprocessing the image or document by adjusting its quality, orientation, and size
(c) Performing layout analysis to identify the text areas, headings, and other features(d) Running OCR on the document to extract the text
(e) Post-processing the recognized text to correct errors and improve its accuracy and formatting
(f) Saving the extracted text in a machine-readable format such as plain text or a searchable PDF.

Furthermore, by training the OCR engine on a specific dataset of languages, it is possible to improve the accuracy of OCR results [3, 4].

1.1.1 Areas

(a) Text extraction
(b) Python
(c) OCR
(d) React JS
(e) Flask

1.1.2 Why Text Extraction?

Text extraction, also known as optical character recognition (OCR), is a valuable technology because it enables the automated extraction of text from images and scanned documents. Here are some reasons why text extraction is important:

(a) Improved efficiency: Text extraction automates the process of converting image-based text into machine-readable text. This saves time and effort compared to manual data entry, especially when dealing with large volumes of data.
(b) Easy to search and index: Text extraction allows users to convert image-based text into searchable and indexable text. This is especially important when dealing with large collections of documents or when searching for specific information within a document.
(c) Improved accessibility: Text extraction makes it possible to convert image-based text into machine-readable text that assistive technologies like screen readers can read. This improves accessibility for people with visual impairments.
(d) Cost-effective: Text extraction can save organizations money by reducing manual data entry and improving the accuracy of document management processes (DMPs).
(e) Multilingual support: OCR technology has advanced significantly in recent years and now supports the recognition of text in multiple languages and scripts. This means that text extraction can be used in various multilingual applications, such as multilingual document processing [5] or translating [6].

1.1.3 Applications of OCR

(a) Digitization of Printed Documents: OCR can be used to scan and digitize printed documents such as books, magazines, and newspapers. The resulting digital text can then be stored, searched, and edited on a computer.
(b) Automatic data entry: OCR can be used to automatically extract data from forms, invoices, and receipts, saving time and reducing errors compared to manual data entry.
(c) Handwriting recognition: OCR can be used to recognize handwritten text, enabling applications such as digital note-taking or handwriting-based input for mobile devices.
(d) Text-to-speech: OCR can be used to convert printed text into speech, enabling visually impaired individuals to access written information.
(e) Language translation: OCR can be used to recognize text in one language and automatically translate it into another.
(f) Automatic license plate recognition: OCR can be used to recognize license plates on vehicles, enabling applications such as automatic toll collection or traffic monitoring.
(g) Document classification: OCR can be used to recognize the type of document being scanned, enabling automatic sorting and routing of documents.
(h) Image indexing: OCR can be used to index and search image collections, enabling users to search for images based on the text they contain.
(i) Historical document analysis: OCR can be used to digitize and analyze historical documents, enabling researchers to study the content and context of these documents in new ways.
(j) Passport and ID verification: OCR can be used to read and verify the text on passports and other identification documents, improving security and reducing the risk of fraud.

1.2 Literature Review

Text extraction using optical character recognition (OCR) has been a hot research topic for several years. The advancements in OCR technology have made it possible to extract text accurately and quickly from images, which has numerous applications in various industries. In this literature

review, we will explore the current state of research in text extraction using Tesseract OCR [7].

Tesseract OCR is an open-source OCR engine developed by Google that has gained popularity due to its accuracy and robustness. Several studies have been conducted to explore the effectiveness of Tesseract OCR in text extraction. One such study by Bhargava *et al.* (2020) compared the performance of Tesseract OCR with other popular OCR engines, such as Adobe Acrobat, ABBYY, FineReader, and OmniPage [8]. The study found that Tesseract OCR outperformed other OCR engines in terms of accuracy, speed, and ease of use.

In another study by Kumar *et al.* (2017), the authors proposed a novel method for text extraction using Tesseract OCR [9, 10]. The proposed method involved preprocessing the input image by applying various filters to enhance the quality of the image. The preprocessed image was then fed to the Tesseract OCR engine for text extraction. The results of the study showed that the proposed method improved the accuracy of text extraction significantly.

Moreover, several studies have focused on improving the accuracy of Tesseract OCR through machine learning techniques. For instance, in a study by M. Alzahrani *et al.* (2020), the authors proposed a machine learning-based approach to improve the accuracy of Tesseract OCR in recognizing handwritten text. The approach involved training a support vector machine (SVM) classifier using a dataset of handwritten characters. The trained SVM was then used to classify the characters extracted by the Tesseract OCR engine. The results showed that the proposed approach significantly improved the accuracy of Tesseract OCR in recognizing handwritten text.

