HANDWRITTEN TEXT RECOGNITION

Kartik Saini, Khushi Sharma, Akshaj Agarwal, Kishan Jayan
Mentor: Ms. Deepali Dev
Computer Science Engineering-Artificial Intelligence Machine learning Department
ABES Engineering College
Khushi.20B1531039@abes.ac.in
Akshaj.20b1531044@abes.ac.in
Kartik.20B1531055@abes.ac.in
Kishan.20B1531001@abes.ac.in

Abstract

The goal of this project is to give and output that displays the best textual representation of given handwritten characters. A system is also described that was built to recognize handwriting of many languages. The input to the system is an ink along with the language it should be interpreted in. Our primary focus is to build this system for mobile devices using touch screens but it can be built for powerful machines for recognition in the cloud as well. We aim to use machine learning as much as possible. We built results from existing techniques. Some challenges in hand writing recognition are like strong variability in writing style between different groups of people, variability in writing speed, sloppiness and ambiguities like accidental apostrophe or dot. Combination of time and position based interpretation of input is implemented to recognize overlapping writing and delayed marks. A Segmenter is trained as a filter to determine which hypothetical cut points are valid character segmentation. CNN(Convolutional Neural Network) as a powerful feature can also be applied to extract feature of a handwritten character and linear SVM (Support Vector Machine) as a high end classifier. In this project we would be creating a model that would be used to read handwriting digits, characters and words from the image using the concept of CNN and an essential technique known as elastic distortion that vastly expand the size of training set.

Keywords: Segmenter, CNN (Convolutional Neural Network),SVM(Support Vector Machine),Machine Learning, Character Segmentation, Elastic Distortion, Character Segmentation.

Introduction

Handwriting Recognition or Handwritten Text Recognition is the ability of a computer to receive and interpret hand written input from sources like documents, photographs, touch-screen and other devices. The image of the written text may be sensed "off line" from a piece of paper by optical character recognition. Similarly, the movements of the pen tip may be sensed "on line", for example by a pen-based computer screen surface. There exists several techniques to recognize handwriting depending on the language but some of them are accurate and some not. Most of the existing systems use machine learning mechanisms such as neural networks. Some of the frequently used techniques involves feature extraction, segmentation, classification, training on dataset, recognition. Handwriting recognition has been a challenging task because recognition depends on various factors like different people have different style of writing, size and shape of characters varies from person to person. Language dependency is another challenge as the shape of the characters takes new forms. Also some accidental dots or apostrophe may create ambiguity during recognition. But vast research is still going on and improvements are being made on the handwriting recognition system.

Problem Statement

The goal is to create a model which can recognize the digits, it can be extended to letters and an individual's handwriting. The major goal of the proposed system is understanding Convolutional Neural Network, and applying it to the handwritten recognition system.

Literature Review

PAPER NAME	DATASET	TECHNIQUE	RESULT
[1]Handwriting Recognition On Form Document Using Convolutional Neural Network and Support Vector Machines (CNN-SVM)	NISTSD 192nd edition dataset. It consists of numeral uppercase lowercase and merger of upper case and lowercase.	Convolutional Neural Networks and Support Vector Machine.	CNN as a powerful feature extraction method applied to extract the feature of the handwritten characters and linear SVM using L1 loss function andL2regularizationusedasendclassifier.
[2]Multi-Language Online Handwriting Recognition	Single-Character Experiments on UNIPEN-1-R01/V07 version of the UNIPEN-1 data with a fixed 2:1 random partition into training and test data Word Recognition Experiments on IAM-On DB-performed experiments on the test set IAM-On DBt2 of the IAM database that contains3,859items.	Preprocessing Search lattice creation, lattice coding, training	Presented the architecture of a real-world, multi-script and multi-language online handwriting recognition system. The system combines several existing and some new components. A key emphasis of the system is on there use of components across many scripts and languages, which makes the problem tractable as an engineering problem.
[3]Offline Handwriting Recognition Using LSTM Recurrent Neural Networks	The training dataset consisted of 118 scanned pages of handwritten medieval Latin texts from two sources-KNMP Chronicon Boemorumand Stanford CCCC	Recurrent Neural Networks, Connectionist Temporal Classification approach, Sequence-to-Sequence Learning approach.	Sequence to Sequence learning approach was better at predicting short words, but CTC model had higher accuracy predicting longer words.
[4]Fastmulti-language LSTM-based online handwriting recognition	IAM-On DB contains forms of handwritten English text acquired on a whiteboard.IBM-UB-1 contains free form cursive handwritten pages in English. And their internal dataset.	Bidirectional long short-term memory recurrent neural networks	Described the online handwriting recognition system that is currently in use at Google for 102 languages in 26 scripts. The system is based on an end-to-end trained neural network. Recognition accuracy of the new system improves by 20–40%.
[5]Handwritten Character Recognition using CNN	EMNIST dataset which consists of English alphabets and numbers are made use of to train the neural network.	Normalizing the pixel values.	The work is extended by adding some more datasets to the EMNIST dataset of characters from the tamil language and training the model.

