IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Handwritten Character Recognition Using CNN

[1]Charan Pote , [2]Aman Khobragade , [3]Vedant Kapse ,[4]Ayushman Choudhary , [5]Amit Patel

[1]Professor, [2]Student, [3]Student,[4]Student,[5]Student

[1]Department of Computer Technology,

[1]Priyadarshini College of Engineering, Nagpur ,India

Abstract: In numerous industries, including banking, healthcare, and many others that deal with handwritten papers, handwritten character recognition has been implemented. The process of converting handwritten text into a machine-readable format is referred to as Handwritten Character Recognition. Even when written by the same individual, handwritten characters might differ in shape, size, and location, making it difficult to recognize them. To simplify the recognition of handwritten text, researchers have employed different methods, classifiers, and features.

In the present world, handwritten text recognition is practically indispensable. When we first started using this technology, we had to write texts by hand, which led to a lot of mistakes. Data administration in the physical realm required physical labour to stringently organize things, leading to inefficiency and wastefulness. In addition, long-term storage of data has been a disadvantage of traditional storing approaches. Fortunately, modern technology is able to provide us with the facility to efficiently manage, store, and access the data. By implementing Handwritten Text Recognition software, we have streamlined saving and fetching of data that was stored in the conventional way. On top of that, this technology also adds another security aspect to the data.

Keywords—CNN, RNN, CTC, TensorFlow

I. Introduction

The technique of effectively deciphering and transferring handwritten text into digital format is known as handwritten character recognition. This technology is essential for many applications, including the digitization of paper documents, automatic form processing, and improving accessibility for those with visual impairments.

To achieve accurate handwritten character recognition, advanced algorithms and machine learning techniques are employed. These algorithms analyze the structural and contextual features of handwritten text, enabling the system to recognize and classify individual characters accurately. The system aims to replicate human capabilities in understanding and interpreting different handwriting styles and variations where relevant characteristics of the handwriting, such as stroke shapes, spatial relationships, and line structures, are identified and measured.

The system uses machine learning models, such as hidden Markov models, support vector machines, and artificial neural networks, to map the extracted features to the proper character classes after feature extraction. These models are trained using large datasets of annotated handwriting samples, allowing them to learn the patterns and variations in different writing styles.

Once the classification is complete, the recognized characters are post-processed to improve the overall accuracy and readability. This may involve correcting any misclassified characters, smoothing out uneven strokes, and applying optical character recognition (OCR) techniques to handle variations in writing styles.

To sum up, handwritten character recognition is a complex process that uses machine learning methods and sophisticated algorithms to reliably translate handwritten text into a digital version. By understanding the original text's meaning and context, modifications can be made to create a fresh and unique paraphrased version while maintaining the intended message.

II. Problem Statement

One of the most intriguing and difficult study topics in the field of image processing and pattern recognition in recent years has been handwriting recognition. It makes a significant contribution to the development of automated processes and enhances the human-machine interface in a variety of applications. In today's environment, handwritten text recognition is a much-needed technology. Prior to the correct application of this technology, we relied on handwriting texts, which is prone to error. Efficient storage and retrieval of physical data poses challenges. To keep the data properly organized, manual labor is needed. The old way of storing data has resulted in significant data loss throughout history. Thanks to advances in technology, people can now save data on computers, which makes it easier to organize, store, and retrieve data.

Numerous studies have concentrated on novel approaches and strategies that would speed up processing while improving recognition accuracy.

The majority of businesses use handwritten customer forms, checks, and other documents to gather information.
These documents are then converted and stored in digital formats for easier retrieval or information gathering. Traditionally, handling this type of information involved manually entering the same data into a computer, which would be laborious and time-consuming. This has led to the need for specialized software called Handwritten Character Recognition (HCR) software, which will automatically recognize texts from images of documents.

