MINIPROJECT REPORT

ON

Marks2CSV

A simple solution to convert tabular mark fields to CSV file

Submitted by

Ajay T Shaju (SJC20AD004)

Emil Saj Abraham (SJC20AD028)

Justin Thomas Jo (SJC20AD046)

Vishnuprasad K G (SJC20AD063)

to

the APJ Abdul Kalam Technological University

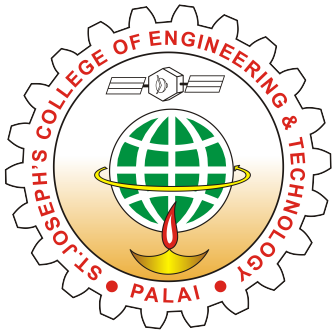
in partial fulfillment of the requirements for the award of the degree

of

Bachelor of Technology

in

Artificial Intelligence and Data Science



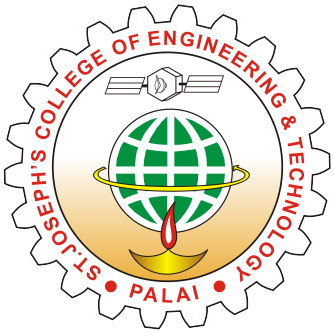
Department of Artificial Intelligence and Data Science

St. Joseph's College of Engineering and Technology, Palai

JUNE : 2023

ST. JOSEPH'S COLLEGE OF ENGINEERING AND TECHNOLOGY, PALAI

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE



CERTIFICATE

This is to certify that the report entitled "**Marks2CSV**" submitted by **Ajay T Shaju (SJC20AD004), Emil Saj Abraham (SJC20AD028), Justin Thomas Jo (SJC20-AD046), and Vishnuprasad KG (SJC20AD063)** to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Artificial Intelligence and Data Science is a bonafide record of the miniproject carried out by them under my guidance and supervision.

Project Guide
Dr.Deepa V
Head of the Department
Department of AD

Project Coordinator
Mr.Jacob Thomas
Assistant Professor
Department of AD

.
.
Place : Choondacherry
Date : 03-08-2023

Head of the Department
Dr.Deepa V
Assistant Professor
Department of AD

Acknowledgement

We wish to record our indebtedness and thankfulness to all who helped us complete this Mini Project work titled Mark2CSV. We would like to convey our special gratitude to Dr. V.P. Devassia, Principal, SJCET, Palai, for the facilities. We express our sincere thankfulness to Dr. Deepa. V, Head of the department, Department of Artificial Intelligence & Data Science for her cooperation and valuable suggestions. Also, we express our sincere thanks to the Mini project co-ordinator Mr. Jacob Thomas for his helpful feedback and timely assistance. We are especially thankful to our guide, Dr. Deepa. V, Head of the department, Department of Artificial Intelligence & Data Science for giving us valuable suggestions and critical inputs through guidance and support. We also extend our thanks to Dr. Mervin Joe Thomas, Postdoctoral Research Fellow, Indian Institute of Technology Palakkad, and Mr. Mithun M Sanjeev, Senior Software Engineer, Bosch Bengaluru for their amazing thoughts and help, and we thank all our friends and others who directly or indirectly helped us during our mini-project work.

Ajay T Shaju

Emil Saj Abraham

Justin Thomas Jo

Vishnuprasad K G

Abstract

The project focuses mainly on handwritten number identification, which address one of the most daunting challenges faced by any educational institutions globally - recognition and classification of handwritten digits from answer scripts. The proposed system can implement the idea of handwritten digit recognition for the pressing problem faced by educators in educational institutions – the manual mark entry. The need for a transformative solution to streamline and enhance the efficiency of the mark entry process in colleges was recognized, through extensive research and analysis.

The idea of a user-friendly software tool that harnesses the power of OCR technology in conjunction with state-of-the-art AI to convert images of mark cells on answer scripts of the institution into a CSV file with minimal intervention of teachers, was conceived. The system aims to simplify the entire mark entry process by providing a user-friendly interface for teachers to capture images of the answer scripts using a camera that converts the obtained images of marks into data that will be stored in a CSV file. The resulting CSV file represents the original content of the answer scripts, enabling the teachers to effortlessly edit, analyze, and evaluate the mark data.

The approach taken involved implementing advanced OCR algorithms, fine-tuned to extract mark cells on answer scripts with utmost precision and reliability. Leveraging cutting-edge frameworks such as TensorFlow, Numpy, and Pandas, a seamless pipeline was engineered to efficiently organize and format the extracted data. The necessary features were only selected from the extracted data using a CNN model, which was built from scratch. The result was an output that depicted that the model performed better compared to the existing solutions available. It also exhibited remarkable flexibility, allowing for effortless customization to cater to the specific needs of users.

In summary, this project aims to reduce the time consumed for the mark entry process in educational institutions. While the focus initially was laid on transforming the mark data entry procedures within educational institutions, one can envision a future where the modular system finds applications in diverse domains, simplifying complex data handling tasks and alleviating manual labor on a grand scale.

Table of Contents

List of Abbreviations

AI Artificial Intelligence

CNN Convolutional Neural Network

CSV Comma Separated Values

LSTM Long Short-Term Memory

OCR Optical Character Recognition

VRNPR Vehicle Registration Number Plate Recognition

List of Figures

List of Tables

Chapter 1

Introduction

The rapid advancement of Artificial Intelligence (AI) has revolutionized various aspects of human workflow, particularly in the field of data processing. One prominent application is the automation of handwritten digit recognition, which holds immense potential for numerous purposes. While recognizing digits from a small set can be relatively straightforward, dealing with vast quantities of handwritten digits poses significant challenges, as it becomes a time-consuming task prone to errors.

To address this critical issue, the project aims to develop a robust and reliable system capable of instantaneously converting handwritten digits into structured Comma Separated Values(CSV) files. By doing so, it alleviates the substantial burden placed on educators during the manual data entry process. This innovative system will empower teachers and researchers to allocate their valuable time more efficiently towards other essential responsibilities, ultimately enhancing the efficiency and accuracy of digit data processing in various domains.

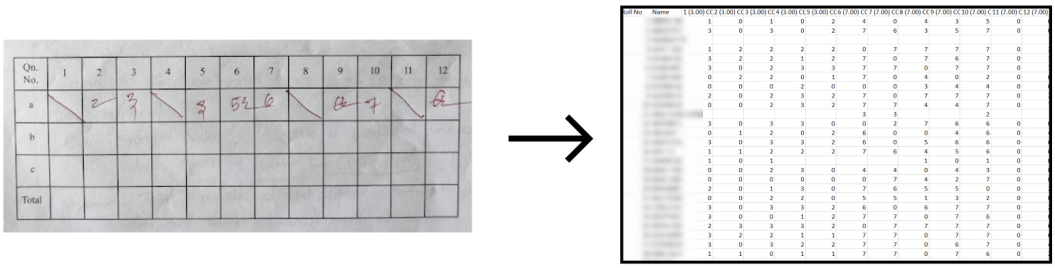


Figure 1.1: Handwritten Text to Digital Text

1.1 Background

In modern education, the digitalization of student exam scores and data is not only crucial for administrative and academic purposes but also a transformative necessity for educational institutions. As the volume of student data grows exponentially, manual data entry becomes increasingly cumbersome and error-prone, leading to inefficiencies and potential inaccuracies in record-keeping. This highlights the urgent need for automated solutions that can handle this task efficiently and accurately.

Computer software, especially powered by AI and machine learning, plays a pivotal role in addressing these challenges and revolutionizing the education sector by the extraction and conversion of handwritten exam scores into standardized formats like Comma Separated Values files. This automation not only saves valuable time and effort for educators but also drastically reduces the risk of transcription errors, ensuring the integrity of student records.

Furthermore, the benefits of digitalization extend far beyond administrative ease. With computer software handling data processing, teachers can dedicate more time and focus towards enriching the educational journey for their students. By gaining insights from the digitized data, educators can identify areas of improvement, tailor teaching approaches to meet individual student needs and implement targeted interventions to support struggling learners.

Standardization through CSV files empowers educational institutions with organized and easily accessible data, enabling sophisticated data management and analysis. These CSV files equip educators and administrators with actionable insights into student performance trends, curriculum effectiveness, and overall institutional performance. Armed with this knowledge, institutions can make data-driven decisions, optimize educational practices, and design evidence-based interventions to maximize student outcomes.

1.2 Motivation

The primary focus of this project is to overcome the existing challenge for the teachers to type-in all the marks scored by a student in an exam. The traditional method of manually entering data into spreadsheets or databases is a time-consuming and error-prone process. By utilizing the power of AI and machine learning, the project aims to create an automated system that can efficiently and accurately extract relevant data from answer scripts and convert it into standardized CSV files.

A key motivation for this project is to relieve educators from the burdening task of manual data entry. With more time teachers can redirect their efforts toward more critical aspects of their profession. For example, teachers can focus on creating effective lesson plans that matches the needs of their students, Teachers can offer personalized support to students, helping them do better in their studies. Moreover, the automation of data conversion empowers teachers to invest in their own professional development. They can use the saved time to attend workshops, engage in research, collaborate with colleagues, and explore innovative teaching methodologies.