[6]HANDWRITTEN TEXT RECOGNITION with Deep Learning and Android	The EMNIST dataset is a collection of hand written alphanumeric derived from the NIST Special Database.	Using modern day techniques like neural networks to implement deep learning	The aim of this project is to make an application for mobile devices that can recognize handwriting using concepts of deep learning.
[7]Handwrittencharacter Recognition using convolutional neural network	NIST	In this research, a deep learning technique CNN is implemented for handwritten character recognition.	It was found that the accuracy obtained from 200 training images as 65.32% is improved gradually with increasing training images.
[8]HandwrittenCharacter Recognition Using Deep-Learning	NIST	Open CV for performing Image processing and have used Tensorflow for training a Neural Network.	Designed a image segmentation based on Handwritten character recognition system. An Android application was developed using which the user can click a photo of handwritten text using their camera.
[9]Online handwriting Recognition system using UNIPEN-online handwritten training set and MNIST training set.	MNIST and UNIPEN.	Back Propagation, Segmentation, Multi Neural Networks, Elastic Distortion.	The proposed model possessed the capability of creating efficient and flexible recognition system for large pattern set such as English Character set etc.
[10]Accurate Acoustic based handwriting recognition system using deep learning.	Acoustic signals generated by pen.	Segmentation, Classification, Word Suggestion.	The Word Recorder extracts the valid temporal-frequency of the handwriting sound to recognize the user's input letter and uses word suggestion algorithm.
[11]Handwriting Recognition using Artificial Intelligence and Image Processing	The dataset consist of handwritten texts for evaluating machine learning models.	Image Acquisition and Digitization, Pre-processing, Segmentation.	After training the machine learning model with the dataset, it was able to give an accuracy of83.4%.
[12]RNN based Online Handwriting Recognition in Devanagari and Bengali Script using horizontal zoning.	In online handwriting recognition systems data is collected from graphic tablets.	Horizontal Zoning(Zone Segmentation Approach),LSTM,BLSTM.	Using the segmentation process both the scripts gave an accuracy of 99.40%(Devanagari) and 97.67%(Bengali).

Tools Used

To design this handwritten test recognition system we have used various tools like python, OpenCV , TensorFlow and Convolutional Neural Networks(CNN).

Required Frameworks are:

- Numpy (version 1.16.5)
- cv2 (openCV) (version 3.4.2)
- Keras (version 2.3.1)
- Tensorflow (Keras uses TensorFlow in backend and
- for some image preprocessing) (version 2.0.0)
- Matplotlib (version 3.1.1)
- Pandas (version 0.25.1)

The dataset contains 26 folders (A-Z) containing handwritten images in size 2828 pixels, each alphabet in the image is center fitted to 2020 pixel box. The images are taken from NIST (https://www.nist.gov/srd/nist-special-database-19) and NMIST large dataset and few other sources which were the formatted as mentioned above.

Methodology

We would be following the above mentioned procedures in the flowchart for our implementation of online handwritten text recognition on jupyter notebook platform. We would be importing libraries useful for the implementation of CNN and will be following different processes such splitting the dataset into images, Reshaping of the dataset, shuffling of the data finally implementing the Convolutional Neural Network to extract features of images.