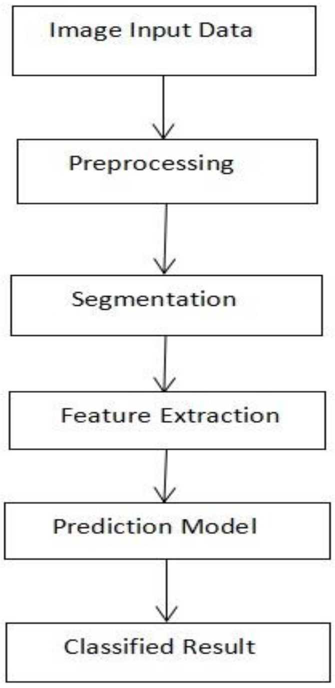
Applications for HCR systems can also be found in recently developed fields where handwritten data entry is necessary, such multimedia databases and electronic libraries.

III. Objectives

The goal of the handwritten character recognition system is to digitalize and convert the handwritten text into machine-readable text by implementing an approachable computer-assisted character representation that will enable the successful extraction of characters from handwritten documents.

IV. Project Methodology

A. Block Diagram



B. Steps For HCR

The steps of the Handwritten Character Recognition System are as follows:
1) Pre-processing.
2) segregation.
3) Extraction of features.
4) Training and Testing

Pre-Processing

Operations involving images at the lowest level of abstraction—both input and output are intensity images—are commonly referred to as pre-processing. These famous pictures are identical to the original sensor data; an intensity picture is often represented by a matrix of image function values (brightnesses). Pre-processing aims to improve the image data by suppressing unwanted distortions and enhancing certain image features that are crucial for further processing. However, since similar techniques are used, geometric image transformations, such as rotation, scaling, and translation, are included in the pre-processing methods category.

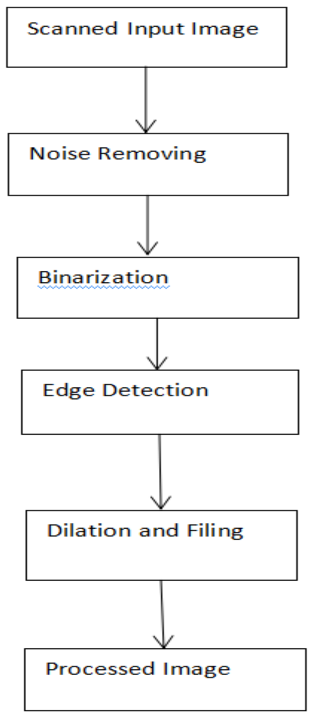


Fig 2: Preprocessing

Noise Removal
Skew correction of an image is executed in the pre processing phase to accurate the article text lines, threshold to switch a gray scale or color image into a binary image, and decrease of noise to decrease extraneous data as shown in Figure. The pre processing stage enhances the superiority of the raw image and places the data of significance. It is also acknowledged as pixel level or low level processing, which is arranged on the detained image to manage it for additional analysis.

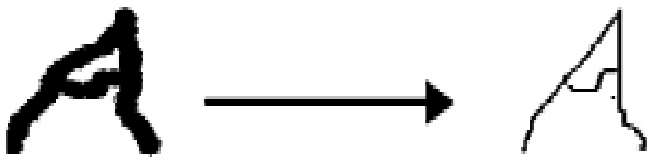


Fig 3: Noise Removal

Skew Correction
If the text lines in scanned document images are not horizontally aligned during the scanning process, skew correction is employed to fix them.

Segmentation
During the segmentation phase, an image with an alphabetic sequence breakdowns into smaller images containing a single character. Using a labeling approach, the pre-processed input image is divided into distinct alphabets by assigning a number to each alphabet. This label provides information about the quantity of characters present in the picture.
All characters are uniformly rearranged into 90 by 60 pixels in order to facilitate classification and identification.

Features Extraction
Systems that recognize handwriting rely heavily on features. Their primary objective is to effectively depict the data in order to raise the recognition rate. Feature extraction involves primarily obtaining the required information from image pixels based on the used data and the degree of difficulty to be overcome. There are two primary feature extraction techniques utilized in handwritten alphabet recognition systems: analytical and holistic. Every word is regarded as a class and recognized as a whole word in holistic recognition. However, character segmentation-free recognition forms the foundation of the analytical recognition approach.