Furthermore, the automated system brings about operational efficiency within educational institutions. By reducing the reliance on manual data entry, the risk of transcription errors is minimized, ensuring the accuracy and integrity of student records. The availability of reliable data enables educational institutions to make informed decisions, assess student performance, and identify areas of improvement in the curriculum.

1.3 Objective and Scope

1.3.1 Objective

The objective of this project is to develop a tool to assist teachers in the data entry process by implementing an efficient Optical Character Recognition (OCR) system using the Convolutional Neural Network (CNN) architecture. The tool aims to streamline the extraction of table marks from various images, enhancing accuracy and reducing manual effort. By utilizing the power of OCR technology, the tool will automate the recognition and extraction of digits from images, thereby saving the time of teachers and minimizing the errors. Furthermore, the customizable nature of the program will enable programmers to define their specific parameters and rules for extracting marks as per institutional requirements. Ultimately, this project aims to provide teachers with a reliable and efficient solution for data entry, optimizing their productivity and facilitating their administrative tasks.

1.3.2 Scope

In recent years, the rapid development of AI has witnessed remarkable advancements, particularly in the domain of computer vision and optical character recognition. These developments become a stepping stone for innovative applications in various sectors, including education. The scope of this project aligns perfectly with these recent AI developments, as it aims to leverage cutting-edge neural networks to automate the conversion of images of answer scripts into CSV format.

The utilization of neural networks for digit classification is a key highlight of this project. Neural networks have demonstrated very high accuracy in recognizing and classifying complex patterns, such as handwritten digits. By utilizing this technology, the automated system can achieve high precision in identifying and extracting numerical scores from answer scripts. Additionally, the system will be designed to detect decimal marks during the OCR processing stage.

As a result, the automated system becomes adaptable to a wide range of educational institutions and their specific scoring requirements, offering customers the flexibility of customization in mark detection. Furthermore, a compact and portable device can be created to encapsulate the tool, enhancing usability and convenience. Advancements in microprocessors and other techniques will help to design powerful AI-driven devices that are small, lightweight, and highly portable. This solution promises to facilitate an easy digitization of scores and streamlines the grading process, saving time and effort for educators while providing valuable insights into student performance.

1.4 Contributions

The primary contribution to this system involved extensive research and development efforts in creating a novel CNN model entirely from scratch. This undertaking required in-depth knowledge of deep learning architectures and image processing techniques.

The newly developed CNN model was carefully integrated into the proposed system, which was designed to address a specific problem or task. Once the model was fully implemented, a comprehensive evaluation process was conducted to assess its performance and capabilities.

To gauge the effectiveness of the proposed system, a comparison was made between the performance metrics of the newly developed model and those of other existing systems that were considered state-of-the-art in the field. This comparison was conducted using standardized evaluation metrics, ensuring a fair and objective assessment.

The results of this comparative analysis revealed a significant difference in efficiency, with the proposed system showcasing superior performance compared to the existing alternatives. The proposed CNN model demonstrated higher accuracy, faster processing speed and other relevant improvements on the specific task at hand. The observed superiority of the proposed system over other existing solutions is a clear testament to the effectiveness of the new CNN model and its successful integration into the overall system architecture.

Chapter 2

Literature Review

2.1 System Description

The system can be described based on the four phases:

1. Obtaining input from images
2. Recognizing table structure from the answer script
3. Applying OCR on the obtained images
4. Obtaining the required result as CSV files

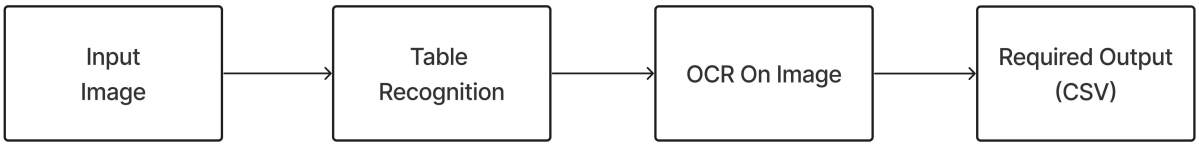


Figure 2.1: Initial concept of the system

The system uses a camera to accept images as input as per the requirements. This obtained input is then processed using the **img2table** library. Using this library, the table structure (that is, cell coordinates) is extracted and cropped to obtain the values inside the cell. These values are then given to the TensorFlow OCR model to accurately classify and predict the values obtained from the cell extraction step. The obtained result from this step is arranged specially in the format of CSV files.

2.2 Existing Solutions

The system description was based on the initial concept that was pitched before extensive research. The phases of this system concept may have existing solutions of various implications and importance which will be explored below.

Aaryan Raj et al. [1] proposed *Revolutionizing Data Entry: An In-Depth Study of Optical Character Recognition Technology and Its Future Potential* in February 2023. The paper provides a comprehensive analysis of the history, current state, and future of OCR technology. By the concluding statements of this research journal, it was deduced that OCR technology is applicable for this system.

Colin G. White-Dzuro et al. [7] proposed *Extracting Medical Information from Paper - COVID-19 Assessment Forms* in 2021. The paper examines the validity of optical mark recognition, a novel user interface, and crowd-sourced data validation to rapidly digitize and extract data from paper COVID-19 assessment forms at a large medical center. The paper also warns that their proposed model is prone to errors and only guarantees an accuracy of 70%.

Jiayi Yuan et al. [8] proposed *An OpenCV-based Framework for Table Information Extraction* in September 2020. They proposed a novel OpenCV-based framework to extract the metadata and specific values from PDF tables.

B.Gatos et al. [14] proposed *Automatic Table Detection in Document Images* in 2005. They proposed a workflow for table detection that comprises three distinct steps: (i) image pre-processing; (ii) horizontal and vertical line detection and (iii) table detection. The Probabilistic Hough Line Transform in the OpenCV package is discussed which can be used to create a bounding box around the desired cells that needed to be extracted.

The above discussed literature provides the system with a possible solution for table recognition and extraction.

Abhishek Das et al. [9] proposed *LSTM-based Odia Handwritten Numeral Recognition* in September 2020. The paper focuses on the loss in recognizing the digits and is evaluated using the categorical cross-entropy loss function. The main architecture discussed in this paper is Long Short-Term Memory (LSTM).

Raajkumar G. et al. [11] proposed *Optical Character Recognition using Deep Neural Network* in July 2020. It aims at analyzing various text images like blurred and tilted images and it identifies the text from these images using deep learning models. The main architecture discussed in this paper is CNN.

The above two references provided possible options on what approach could be followed for the process of handwritten digit recognition.

Chen ShanWei et al.[6] proposed *A CNN-based Handwritten Numeral Recognition Model for Four Arithmetic Operations* in 2021. It proposes an automatic checking system based on CNN for handwritten numbers. By the concluding statements of this paper, it was made clear that CNN was the most suitable choice for the proposed system. In future scope implementations, it is suggested to add layers of LSTM to compare the difference between the previous and updated versions of the model.

Ömer Aydin [4] proposed *Classification of Documents Extracted from Images with Optical Character Recognition Methods* in June 2021. It focuses on the fact that handwriting or printed documents have been digitalized by a scanner or digital camera and these documents have been processed with two different OCR operations.

Sachin Shrivastava et al. [5] proposed *CNN-based Automated Vehicle Registration Number Plate Recognition System* in March 2021. It focuses on the different methods of VRNPR and emerging technologies that are used to get accurate results.

These research papers helped gain more perspective and understanding of how CNN can be applied to real-life scenarios and situations of the most importance.

Bjorn Barz et al. [12] proposed *Deep Learning on Small Datasets without Pre-Training using Cosine Loss* in May 2020. It shows that the cosine loss function provides substantially better performance than cross-entropy on datasets with only a handful of samples per class.

According to the TensorFlow documentation, it is possible to apply the sparse categorical cross-entropy function. This is made possible because the labels for the number images stored in the folders have the same names as the folders themselves.

Akkem Yaganteeswarudu [10] proposed *Multi Disease Prediction Model by using Machine Learning and Flask API* in July 2020. It focuses on Python pickling, which is used to save the model behavior, and Python unpickling, which is used to load the pickle file whenever required. Using the conclusion from this paper, Flask API stands out to be the better tool that can be used to create a simple yet appealing application interface.

2.3 Summary

The literature review presented a thorough examination of different research studies and works relevant to the proposed system. It offered valuable insights and multiple potential solutions for addressing each phase of the development of the system.

Firstly, the comprehensive analysis of OCR technology [1] provided a very good understanding of the history, current state, and potential future advancements in this field. This analysis proved to be crucial in determining the most appropriate OCR technology to be integrated into the proposed system. By identifying the strengths and limitations of various OCR techniques, the study guided the selection of the most suitable technology to ensure accurate and efficient digit recognition from the answer script images.

Moreover, the research overviews on table extraction [8] and table detection [14] methods contributed significantly to the development of the capability of the system to detect and extract table structures from the scanned answer scripts.

By exploring different approaches and algorithms, these studies offered valuable insights into the most effective methods for identifying and capturing tabular data. Consequently, this enabled the system to accurately process and interpret table information, which is often an essential part of academic and assessment-related documents.