Finally we would be train and validate the CNN module against `our testing dataset to gain a range of accuracies and losses .In the end we will perform predictions on our test dataset and any external data.

Implementation

First of all, we do all the necessary imports including:

1.matplotlib.pyplot
2.cv2
3.numpy
4.keras.models
5.keras.layers
6.tensorflow.keras.optimizers
7.keras.callbacks
8.tensorflow.kera.utils

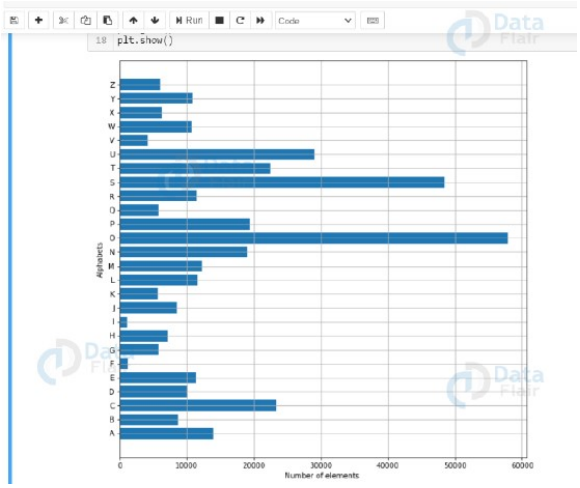
Now we are reading the dataset using the pd.read_csv() and printing the first10 images using data.head(10).

1)Split data into images and their labels:
Splitting the data read into the images & their corresponding labels. The '0' contains the labels, & so we drop the '0' column from the data dataframe read & use it in the y to form the labels.

2)Reshaping the data in the csv file so that it can be displayed as an image:
In the above segment, we are splitting the data into training & testing dataset using train_test_split().
Also, we are reshaping the train & test image data so that they can be displayed as an image, as initially in the CSV file they were present as 784 columns of pixel data. So we convert it to 28×28 pixels.

All the labels are present in the form of floating point values, that we convert to integer values, & so we create a dictionary word_dict to map the integer values with the characters.

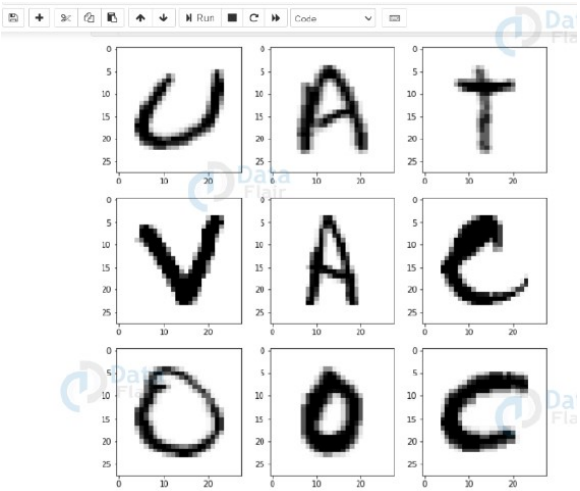
3)Plotting the number of alphabets in the dataset:



Here we are only describing the distribution of the alphabets.
Firstly we convert the labels into integer values and append into the count list according to the label. This count list has the number of images present in the dataset belonging to each alphabet.

Now we create a list – alphabets containing all the characters using the values() function of the dictionary. Now using the count & alphabets lists we draw the horizontal bar plot.

4)Shuffling the data:
Now we shuffle some of the images of the train set.



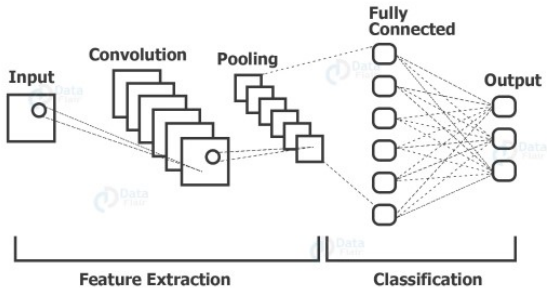
The shuffling is done using the shuffle() function so that we can display some random images.
We then create 9 plots in 3×3 shape & display the threshold images of 9 alphabets.