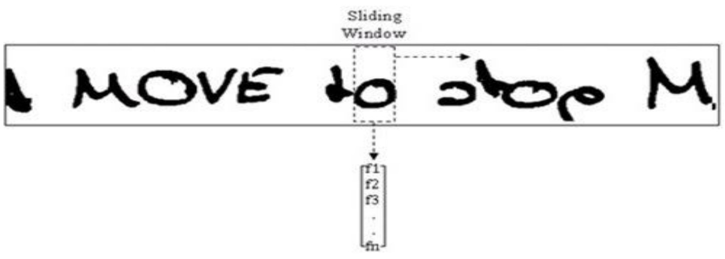


Fig 5: Feature Extraction using sliding window

Recognition

Handwritten English alphabet recognition is a very challenging problem. It is possible to write the English alphabet in many sizes, directions, widths, arrangements, and measurements. This might yield countless variations. The ability of neural networks (NNs) to simplify and become insensitive to lost information could prove to be highly beneficial in the identification of handwritten English alphabets. The assessment-building component of a recognition system, the categorization phase makes use of the traits eliminated in the preceding phase. The symbol for the feature vector is X, where X = (f1, f2,....., fd), where d is the number of features that have been eliminated from the English alphabet and f denotes characteristics. English alphabets are particularly effective because of their feature vector similarity.

English alphabets are powerfully sorted into appropriate categories and acknowledged based on feature vector similarity. Classifiers operate using one of two categories of learning strategies:

• **Supervised learning**: To train a novel model, supervised learning uses training data that accurately specifies a category. Information is tested for proper categorization using this innovative technique. Training data includes both the necessary input and output. The new model goes through a learning process, and then it learns and categorizes test data. For instance, SVM, HMM, etc.

• **Unsupervised learning**: Training data is not provided for this method of learning. There is no need to learn it. The method groups test data according to statistical traits, their spatial relationship, and the neighbor that is nearest to them.

C. Convolutional Neural Network

The majority of computation in a CNN takes place in the convolutional layer, which is its fundamental building component. Three things are needed: a feature map, a filter, and input data. Assume for the moment that the input will be a color image composed of a 3D matrix of pixels. This indicates that the three dimensions of the input—height, width, and depth—will match the RGB values in an image. Additionally, we have a feature detector—also referred to as a kernel or filter—that will scan the image's receptive fields to determine whether the feature is there. We call this process a convolution.

A two-dimensional (2-D) array of weights that represents a portion of the image serves as the feature detector. The filter size is usually a 3x3 matrix, however they can vary in size; this also establishes the size of the receptive field. After applying the filter to a portion of the image, the dot product between the input pixels and the filter is computed. An output array is then supplied with this dot product. The filter then moves one step forward and backward until the kernel has scanned the entire image. A feature map, activation map, or convolved feature is the result of the series of dot products created from the input and filter.

D. Connectionist Temporal Classification

In sequential applications where there is no explicit information about alignment between the input and output, such as handwriting and speech recognition, CTC is a technique used to train deep neural networks. In situations where we are unsure of how the inputs map to the outputs, CTC offers a workaround.There may not be a one-to-one correlation between the input and destination sequences in sequence-to-sequence challenges. As an illustration, Think about the Automatic Speech Recognition (ASR) job. An audio clip serves as the input, and the words that are transcribed are the output. The issue arises from the unknown word alignment in the transcription and the audio input. Furthermore, this alignment will vary from person to person. Think about the word "hello.".Saying "hello" or "hello" with emphasis on various word components is possible. Model training is challenging as a result. Labeling each alignment by hand would be an easy fix. However, this strategy is simplistic and unfeasible for huge datasets.

E. Modules

Modules used for this system are as follows

- OpenCV
- Tensorflow
- Keras
- Sklearn
- Numpy
- Matplotlib

OpenCV

A software library for computer vision and machine learning that is available for free is called OpenCV (Open Source Computer Vision Library). In order to facilitate the use of machine perception in commercial goods and to give computer vision applications a common foundation, OpenCV was developed. OpenCV's license for Apache 2 makes it simple for companies to use and alter the code.