Furthermore, the comparison study conducted in [9] and [11] was instrumental in determining the optimal approach for the implementation of the system. By evaluating LSTM and CNN models, the research highlighted the unique strengths of each method and their suitability for different aspects of the system. As a result, it provided the necessary knowledge for making informed decisions on how to leverage both LSTM and CNN effectively within the proposed system, ensuring enhanced performance and accuracy in various processing tasks.

The discussions on the applications of CNN and LSTM in different contexts [4], [5], and [6] further corroborated the suitability of the chosen approaches for the proposed system. By analyzing how these models have been successfully deployed in diverse scenarios, the literature showcased the versatility and efficacy of CNN in handling image-related tasks and reinforced the decision to integrate it into the system.

Lastly, the focus on Python pickling and unpickling methods [10] offered insights into an efficient means of serializing and deserializing Python objects. Drawing from this study, the suggestion to utilize Flask API for building the user interface of the proposed system emerged as a pragmatic approach. The lightweight and versatile nature of Flask API, along with the compatibility with the pickling methods in Python, made it an ideal candidate for providing a user-friendly interface to interact with the system.

According to this, each phase of the proposed system is planned, which will be discussed in the subsequent chapter.

Chapter 3

Proposed Methodology

The proposed system is part of the ongoing work on recognizing handwritten numbers on documents and their classification. All the teachers face the time-consuming task of manually entering the marks scored by students for each question in an exam. This process can take a minimum of 4 to 6 hours, but it can be significantly reduced to 30 minutes or less by implementing efficient technologies built on top of robust AI algorithms. The system aims to comprehend the handwritten digits provided by humans and convert them into the required format, here CSV.

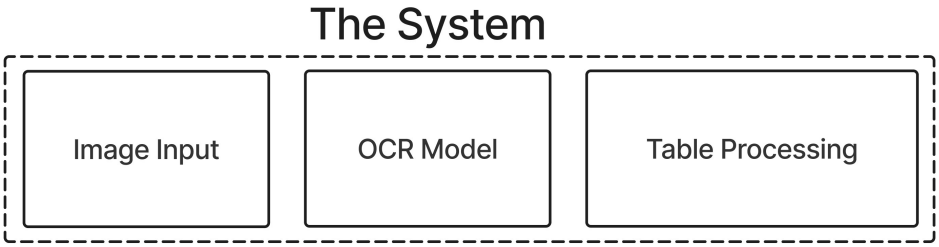


Figure 3.1: Main components of the system

As per the overall literature survey and research, it is evident that the modular approach for system building is the best. A modular design for the system is illustrated using Figure 3.1, where the idea of modules would be the building blocks for the system and they serve the purpose of ease of development and modification. The explanation for each block is as follows:

1. Image Input: Users can edit this module to make the system work with any input type like PDF, at the end the system should work on a single image at a time.

2. OCR Model: To make the system more robust to different handwritings this module can be edited or replaced with a better model weights file.

3. Table Processing: Users can edit this module to process tables of different formats (rows and columns).

.

3.1 Overview of the Proposed System

The core objective of this project is to create a highly intuitive and user-friendly software tool that facilitates the recognition of handwritten numbers and seamlessly converts them into structured CSV files. By employing Optical Character Recognition as its primary technology, the software allows users to capture a picture of the front page of an answer script using a simple camera as input. The captured image cells are then skillfully processed through a specialized Convolutional Neural Network Model, skillfully built using TensorFlow, to ensure precise and reliable conversion of handwritten marks into digital text.

This novel approach serves as a powerful solution to the challenges faced during manual data entry, offering educators and researchers an efficient means to transform vast quantities of handwritten data into easily manageable and structured CSV files. With the integration of OCR and CNN technologies, this software tool not only enhances the speed and accuracy of digit data processing but also paves the way for informed decision-making, data analysis, and optimal results across diverse domains.

Upon extracting the marks and converting the handwritten data into digital text, the software proceeds to process the information, generating a comprehensive CSV sheet that accurately reflects the content of the original answer scripts. This automated approach saves time and minimizes errors typically associated with manual data entry. The resulting CSV sheet is structured, organized, and ready for data analysis, providing the teachers with actionable insights to optimize their teaching approaches and interventions.

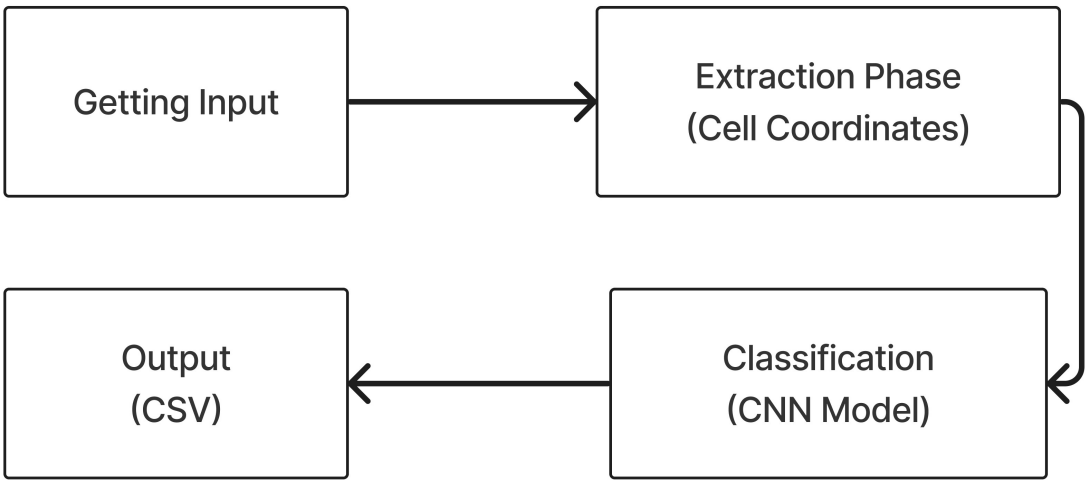


Figure 3.2: Overview of proposed system

3.2 Detailed Description Of The System

The application features a professional and efficient user interface, developed using HTML and Flask. Designed to enhance interactivity and performance, this interface serves as the entry point for teachers to interact seamlessly. A camera icon in the center enables users to quickly open the camera and capture images of answer scripts, streamlining the process. This intuitive design fosters a professional environment, empowering teachers with a streamlined approach to their tasks.

The input images are processed by the img2table library for table recognition. Figure 3.3 shows the recognized table structure.

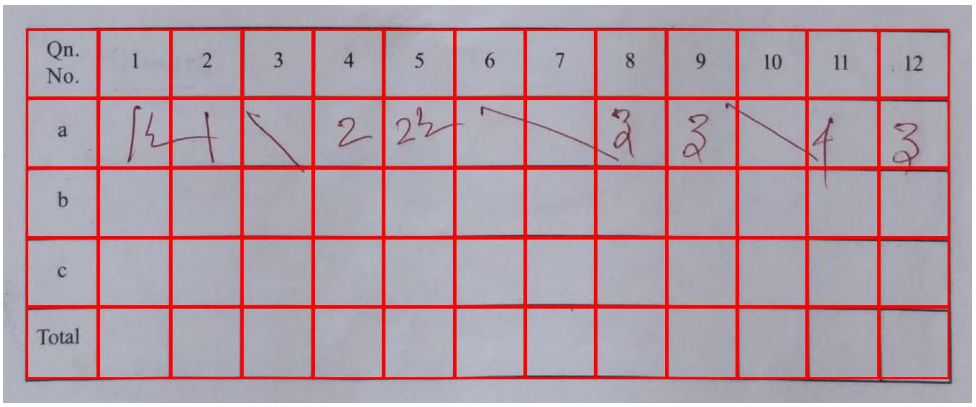


Figure 3.3: Sample Output Of Table Detection Process

The table processing phase of the system is designed by taking a sample of the answer scripts of SJCET Palai. SJCET college answer script is in the format of a horizontal table that consists of question number written on the first row (13 cells), the first column is a place-holder for headings (Qn. No. and Total), and possible divisions of questions in each question number (A, B, C). In overall, the table has the total number of cells and the total number of mark cells as 65 and 36 respectively.

The data preprocessing phase for the **mark table of SJCET Palai** answer script isolates the 36 mark cells, which may or may not contain the marks written on them. For this, the first and the last columns, as well as the first row are removed.
For extracting cells from the image and converting the extracted cells to its digital text, the model makes use of the img2table library and an OCR engine respectively, and for executing the whole process, there are two methods: one involves the use of **img2table library**, and the other involves **extracting cell coordinates**. A detailed explanation and side-by-side comparison are written in paragraphs and with a comparison Table 3.1 respectively.

Method 1: **img2table library**

In this method, the system uses the img2table library in combination with PaddleOCR (built-in to the same library) to detect the table, extract coordinates of table cells, and perform OCR on each extracted cell using PaddleOCR. These three processes are executed using an attribute called on the input image, which is converted to the proprietary document type of the library.

Qn. No.	1	2	3	4	5	6	7	8	9	10	11	12
a	2	12	None	3	None	None	None	None	6	6	None	6
b	None	None	None	None	None	None	None	None	None	None	None	None
c	None	None	None	None	None	None	None	None	None	None	None	None
Total	None	None	None	None	None	None	None	None	None	None	None	None

Table 3.1: DataFrame output of img2table library method using PaddleOCR

Consequently, the programmer does not have explicit control over the individual stages to meet specific requirements. The resulting DataFrame (shown in Table 3.1) is then post-processed to remove the first column and the first and last rows.