Furthermore, a recent study by Bhargava *et al.* (2022) proposed a deep learning-based approach for text extraction using Tesseract OCR [11, 12]. The proposed approach involved training a convolutional neural network (CNN) using a dataset of images and corresponding ground truth text. The trained CNN was then used to preprocess the input image before feeding it to the Tesseract OCR engine for text extraction. The results of the study showed that the proposed approach outperformed traditional OCR engines in terms of accuracy and speed [13].

In addition to improving the accuracy of Tesseract OCR, several studies have explored its applications in various industries. For instance, in a study by S. J. Alzahrani *et al.* (2018), the authors proposed a method for extracting text from bank statements using Tesseract OCR. The proposed method involved preprocessing the input image by removing noise and enhancing the quality of the image. The preprocessed image was then fed

to the Tesseract OCR engine for text extraction. The results showed that the proposed method was effective in extracting text from bank statements, which can be used for financial analysis and fraud detection.

Similarly, in a study by R. K. Samal *et al.* (2017), the authors proposed a method for extracting text from medical images using Tesseract OCR. The proposed method involved preprocessing the input image by removing artifacts and enhancing the quality of the image. The preprocessed image was then fed to the Tesseract OCR engine for text extraction. The results showed that the proposed method was effective in extracting text from medical images, which can be used for medical diagnosis and research.

In conclusion, Tesseract OCR has proven to be an effective tool for text extraction from images. Several studies have explored its effectiveness and applications in various industries. The advancements in machine learning and deep learning techniques have further improved the accuracy of Tesseract OCR in recognizing text from images. The future scope of research in text extraction using Tesseract appears promising, with potential enhancements in areas like multi-language support, complex document formatting, and real-time processing [14–17].

1.3 Development Areas

1.3.1 React JavaScript (JS)

(a) React JS is a front-end JavaScript library used for building user interfaces.
(b) React JS uses a component-based architecture, which makes it easy to build reusable components for the UI.
(c) React JS can be used with other JavaScript libraries and frameworks to create powerful web applications.

1.3.2 Flask

(a) Flask is a Python web framework used for building web applications and APIs.
(b) It is a micro-framework, meaning that it provides the minimum set of tools needed to build a web application.
(c) Flask is flexible and allows developers to customize and configure it to meet their needs.
(d) It provides a built-in development server, which makes it easy to get started with building a web application.

(e) Flask is highly extensible and can be used with various Python libraries and tools to build powerful web applications.

Here are some other differences between React JS and Flask to keep in mind:

(a) React JS is a front-end library, while Flask is a back-end web framework.
(b) React JS is written in JavaScript, while Flask is written in Python.
(c) React JS is used for building the user interface, while Flask is used for handling server-side processing and database integration.
(d) React JS is used in combination with other front-end tools and libraries, while Flask is used with other Python tools and libraries.

In terms of using React JS and Flask together, we can build a web application where React JS handles the front-end user interface and Flask handles the back-end server-side processing. This can be done by creating a REST API in Flask and making API calls to it from React JS. Alternatively, we can serve Flask as a static file from the React JS application's public directory and make API calls to it from there.

1.4 Existing System

The automatic conversion of text in an image into letter codes that can be used in computer and text-processing applications is known as offline handwriting recognition. This process captures a static representation of handwriting. Due to variations in handwriting styles, offline handwriting recognition is a challenging task. Currently, OCR engines primarily focus on machine-printed text, and Intelligent Character Recognition (ICR) on hand-printed text (written in capital letters). Charles *et al*. have presented a comprehensive review of various techniques used for the design of OCR systems. The paper discusses slow techniques that provide accurate results and fast techniques that provide inefficient results. Ray Kurzweil invented the first OCR software in 1974, allowing recognition of any font. This software utilized an advanced matrix method (pattern matching) to compare bitmaps of the template character with the bitmaps of the read character and determine the closest match [18–20].

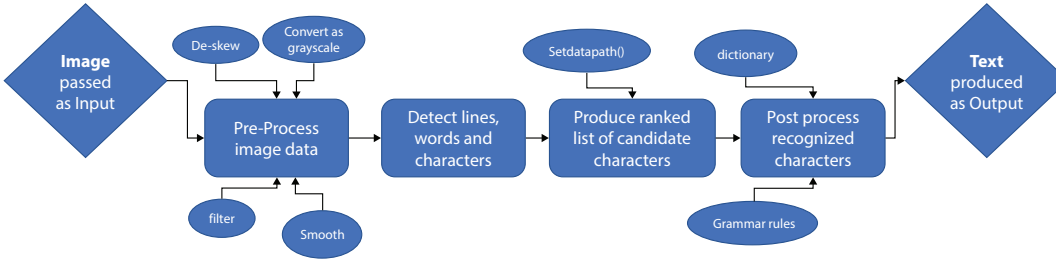

Figure 1.1 Methodology.