More than 2500 optimized algorithms are available in the collection, including a wide range of both traditional and cutting-edge computer vision and machine learning techniques. These techniques may be used to monitor camera motions, track moving objects, detect and identify faces, identify items, classify human behaviors in films, extract 3D models of objects, create 3D point clouds from stereo cameras, recognize scenery, and more. With a preference for real-time vision applications, OpenCV uses MMX and SSE instructions when they are available. CUDA and OpenCV interfaces with full functionality are now under active development. More than 500 algorithms exist, and around ten times as many functions either support or comprise those algorithms. OpenCV features a templated interface that integrates well with STL containers and is written entirely in C++.

TensorFlow

TensorFlow is an open-source framework that uses data flow graphs for machine learning. The multidimensional data arrays (tensors) that flow between the graph's nodes, which reflect mathematical processes, are represented by the graph's edges. Machine learning algorithms can be represented as a graph of connected processes thanks to its adaptable architecture. Without changing code, they may be taught and used on a variety of platforms, including high-end servers, desktop computers, and portable devices, using GPUs, CPUs, and TPUs.

Keras

Keras is a Python deep learning API that can be used with either JAX, TensorFlow, or PyTorch.Keras is:
powerful – it offers industry-strength performance and scalability; it is used by NASA, YouTube, and Waymo.
Simple – but not simplistic. Keras reduces developer cognitive load to free you to focus on the parts of the problem that really matter. Flexible – Keras adopts the principle of progressive disclosure of complexity: simple workflows should be quick and easy, while arbitrarily advanced workflows should be possible via a clear path that builds upon what you've already learned.

Keras is a multi-framework API that enables the development of modular components that work with any framework, including PyTorch, TensorFlow, and Java.

Sklearn

A machine learning package for the Python programming language, scikit-learn (formerly known as scikits.learn and also called sklearn) is available as free software.With support-vector machines, random forests, gradient boosting, k-means, DBSCAN, and other classification, regression, and clustering techniques, it is compatible with the NumPy and SciPy scientific and numerical libraries for Python. Scikit-learn is a financially supported project by NumFOCUS.

Numpy

A vast set of high-level mathematical functions to operate on huge, multi-dimensional arrays and matrices, as well as support for these arrays, are provided by the NumPy library for the Python programming language. Jim Hugunin initially developed Numeric, the forerunner to NumPy, with assistance from a number of other developers. Travis Oliphant developed NumPy in 2005 by heavily altering Numeric and adding elements of the rival Numarray. NumPy is open-source software with a large community of developers. NumPy is a financially supported project by NumFOCUS.

Matplotlib

For the Python programming language and its NumPy numerical mathematics extension, Matplotlib is a graphing library. It offers an object-oriented API that may be used with general-purpose GUI toolkits such as Tkinter, wxPython, Qt, or GTK to embed plots into applications. Although its use is discouraged, there is another procedural "pylab" interface that is based on a state machine (like OpenGL) and is intended to closely resemble MATLAB.SciPy utilizes Matplotlib.

The original author of Matplotlib was John D. Hunter. It has an active development community and is released under a license akin to BSD since then. Thomas Caswell joined Michael Droettboom, who had been nominated soon before John Hunter passed away in August 2012, as matplotlib's principal developer.Matplotlib is a financially supported project by NumFOCUS.

V.Software Used

Jupyter IDE

Jupyter is a non-profit, open-source project, born out of the IPython Project in 2014 as it evolved to support interactive data science and scientific computing across all programming languages. Jupyter will always be 100% open-source software, free for all to use and released under the liberal terms of the modified BSD license.

Jupyter Notebook is a web-based application used to create and share interactive notebook documents, which can contain live code, text, data visualizations, videos and other computational outputs. Created by Project Jupyter, the application is open-source and supports the use of over 40 programming languages, including Python, R and Scala.

Jupyter Notebook showcases real-time code results and imagery, and can execute cells in any order. This makes it a useful tool for quick code experimentation, designing code presentations or facilitating data science workflows.