Method 2: **Cell Coordinates Extraction**

In this method, the system extracts only the cell coordinates using the img2table library. The extracted table cells are stored in an ordered dictionary.

OrderedDict is a dictionary subclass in Python that maintains the order of key-value pairs. Even if the value of a key is modified, the order of the keys remains unchanged. In contrast, a regular dictionary does not guarantee a specific order and may reorder the keys when their values are modified.

In the ordered dictionary data structure, each key represents a single row of the mark table, so deleting a key is equivalent to removing a row from the table, thus deleting the first and last keys of the ordered dictionary. There is a need to remove the first column (column with A, B, and C written) also, but it could be easily done after converting the presently existing cells to their DataFrame format.

From the two methods above, the second method was chosen as it has the following benefits (shown in Table 3.2).

	Cell Coordinates Extraction Method	**PaddleOCR Method**
Speed	Very fast as it works with built-in data structures.	Slow, as it runs the big PaddleOCR engine.
Programmer's Control	Highly controllable	No control
Ease of programming	Difficult	Easy
Ease of understanding	Medium	Easy Outwards, Internal working codes are complex.
Time taken	13 seconds for 5 papers	44 seconds for 5 papers

Table 3.2: Comparison of Table Processing Methods

Once the ordered dictionary is processed, the remaining table cells are cropped based on these coordinates and forwarded to the CNN model. Table 3.3 depicts the classification time taken by the CNN model for each individual image, showcasing the exceptional speed and efficiency of the model. The CNN model is specifically designed with a minimum number of layers to make it perform efficiently on smaller images. This way, it ensures that the proposed system prioritizes speed without compromising accuracy. It boasts an impressive capability to process five images in a mere 13 seconds, showcasing its remarkable performance.

Image Count	Time per step (ms)
1	23
2	18
3	17
4	17
5	16
6	16
7	16
8	23
9	20
10	19
Average time:	18.5

Table 3.3: Classification time taken for CNN OCR Model Version 1

The output after the classification and some processing is a DataFrame. This DataFrame is post-processed to remove the first column using a *Pandas* function.

The final dataframe will be converted to a NumPy array for flattening the DataFrame; And the two-dimensional array is flattened column-wise to get the marks corresponding to each sub-division (1A, 1B, ..., 12B, 12C). The output is a one-dimensional array that represents the marks scored by a student in individual questions.

The flattened array is incorporated into a dictionary to store the marks of individual students. Following this step, additional coding is applied for post-processing, which includes the removal of columns with identical entries and adding the columns *Roll No.* and *Name* to the left side of the DataFrame. This DataFrame is then converted to CSV format without the index values.

Roll No	Name	1a	2a	3a	4a	5a	6a	7a	8a	9a	10a	11a	12a
		0	0	3	0	0	4	0	0	0	0	0	0
		3	0	3	0	0	4	4	0	0	0	0	0
		3	3	2	0	0	5	0	0	3	0	3	2
		3	3	2	3	3	0	5	5	3	0	0	7
		3	3	3	0	0	0	0	0	5	5	0	0

Table 3.4: Output CSV File

The result is a refined CSV file that has the marks of all students (see a sample of output CSV in Table 3.4). The Roll Numbers and Names can be filled in by the faculty as roll numbers and names were not included in the work. Notably, this CSV file is automatically downloaded through the interface of the application, enabling seamless access to the finalized output on the local system.

3.3 Block Diagram

3.3.1 Overall working of the system

A camera is used to acquire the images and is stored in a list data structure. From each of these images, the table structure of the mark table is detected. This is done by a table detection algorithm, where the image is first cropped to half, and the table in the lower part is recognized by an optimized OpenCV algorithm (houghlinesP) then each cell coordinates are taken from the recognized table and is returned in an ordered dictionary.

From this ordered dictionary, the first and last rows are removed by deleting the first and last keys of the ordered dictionary and transferring the cropped image using cell coordinates to the TensorFlow OCR model for classification. After the classification result is returned as a dataframe, from which the first column is removed. Now the dataframe is flattened to add the classified values to the marks dictionary. After this process runs on every image, this marks dictionary is transformed into a CSV file to obtain the final output.

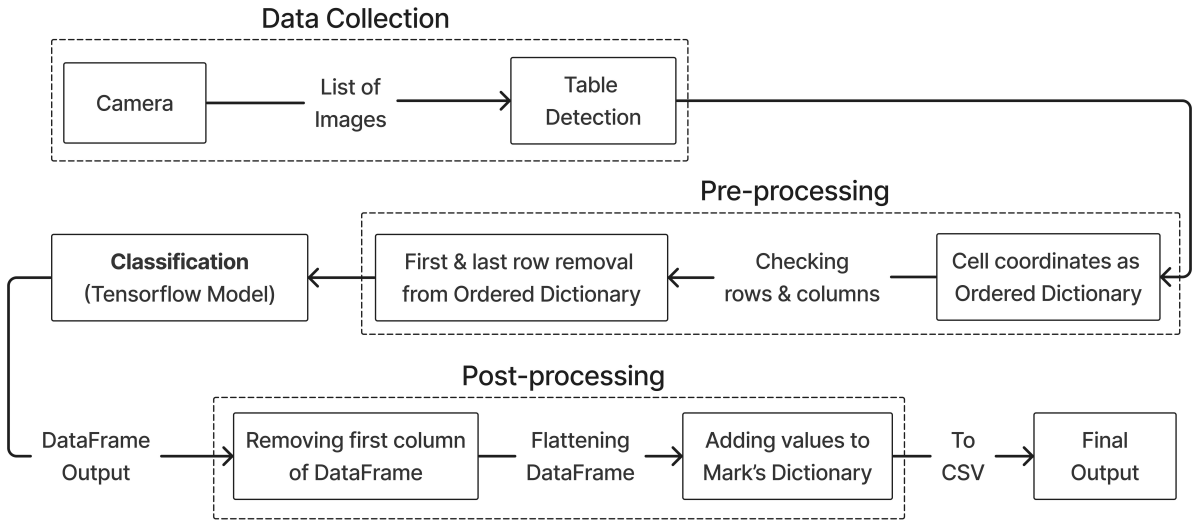


Figure 3.4: Flow diagram of the proposed system

3.3.2 Data Collection

Initially, 614 images of data were collected privately. Understanding the fact that this data was insufficient to train the model properly, a collection of an additional 21,600 images from a public Kaggle dataset was made. As per the project requirements, the removal of image classes - numbers 8 and 9 from the public dataset reduced the public dataset image count from 21,600 images to 17,454 images. So the final dataset has a sum of 18,068 images.

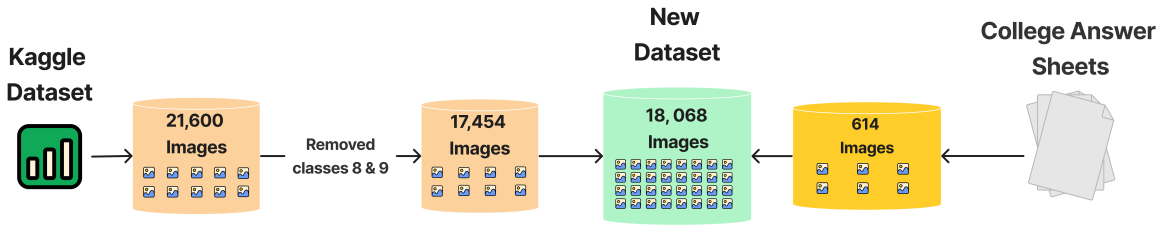


Figure 3.5: Dataset Collection

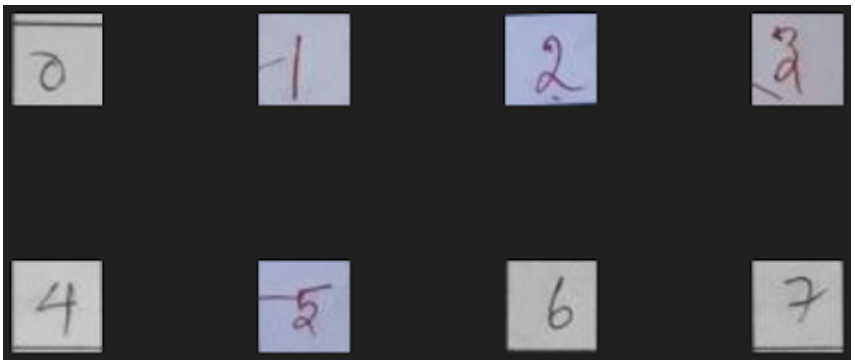


Figure 3.6: Mark cells of private dataset

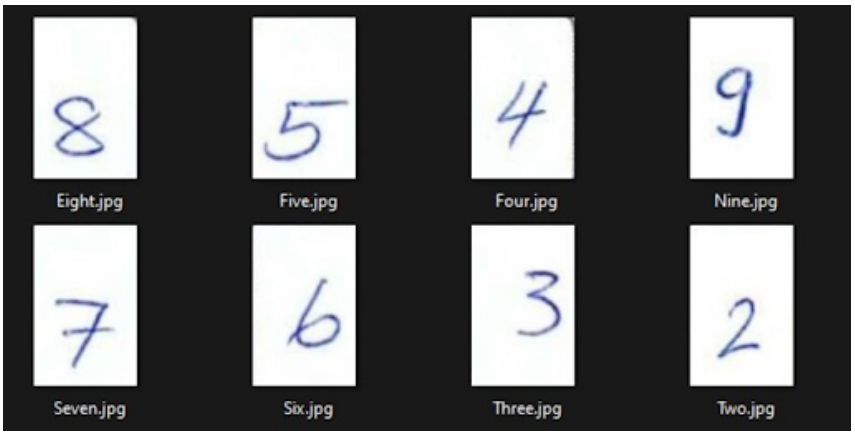


Figure 3.7: Mark cells of the public dataset from Kaggle

3.3.3 Data Pre-processing

In the process of automated marks extraction, a crucial feature to be extracted is the mark associated with each cell of the table. A table extraction algorithm is employed to achieve this, which facilitates the extraction of marks from each cell when the entire table is detected. The algorithm operates by identifying the boundaries of the table and dividing it into square dimensions, representing individual cells.