(The above image depicts the grayscale images that we got from the dataset)

5) Data Reshaping:
Reshaping the training & test dataset so that it can be put in the model. Now we reshape the train & test image dataset so that they can be put in the model.
New shape of train data: (297960, 28, 28, 1)
New shape of train data: (74490, 28, 28, 1)

Now we convert the single float values to categorical values. This is done as the CNN model takes input of labels & generates the output as a vector of probabilities.

6)CNN:



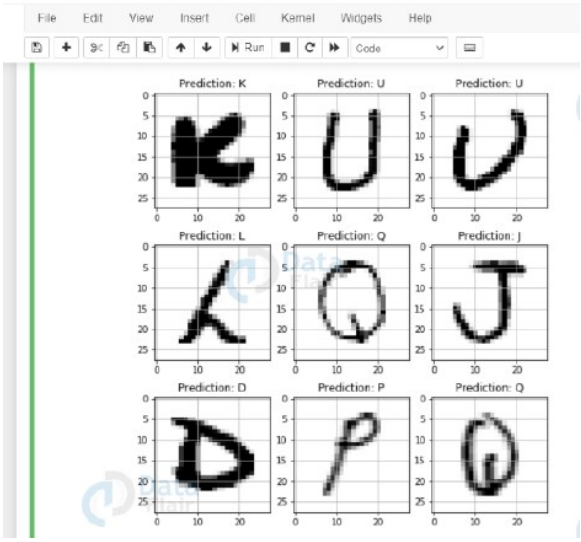
CNN stands for Convolutional Neural Networks that are used to extract the features of the images using several layers of filters.

The convolution layers are generally followed by maxpool layers that are used to reduce the number of features extracted and ultimately the output of the maxpool and layers and convolution layers are flattened into a vector of single dimension and are given as an input to the Dense layer (The fully connected network).

7)Compiling & Fitting Model:
Now we are compiling the model, where we define the optimizing function & the loss function to be used for fitting.
The optimizing function used is Adam, that is a combination of RMSprop & Adagram optimizing algorithms.
The dataset is very large so we are training for only a single epoch, however, as required we can even train it for multiple epochs (which is recommended for character recognition for better accuracy).

Now we are getting the model summary that tells us what were the different layers defined in the model & also we save the model using model.save() function.
We train & validate accuracies along with the training & validation losses for character recognition.

8)Predictions on Test Data & External Image:



Here we are creating 9 subplots of (3,3) shape & visualize some of the test dataset alphabets along with their predictions, that are made using the model.predict() function for text recognition.

Now we have read an external image that is originally an image of alphabet ‘B’ and made a copy of it that is to go through some processing to be fed to the model for the prediction that we will see in a while.

The img read is then converted from BGR representation (as OpenCV reads the image in BGR format) to RGB for displaying the image, & is resized to our required dimensions that we want to display the image in.

Now we do some processing on the copied image (img_copy).

We convert the image from BGR to grayscale and apply thresholding to it. We don’t need to apply a threshold we could use the grayscale to predict, but we do it to keep the image smooth without any sort of hazy gray colors in the image that could lead to wrong predictions.

The image is to be then resized using cv2.resize() function into the dimensions that the model takes as input, along with reshaping the image using np.reshape() so that it can be used as model input.

Now we make a prediction using the processed image & use the np.argmax() function to get the index of the class with the highest predicted probability. Using this we get to know the exact character through the word_dict dictionary.
This predicted character is then displayed on the frame.

Finally we set up a waitKey in a while loop that will be stuck in loop until Esc is pressed, & when it gets out of loop using cv2.destroyAllWindows() we destroy any active windows created to stop displaying the frame.

Gap Analysis

After reading several Research Papers we have identified some shortcomings which can certainly be modified in the future.

Some ROI are difficult to be recognized, it happened on ROI which contain the connected character. It is hard to do the segmentation with the CCL method because the CCL segmentation is based on the connectivity principle.

Systems trained on IBM-UB-1 or IAM-On D Balone perform significantly worse on our internal datasets, as our data distribution covers a wide range of use-cases not necessarily relevant to, or present, in the two academic datasets: sloppy handwriting, overlapping characters for handling writing on small input surfaces, non-uniform sampling rates, and partially rotated inputs. The network trained on the internal dataset performs well on all three datasets.