VI. Results and Discussions

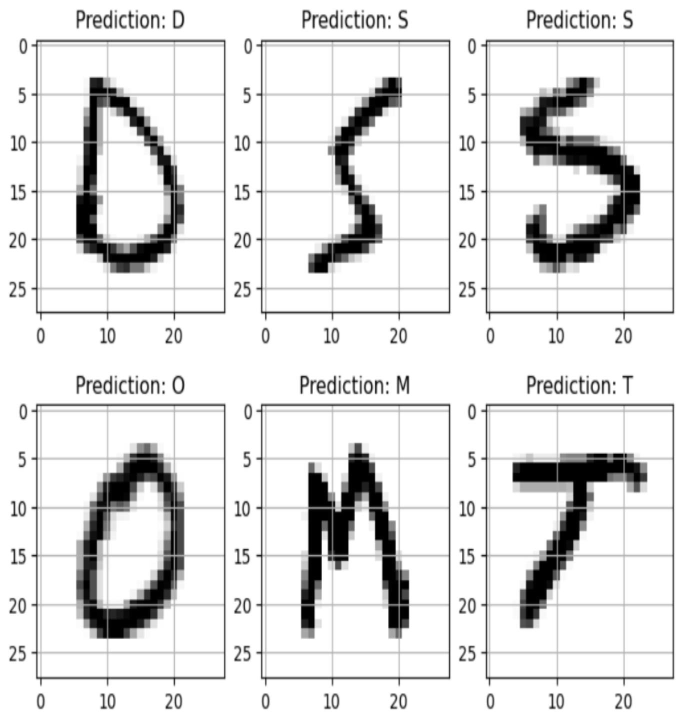
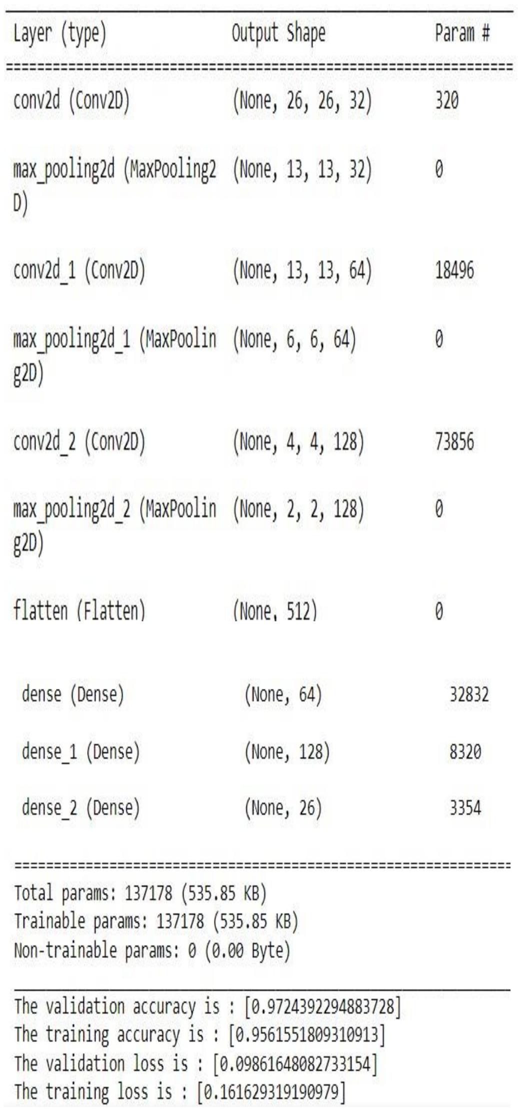