Once the table is successfully detected and divided into cells, the marks extraction process begins. Each cell is isolated, and the algorithm focuses on extracting the mark contained within it. This is where the CNN model comes into play. The extracted cell is passed through the CNN model, which has been trained to classify and interpret the marks present within the cells accurately.

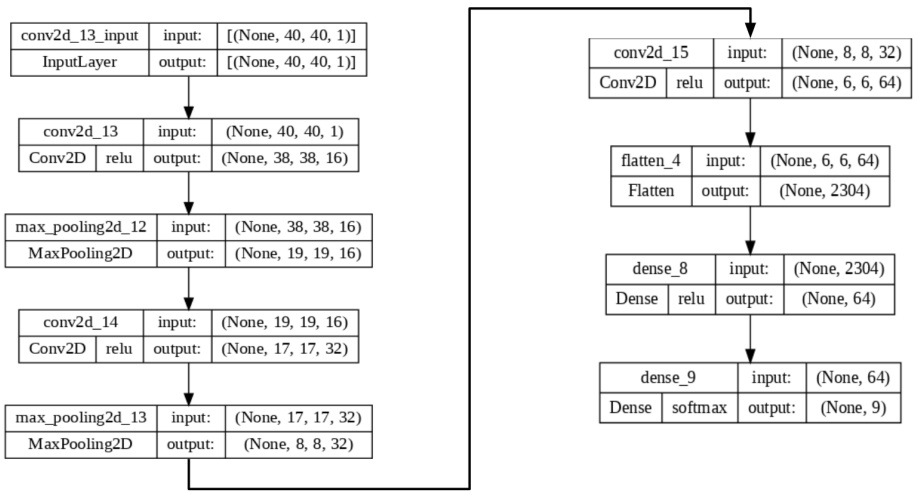


Figure 3.8: CNN_Model_1 Network Architecture

The model architecture (refer to Figure 3.8) consists of a sequential arrangement of layers. It begins with a convolutional layer, utilizing 16 filters of size 3x3 and employing the ReLU activation function. This layer processes input images with dimensions of 40x40 pixels and a single color channel. Subsequently, a max-pooling layer is applied to downsample the feature maps.

The next convolutional layer incorporates 32 filters of size 4x4 and also uses the ReLU activation function. However, there is no max-pooling layer following this convolutional layer, which helps preserve spatial information in the feature maps. The flattening layer is then employed to transform the multidimensional feature maps into a flat representation.

Two dense layers are added next. The first dense layer consists of 64 units and uses the ReLU activation function. The final dense layer has a number of units equal to the number of output classes and employs the softmax activation function, providing probability distribution predictions.

Moreover, the model is compiled using the Adam optimizer. The loss function used is sparse categorical cross-entropy, suitable for multi-class classification tasks. The metric used for evaluation is accuracy, measuring the performance of the model during training and testing. This architecture, with the absence of a pooling layer after the last convolutional layer, is designed to effectively process and classify images with high accuracy.

In conclusion, the process of marks extraction from table cells involves the utilization of a table extraction algorithm and a CNN model. The algorithm enables the identification and isolation of individual cells within the table, allowing for the extraction of marks from each cell. The CNN model plays a vital role in accurately classifying and interpreting the marks within the cells. The inclusion of exception checking within the algorithm ensures error handling and robustness in cases where the expected number of cells is not detected. By incorporating these components and mechanisms, the automated marks extraction system achieves reliable and accurate results, facilitating streamlined data processing and analysis in educational assessment processes.

3.3.4 Classification

The marks extraction process in the system employs a Convolutional Neural Network model for classification. The CNN is trained to categorize input images into nine classes, representing numerical values from 0 to 7, and an additional class for empty cells. It classifies the numerical equivalent of image cells or identifies them as null if empty.

During the prediction phase, the CNN model takes an input image and processes it through its layers. By analyzing the image features and extracting meaningful representations, CNN makes accurate predictions and classifications.

To ensure the correct sequence of marks, the extracted predictions are stored in a dictionary structure. Each dictionary key corresponds to a table-detected cell, ensuring that the predicted cell values are associated with their respective positions within the table. This maintains the integrity and accuracy of the marks extraction process.

By predicting all cell values before receiving the next input, the CNN model ensures efficient and consistent processing. This approach swiftly extracts and stores the predicted marks in their corresponding positions, facilitating streamlined data management and subsequent analysis.

Thus the classification of the system is achieved through a CNN model, effectively categorizing image cells into numerical classes or identifying them as null. The trained representations of the model enable accurate classification, and the extracted marks are efficiently stored in a dictionary, preserving their positions within the table.

This methodology ensures the efficiency and accuracy of the system in the extraction of marks, making it a reliable and valuable tool for educational institutions.

3.3.5 Data Post-processing

After obtaining the classification result in the form of a dataframe, the first crucial step in the data post-processing phase involves removing the first column of the dataframe, this is done to remove the column that contains the possible sub-sections(A, B, C) corresponding to each question numbers(1A, 1B, 1C, ..., 12B, 12C). This column removal will give the marks which is the essential thing need from the table and it simplifies the data for further processing.

Once the dataframe is appropriately cleaned, the next step is to flatten the resultant dataframe column-wise. This process allows to collect the marks corresponding to each sub-section of a question (if they exist). The final one-dimensional array is integrated into the marks dictionary, with respect to the dictionary keys(where keys represent the question number with sub-section alphabet).

This ensures that the extracted marks are well-organized within the dictionary. Each image inputted into the system will go through this process, consolidating the extracted marks from multiple answer scripts into the marks dictionary.

After processing all the images, the marks dictionary will contain the scores obtained by students in each question. However, it may also include columns with no marks (for questions without sub-sections). To remove these empty columns, the dictionary is converted to a dataframe for easy processing then the empty columns are dropped using appropriate code to make the output look similar to what is manually created by humans. For convenience in editing, two empty columns named 'Name' and 'Roll Number' will be added to the right side of the dataframe, so that the teachers can add the name and roll number data seamlessly.

The final dataframe is converted to CSV format and this file will have marks that are scored by the students in each question, or in other words this output is what the teachers create in 4-6 hours using their valuable time. The CSV format provides an easy-to-read way of storing the extracted marks, making it simple to use with other applications and allowing for further analysis and edits if needed. Successfully completing the data post-processing phase signifies achieving the objectives of the project- developing an efficient and effective system that automates the mark extraction process from answer scripts, significantly reducing workload of teachers.

3.4 Summary

The proposed system which is discussed in detail performs very efficiently in comparison with existing systems. The computation time of the proposed system varies from computer to computer as the specifications of the computer in use determine the difference in computation time when compared to other computers. Based on calculations, the proposed system is capable of achieving an average time of 18.5 milliseconds for the classification of images. Also, the proposed system also exhibits the ability to process 5 samples of answer scripts in 13 seconds, which pushes the advantage of using the system even more.

Chapter 4

Results and Discussions

The aim of the system is to provide users with a reliable and efficient solution for data entry in mark entry systems that in turn, optimizes their productivity and provides more opportunities for their administrative tasks.

The main platform used for the development of the system is Python. TensorFlow, a framework in Python, is used to create the OCR model using the implementation of a CNN network model. All the significant sections of code, excluding the front-end interface was purely implemented in Python.

CNN_Model_0 was the initially pitched model plan for the proposed system. It consisted of 7 layers, each layer possessing a small filter size just as in the case of a classical CNN model. CNN_Model_1 is an improvement over CNN_Model_0, with the addition of a convolutional layer possessing a comparatively bigger filter size. This pushes the advantage of CNN_Model_1 over CNN_Model_0 by a high margin.