Neural networks are quick to set up however, they can be inaccurate if they learn properties that are not important in the target data.

Without the use of EMNIST data set it would be practically impossible to achieve this accuracy.

Exceeding the number of training images would lead to numerical errors and constraints on the CNN.

Different people have different styles of writing. There are lot of characters like Capital letters , Small letters , Digits and Special symbols. Thus a large dataset is required.

Use of spell checker and a voting module.

This model only identifies upper case letters but not lower case letters.

Use of this model can be extended to character recognition for other languages.

Misclassification between two similar shaped characters.

Conclusion

Code:
We have successfully developed Hand written character recognition (Text Recognition) with Python, Tensorflow, and Machine Learning libraries. Handwritten characters have been recognized with more than 97% test accuracy. This can be also further extended to identifying the handwritten characters of other languages too.

Research:
Based on the 12 research papers we have read so far , we have observed that in order to tackle different challenges in handwriting recognition like variation in in style of writing, variation in shape and size of characters, accidental comas and dots, overwriting etc. different types of algorithms,architecture,methods have been used in those problems respectively to improve accuracy of recognition. there are variations too in accuracy of algorithms during training of models on different datasets and vice versa.
Some of them are still not giving good accuracy but heavy research is still going on.

Although Researchers have gained a significant progress in recognizing handwritten text as compared to two decades back and is expected to improve in future. We will also try to implement some existing methods to compare accuracy on various datasets.

Acknowledgement

We take this opportunity to thank our teachers and friends who helped us throughout the project.

First and foremost, we would like to thank our guide for the project Ms. Deepali Dev (Computer Science Department-Artificial Intelligence and Machine

Learning) for her valuable advice and time during development of project.

We would also like to thank Dr. Pankaj Kumar Sharma(HOD, Computer Science Department) for his constant support during the development of the project.

References

1. Darmatasia and Mohamad Ivan Fanany(2017). Handwriting Recognition on Form Document Using Convolutional Neural Network and Support Vector Machines(CNN-SVM).
2. Daniel Keysers, Thomas Deselaers, Henry Rowley, Li-Lun Wang, and Victor Carbune(2016). Multi-Language Online Handwriting Recognition.
3. Yaroslav Shkarupa, Roberts Mencis, Matthia Sabatelli (2016). Offline Handwriting Recognition Using LSTM Recurrent Neural Networks.
4. Victor Carbune, Pedro Gonnet, Thomas Deselaers, Henry A, Rowley, Alexander Daryin, Marcos Calvo, Li-Lun Wang, Daniel Keysers, Sandro Feuz, Philippe Gervais (2020).Fast multi-language LSTM-based online handwriting recognition.
5. Anandh Kishan, J. Clinton David (2018).Handwritten Character Recognition Using Cnn.
6. Shubham Sanjay Mor, Shivam Solanki, Saransh Gupta, Sayam Dhingra, Monika Jain, Rahul Saxena (2019). HANDWRITTEN TEXTRECOGNITION: with Deep Learning and Android.
7. Khandokar, MdMHasan, FErnawan ,MdSIslam, M N Kabir (2020). Handwritten character recognition using convolutional neural network.
8. Rohan Vaidya1 , Darshan Trivedi1 , SagarSatra1, Prof. Mrunalini Pimpale2 (2018).Handwritten Character Recognition Using Deep-Learning.
9. Dũng Việt Phạm(2017). Online handwriting recognition system using UNIPEN-online handwritten training set and MNIST training set.
10. Haishi Du, Ping Li, Hao Zhou, Wei Gong, Gan Luo , Panlong Yang(2018). Accurate Acoustic based handwriting recognition system using deep learning.
11. Sara Aqab, Muhammad Usman Tariq(2020). Handwriting Recognition using Artificial Intelligence and Image Processing.
12. Rajib Ghosh, Chirumavika Vamshi, Prabhat Kumar (2020). RNN based Online Handwriting Recognition in Devanagari and Bengali Scripts using horizontal zoning.