Fig: Prediction of Dataset Image



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 26, 26, 32)        320

max_pooling2d (MaxPooling2   (None, 13, 13, 32)        0
D)

conv2d_1 (Conv2D)            (None, 13, 13, 64)        18496

max_pooling2d_1 (MaxPoolin   (None, 6, 6, 64)          0
g2D)

conv2d_2 (Conv2D)            (None, 4, 4, 128)         73856

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

flatten (Flatten)            (None, 512)               0

dense (Dense)                (None, 64)                32832

dense_1 (Dense)              (None, 128)               8320

dense_2 (Dense)              (None, 26)                3354

=================================================================
Total params: 137178 (535.85 KB)
Trainable params: 137178 (535.85 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
The validation accuracy is : [0.9724392294883728]
The training accuracy is : [0.9561551809310913]
The validation loss is : [0.09861648082733154]
The training loss is : [0.161629319190979]
```

Fig: CNN Model Results

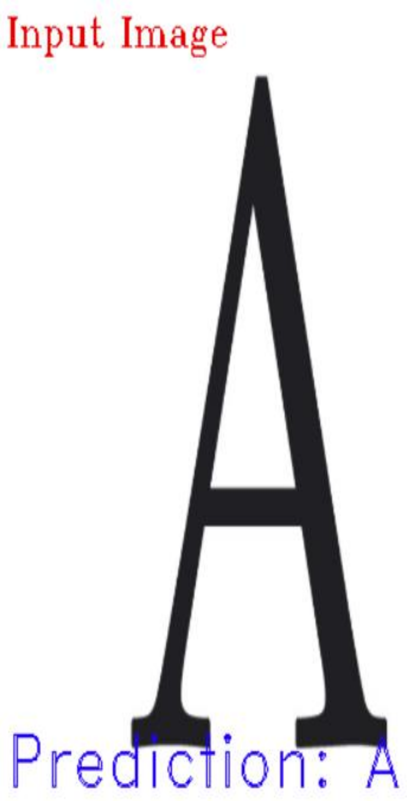


Fig: Prediction on External images

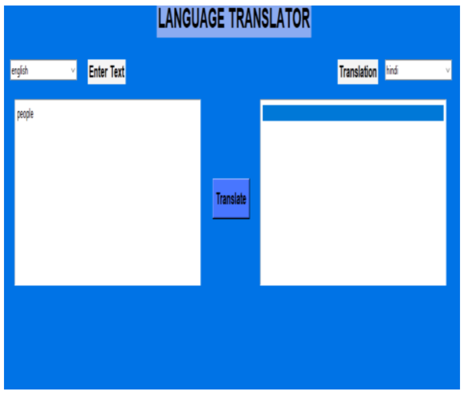


Fig : Translator

Conclusion

The potential of machine learning goes far beyond basic tasks like text recognition, thanks to modern techniques such as neural networks and deep learning. Traditional OCR used to rely on photo sensor technology to gather physical attributes and convert them into a database, but now, with convolution neural networks, We can scan and comprehend words with even greater accuracy. Many approaches have been investigated for the recognition of handwritten English alphabets throughout the years, including Pre-processing, segmentation techniques, feature extraction procedures, and classification methods. A workable software program is still required, nevertheless, in order to improve accessibility for handwritten English alphabets. One possible solution is to train a model to read entire handwritten or non-handwritten documents and convert the printed text into different languages.

References

[1] J Pradeep & Srinivasan, E. & Himavathi, S. (2011). Neural network-based handwritten character recognition system without feature extraction.

[2] Shah, Lipi & Patel, Ripal & Patel, Shreyal & Maniar, Jay. (2014). Handwritten Character Recognition using Radial Histogram.

[3] Sonkusare, Manoj & Sahu, Narendra. (2016). A Survey on Handwritten Character Recognition (HCR) Techniques for English Alphabets.

[4] Chowdhury, Rumman & Hossain, Mohammad & Islam, Raihan & Andersson (2019). Bangla Handwritten Character Recognition using Convolutional Neural Network with Data Augmentation.

[5] Memon, Jamshed & Sami, Maira & Khan, Rizwan. (2019). Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR).

[6] Manchala, Sri & Kinthali, Jayaram & Kotha, Kowshik & Kumar, Jagilinki. (2020). Handwritten Text Recognition using Deep Learning with TensorFlow.

[7] Khalkar, Rohini & Dikhit, Adarsh & Goel, Anirudh. (2021). Handwritten Text Recognition using Deep Learning (CNN & RNN).

[8] M J, Elizabeth & Joseph, Alex & vm, Praseetha & V.M, Athira. (2021). Handwritten Character Recognition using Deep Learning in Android Phones.