Table 4.1: Image Size and Channels

Image Size	Number of Channels	Channel Name
40x40	1	Grayscale

4.1 Performance Evaluation

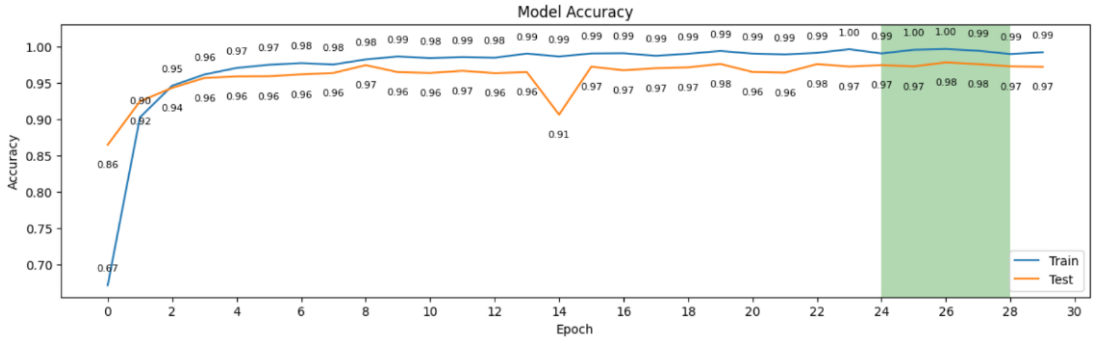


Figure 4.1: Training accuracy per epoch of CNN Model

CNN_Model_0 is improved upon by the addition of a new convolutional layer, which leads to the creation of CNN_Model_1. The smallest of changes has an impact on various performance measures related to the CNN model. Though CNN_Model_0 outperforms CNN_Model_1 in training accuracies, CNN_Model_1 gets the upper hand when it comes to validation and testing accuracies. Upon analyzing Figure 4.1, CNN_Model_1 has a dip in accuracy during the 14th epoch of training and it can be attributed to a phenomenon called "overfitting".

Overfitting occurs when the model becomes too specialized in learning the training data, losing its generalization ability to new, unseen data.

At the 14th epoch, the model might have started to memorize specific patterns in the training set, leading to a decrease in accuracy on the validation data. This memorization can cause the model to perform poorly on examples it has not encountered before, resulting in a temporary drop in accuracy.

As a remedy to this issue, several approaches such as early stopping and regularization can be implemented. But the overall graph shows only a negligible sign of overfitting, indicating effective training and resource utilization. The green shade, representing the best epoch-accuracy range, is prominently located towards the end. This signifies optimal performance without wasting resources.

As per the performance, the system is capable of recognizing most of the number written on a paper correctly (Figure 4.2), but still, there are limitations for the model, like the decimal marks (Figure 4.3) or unrecognizable writing styles (Figure 4.4) or due to cut and corrections in the image (Figure 4.5).

Figure 4.2: Cells with mark written correctly

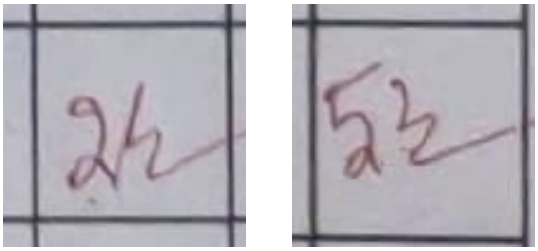


Figure 4.3: Cells with half marks (unable to detect)

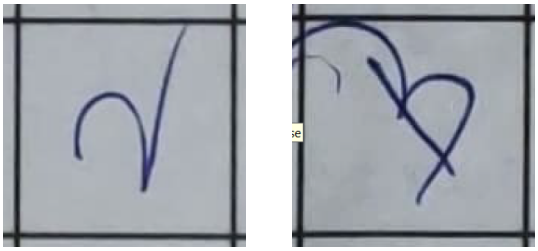


Figure 4.4: Cells with hard-to-recognize marks (may give false result)

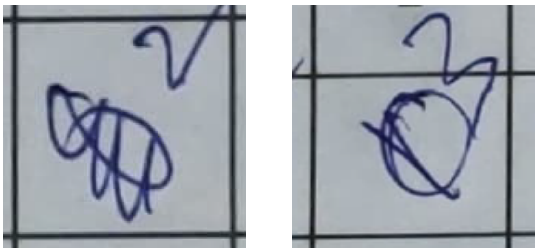


Figure 4.5: Cells with cuts and corrections (unable to detect)

4.2 Comparison with Model Versions

There have been two versions of the model named CNN_Model_0 and CNN_Model_1. The two models differ in their purpose and structure.

4.2.1 Phases of Model Development

Phase-1: CNN_Model_0

The CNN_model_0 is the 1st model developed that showed par results with the required accuracy, precision, and recall. This model contains 2 convolutional layers, a flattening layer, 3 max-pooling layers, and a dense layer. This model tends to unevenly train all the classes resulting in the high performance of highly trained classes and the low performance of lowly trained classes. Imbalanced learning of classes often leads to a decrease in the classification ability of the model. Operations like image recognition require its effectiveness and are crucial for its practical utility in real-world applications. This problem of uneven learning paved the way for the development of a better model in the next phase.

Phase-2: CNN_Model_1

This is the fully developed and optimized model that was made by fine-tuning the CNN_Model_1 that is used in Marks2csv. It was developed by adding another convolution layer and removing a max-pooling layer. This model thus consists of three Convolutional layers with 16, 32, and 64 filters, each followed by a MaxPooling layer. The output is then flattened and passed through a Dense layer with 64 neurons and ReLU activation. Finally, the output layer has the number of classes with softmax activation for classification. This solved the problem by making all the classes evenly learn to its training data.

Model Name	Epochs	Learning Rate	Layers Count	Features of Layers
CNN_Model_0	30	0.001	7	Smaller Filter Size.
CNN_Model_1	30	0.001	8	New Conv2D Layer And Bigger Filter Sizes.

Table 4.2: Comparison Of Two CNN OCR Models

4.2.2 Comparision of CNN_Model_0 and CNN_Model_1

The models have achieved a testing accuracy of 99% and 99.2% for CNN_Model_0 and CNN_Model_1 respectively. For an in-depth comparison, Table 4.3 is useful as it can be seen that class 3 of CNN_Model_0 has a little dip while in the figure of CNN_Model_1, the model has learned all classes equally.

Comparing the confusion matrices (given in Table 4.3), CNN_Model_0 showed a slight dip in accuracy for classes 3 and 5, indicating room for improvement. In contrast, CNN_Model_1 demonstrated balanced learning across all classes, indicating its ability to classify instances accurately. These insights gives guidance in refining the models for enhanced performance and accuracy.

The data, given in Table 4.5, compare the performance metrics for the two versions of CNN models. It can be seen that the precision values have not changed with the change in versions. But the recall values have decreased by 0.001 to reach 0.986 in CNN_ Model_1. Also, the F1-scores have decreased by 0.011 to reach 0.988 in CNN_ Model_1.

Table 4.3: Accuracy By Class Comparison CNN_Model_0 and CNN_Model_1

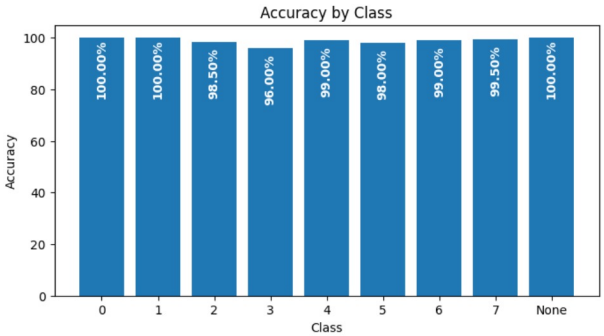
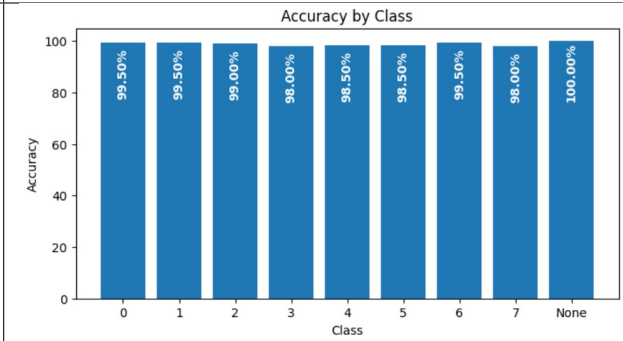
CNN_Model_0	CNN_Model_1
	

Table 4.4: Confusion Matrix Comparison CNN_Model_0 and CNN_Model_1

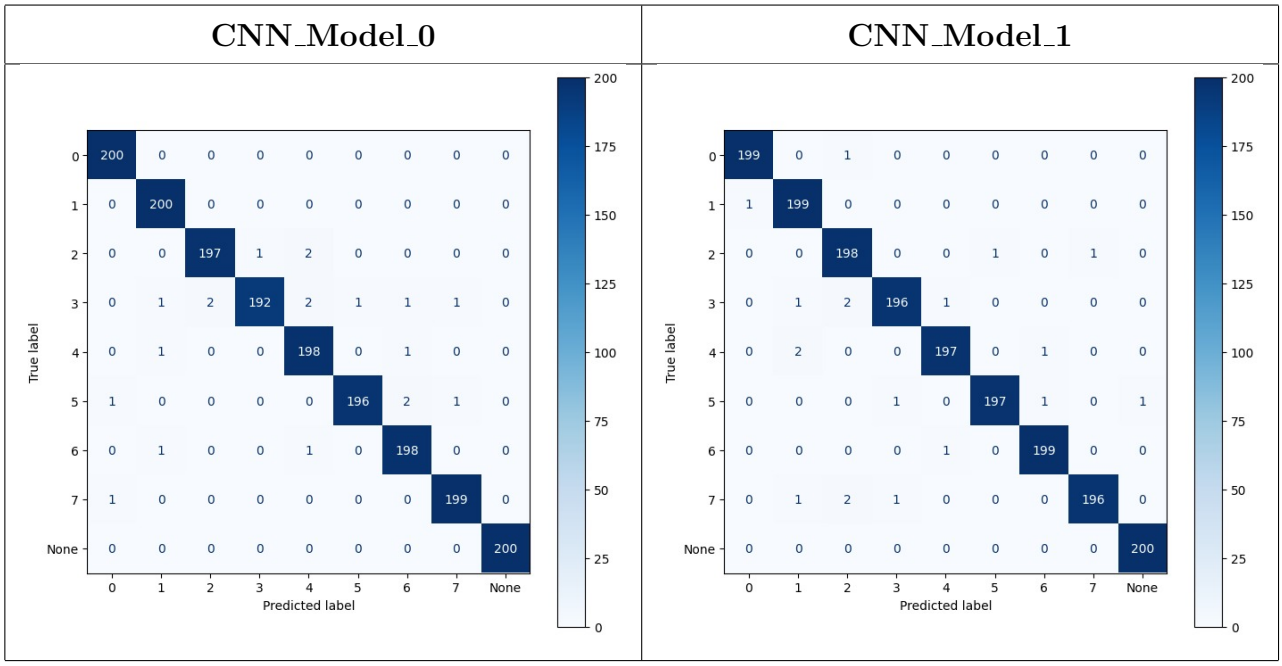


Table 4.5: Performance metrics comparison for two versions of CNN Models

	0	1	2	3	4	5	6	7	None	Overall
CNN_Model_0										
Precision	0.99	0.99	0.99	0.99	0.98	0.99	0.98	0.99	1	0.988
Recall	1	1	0.98	0.96	0.99	0.98	0.99	0.99	1	0.987
F1-Score	1	0.99	0.99	0.98	0.98	0.99	0.99	0.99	1	0.999
CNN_Model_1										
Precision	0.99	0.98	0.98	0.99	0.99	0.99	0.99	0.99	1	0.988
Recall	0.99	0.99	0.99	0.98	0.98	0.98	0.99	0.98	1	0.986
F1-Score	0.99	0.99	0.98	0.98	0.99	0.99	0.99	0.99	1	0.988

4.3 Comparison with State-of-the-Art Methods

The Marks2CSV application utilizes a cutting-edge CNN model developed from scratch, incorporating diverse libraries and modern techniques. This model represents a significant advancement in accuracy, precision, and recall (see Figure 4.7), and can even outperform previous state-of-the-art technologies.

4.3.1 Lenet5 vs CNN_Model_1

LeNet-5 is a classic convolutional neural network architecture designed by Yann LeCun et al. in 1998. LeNet-5 was a pioneering CNN architecture that demonstrated the potential of deep learning for image recognition tasks. It laid the foundation for the development of more advanced CNN architectures and significantly contributed to the success of deep learning in various computer vision applications.
One key advantage of the Marks2CSV application is its exceptional efficiency, providing outputs within seconds. The output is in CSV format, which is both machine-readable and writable, allowing seamless integration with existing data processing workflows. This time-saving capability and data manipulability distinguish Marks2CSV from other tools, making it invaluable for various applications.
The LeNet5 model, on the other hand, serves the specific purpose of detecting numerical and mathematical operations. While sharing a similar architecture with the CNN model, the divergent outcomes arise from their distinct task focus.
The Marks2CSV application utilizes the model CNN-Model-1 and is largely comparable to the popular LeNet5. Here is a detailed comparison between the two models:

1. **Architecture:**

   CNN_Model_1: It has a simpler architecture with fewer layers and neurons compared to LeNet-5.

   - uses 40x40 input images
   - Convolutional Layers: 3 layers (with 16, 32, and 64 filters)
   - Max-Pooling Layers: 2 layers

- Flatten layer: 1 layer
- Hidden Layers: 1 fully connected layer with 64 neurons
- Output Layer: 1 fully connected layer with num_classes neurons (using softmax activation)
- uses the ReLU activation function in the fully connected layers

LeNet5: The architecture consists of 7 layers, including 2 convolutional layers and 2 fully connected layers. The LeNet5 model shares a similar architecture with the CNN-Model-1, but it is specifically designed for detecting numerical and mathematical operations.

- Input Layer: Grayscale images with a shape of 32x32 pixels.
- Convolutional Layers: 2 layers (with 6 and 16 filters followed by Tanh activation).
- Average Pooling Layers: 2 layers of 2x2 pooling with a stride of 2.
- Fully Connected Layers: 2 layers (120 neurons and 84 neurons with Tanh activation).

2. **Performance:**

CNN_Model_1: The CNN-Model-1 demonstrates superior performance compared to previous state-of-the-art technologies. It achieves an accuracy of 0.99, precision of 0.99, and recall of 0.99, indicating its high accuracy in classifying marks from images. It was tested with both ADAM and Stochastic Gradient Descend Optimizers and results show that it is robust and maintained an accuracy of no significant dips in the value. (see Table 4.6)

LeNet5: The LeNet5 model achieves a lower accuracy of 0.87, precision of 0.88, and recall of 0.87. While it still performs reasonably well, it falls short compared to CNN-Model-1 in terms of accuracy, precision, and recall. Although it performed as well as CNN-Model-1 in terms of accuracy, Table 4.6 shows a significant dip in precision and recall when using SGD Optimizer.

3. **Task Focus:**

   CNN_Model_1: The CNN-Model-1 is designed to handle a broad range of tasks related to marks extraction and processing. It excels in accurately classifying images of whole numbers.

   LeNet5: The LeNet5 model is specifically tailored for detecting numerical and mathematical operations. It focuses on identifying and recognizing numerical characters, symbols, and mathematical expressions within the images.

4. **Efficiency:**

   CNN_Model_1: The Marks2CSV application utilizing CNN-Model-1 offers exceptional efficiency, providing outputs in seconds. This fast delivery of results enables efficient data processing and analysis, enhancing productivity.

   LeNet5: The efficiency of the LeNet5 model may vary depending on the complexity of the numerical and mathematical operations being detected. However, it is still an effective tool for identifying and extracting specific types of information within the images.

Table 4.6: Tabular comparison of performance metrics (CNN Model and LeNet5)

Optimizer	**ADAM Optimizer**			**SGD Optimizer**		
Model	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
CNN_Model_1	0.99	0.99	0.99	0.99	0.97	0.97
LeNET5	0.87	0.88	0.87	0.87	0.46	0.25

Given below is the graphical representation of Table 4.6

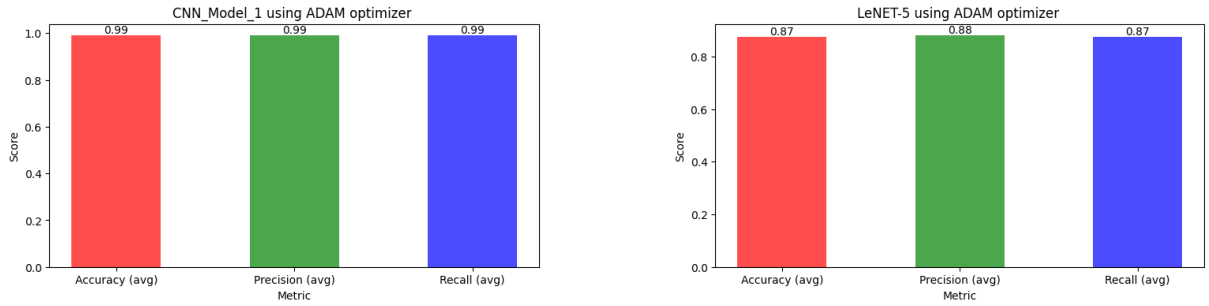


Figure 4.6: Model Comparison: CNN Model 1 vs LeNet5 with ADAM Optimizer

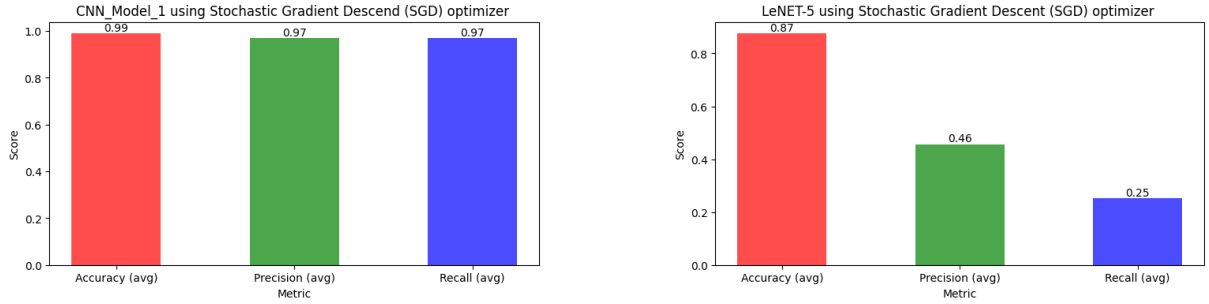


Figure 4.7: Model Comparison: CNN Model 1 vs LeNet5 with SGD Optimizer

4.4 Discussion

In the evaluation of two models, namely CNN-Model-1 and LeNet5, their performance was analyzed using two different optimizers: Adam and SGD. The objective was to assess the impact of optimizer choice on the accuracy of the models.

When testing CNN-Model-1 with both Adam and SGD optimizers, it was observed that there was no significant change in accuracy. Regardless of the optimizer used, CNN-Model-1 consistently performed well, indicating its robustness and stability. This suggests that the choice of optimizer had minimal influence on the overall accuracy of the model. Thus, CNN-Model-1 demonstrated its superiority by consistently maintaining a high level of accuracy in both optimizer scenarios.

On the other hand, LeNet5 exhibited a contrasting behavior when tested with Adam and SGD optimizers. While LeNet5 initially showed promising results with the Adam optimizer, a considerable performance drop was observed when switching to the SGD optimizer. This performance degradation indicates that the SGD optimizer was not suitable for the LeNet5 architecture, leading to a significant decrease in accuracy. This discrepancy highlights the sensitivity of LeNet5 to the choice of optimizer and emphasizes the importance of selecting an appropriate optimizer for optimal performance.

Based on these findings, it is evident that CNN-Model-1 outperformed LeNet5 in both optimizer scenarios. CNN-Model-1 consistently demonstrated higher accuracy and showcased its robustness by maintaining its superior performance regardless of the optimizer choice. These results emphasize the effectiveness and superiority of CNN-Model-1 over LeNet5, positioning it as a more reliable and accurate model for the given task.

It is worth noting that further analysis and experimentation may be required to determine the underlying factors contributing to the contrasting performances of the two models with different optimizers. Additional investigations into model architecture, dataset characteristics, and hyperparameter tuning could provide valuable insights into the observed performance differences.

Chapter 5

Conclusion

The Marks2CSV application has emerged as an invaluable solution for educational institutions in efficiently converting and processing the marks of students. With its utilization of a meticulously developed Convolutional Neural Network Optical Character Recognition Model, specifically tailored to the requirements of the institution, the application has surpassed existing state-of-the-art systems in terms of accuracy, precision, and recall. This accomplishment is a result of extensive research and the incorporation of modern techniques, positioning Marks2CSV at the forefront of marks data management.

One of the standout features of the Marks2CSV application is its speed of processing the papers, as given in the summary of the chapter 'Proposed System' the system can completely process 5 papers in just 13 seconds, which shows exceptional speed of application. By delivering outputs in the form of machine-readable and writable CSV files within a matter of seconds, the application significantly accelerates data processing workflows. This expedited turnaround time translates into enhanced productivity, allowing educational institutions to swiftly access and analyze marks data, driving informed decision-making and timely interventions when necessary.

‘

5.1 Future Scope

The Marks2CSV application has tremendous potential for further enhancements and expansion. Some of the future scopes include:

1. **Continuous Model Improvement:** Regular updates and enhancements to the underlying CNN model can further improve the accuracy and performance of the application, ensuring it stays at the cutting edge of technology.

2. **Mobile Application:** Developing a mobile application for Marks2CSV would enable educators and administrators to access and manage marks data on-the-go. This would provide flexibility and convenience, allowing them to efficiently perform tasks such as data entry, verification, and analysis directly from their mobile devices. The application forms a foundation for a Cloud service platform too.

3. **Integration with Cloud Services:** Leveraging cloud services paired with the application software would offer scalability and accessibility, enabling users to access and process marks data from anywhere, at any time, with enhanced security and reliability.

4. **Integration with Student Information Systems:** Seamless integration with existing student information systems (LMS, Moodle, etc.)used by educational institutions would automate the extraction of marks data, ensuring a smooth and streamlined workflow. This would enable faculties to rapidly upload and update data into Student Information Systems as soon as Marks2CSV generates the output.

5. **Advanced Data Analysis and Predictive Analysis:** By incorporating advanced statistical analysis and data visualization capabilities, the application can provide deeper insights and facilitate the discovery of trends and patterns within the marks data.

Integrating predictive analytics capabilities into the Marks2CSV application would enable institutions to forecast future academic performance based on historical marks data. This would facilitate proactive intervention strategies and personalized support for students at risk, ultimately improving overall student success rates.

These future scopes demonstrate the potential for the Marks2CSV application to evolve into a comprehensive and powerful tool for educational institutions, catering to their specific needs and driving further advancements in marks data management and analysis.

5.2 Limitations

This project aims to assist teachers and educational institutions in the efficient digitalization of handwritten marks on answer scripts. By implementing this system, users can benefit from significant time savings. However, it is important to acknowledge that the project does have certain limitations. The limitations are:

- The inability of the system to accurately detect decimal numbers, especially when they are represented using fractions, such as 0.5. This issue arises due to the different writing styles of teachers while writing decimal numbers (as in Figure 4.3 above) and the constraints of the algorithm employed for number detection.

- The efficiency of the Optical Character Recognition tool used in the project. OCR tools are designed to convert scanned images that contain handwritten digit into machine-readable text, but they can be susceptible to errors in certain scenarios. Specifically, the performance of the tool may be adversely affected by the presence of stray marks, corrections, or unsteady cell lines (as in above Figure 4.4 and Figure 4.5), which may result in inaccurate recognition or misinterpretation of the text.

- The impact of lighting conditions on the performance of the system. In instances where images are captured under poor lighting conditions, such as low light or uneven illumination, the system may encounter difficulties in accurately detecting and extracting tables and marks from the images.

- Limitations in terms of scalability and compatibility with different formats and systems. The system may work well with specific types of answer scripts but may

face difficulties in adapting to different formats used by various educational institutions. This can require additional customization or integration efforts to ensure compatibility across different systems.

References

[1] A.Raj, S.Sharma, J.Singh, A.Singh (Feb 2023), *Revolutionizing Data Entry: An In-Depth Study of Optical Character Recognition Technology and Its Future Potential*, International Journal for Research in Applied Science & Engineering Technology, Vol. 11 No.2, pp: 645-653.

[2] Md. Ajij, S.Pratihar, Diptendu S.Roy, T.Hanne (Feb 2022) *Robust Detection of Tables in Documents Using Scores from Table Cell Cores* SpringerNature Computer Science Journal, Vol.3, No.161, pp: 1-19.

[3] A. Arivoli, D.Golwala, R.Reddy (2022) *CoviExpert: COVID-19 detection from chest X-ray using CNN* Journal of Measurement: Sensors 23, pp: 1-8.

[4] Ömer Aydin (Jun 2021) *Classification of Documents Extracted from Images with Optical Character Recognition Methods*, Anatolian Journal of Computer Sciences, Vol.6 No.2 pp:46-55.

[5] S.Shrivatsava, Sanjeev K.Singh, K.Shrivatsava, V.Sharma (Mar 2021) *CNN based Automated Vehicle Registration Number Plate Recognition System* 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), IEEE Xplore, pp: 795-802.

[6] C.ShanWei, S.LiWang, Ng T.Foo, Dzati A.Ramli (2021) *A CNN based Handwritten Numeral Recognition Model for Four Arithmetic Operations* 25th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems, Procedia Computer Science, Vol.192, pp:4416-4424.

[7] Colin G.White-Dzuro, Jacob D.Schultz, C.Ye,Joseph R. Coco, Janet M. Myers, C.Shackelford, S.T.Rosenbloom,D.Fabbri (2021) *Extracting Medical Information from Paper COVID-19 Assessment Forms* Applied Clinical Informatics Vol. 12 No. 1, pp:170–178.

[8] J.Yuan, H.Li, M.Wang, R.Liu, C.Li, B.Wang (Sep 2020) *An OpenCV-based Framework for Table Information Extraction* 2020 IEEE International Conference on Knowledge Graph (ICKG), IEEE Xplore, pp: 621-628.

[9] A.Das, Gyana R.Patra, Mihir N.Mohanty (Sep 2020) *LSTM based Odia Handwritten Numeral Recognition* 2020 International Conference on Communication and Signal Processing (ICCSP), IEEE Xplore, pp: 538-541.

[10] A.Yaganteeswarudu (July 2020) *Multi Disease Prediction Model by using Machine Learning and Flask API* 2020 5th International Conference on Communication and Electronics Systems (ICCES), IEEE Xplore, pp: 1242-1246.

[11] Raajkumar G., Indumathi D. (July 2020) *Optical Character Recognition using Deep Neural Network* International Journal of Computer Applications, Vol. 176 No. 41, pp:61-65.

[12] B.Barz, J.Denzler (May 2020) *Deep Learning on Small Datasets without Pre-Training using Cosine Loss* 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE Xplore, pp: 1360-1369.

[13] J.Memon, M.Sami, Rizwan A.Khan, M.Uddin (August 2020) *Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR)* IEEE Access, Vol. 8, pp:142642-142668.

[14] B.Gatos, D.Danatsas, I.Pratikakis, S.J.Perantonis (2005) *Automatic Table Detection in Document Images* International Conference on Pattern Recognition and Image Analysis (ICAPR), Pattern Recognition and Data Mining, pp: 609–618.