Text Recognition Using Machine Learning

Prathamesh Yakkeldevi, Akshay Manekari, Saitarun Patel, Shahrukh Shaikh,
Shrikant Gullapalli, Prof/Guide Maindargi LC
Department of Computer Science and Engineering
N.B. Navale Sinhgad College of Engineering Solapur.
aapathak.nbnscoe@sinhgad.edu

Abstract

The field of Computer Science is vast in scope and is basis of all the modern IT and Tech Industries. The role of IT and High Tech is of unprecedented importance in the World. The Artificial Intelligence and Machine Learning and emerging sub fields of computer science and the ability they are providing to perform the desired work/task is ever efficient and resourceful. Our project capitalizes on the modern findings in Artificial Intelligence and Machine Learning to easify the digitization process. The basic idea is to provide our software with an Image File that contains textual information and it will be processed to extract that text information into an actual editable dot txt file. This can revolutionize the the current methodology employed of manual typing required for digitizing certain portions of scripts or books and digitization of old records written using typewriter machines. Also in future our project can be expanded into digitizing the handwritten scripts and work.

Keywords : Artificial Intelligence, Machine Learning, Text Recognition, GUI, Image croping, Black and white images, Computer Science, ML models, Image processing

1. Introduction

At the dawn of twenty first century mankind witnessed the phenomenon of Internet which is rapidly changing the World. At the core of this rapid expansion of internet are the digital electronic devices called computers. Today we live in the era that can be remarked as an Digital Epoch. The old methods of doing work with paper and pen are fading and are being replaced with digital methods. Furthermore with the help of Artificial Intelligence the tedious and repetitive tasks are being automated. Today digitization is everywhere. With all this in mind we decided to simplify and enhance this process of digitization. Although digitization can mean many things and can be enhanced in many ways. We focused on the idea that will enable anyone anywhere who wants to digitize their work to simply take its picture and get all the information into it as a text file. This idea through our project we have turned into a software which does the tedious, repetitive and unproductive work job for the user with the help of Artificial Intelligence and Machine Learning

2. Related Work

Artificial neural networks (ANNs), usually simply called Neural Networks (NNs), are
computing systems vaguely inspired by the biological neural network that constitute
animal brains. An ANN is based on a collection of connected units or nodes called artificial neurons,
which loosely model the neurons in a biological brain. Each connection, like the

synapses
in a biological brain, can transmit a signal to other neurons. An artificial neuron that receives a signal then processes it and can signal neurons connected to it. The "signal" at a connection is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. The connections are called *edges*. Neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times.

Components:

1.Neurons:

ANNs are composed of artificial neurons which are conceptually derived from biological neurons. Each artificial neuron has inputs and produces a single output which can be sent to multiple other neurons. The inputs can be the feature values of a sample of external data, such as images or documents, or they can be the outputs of other neurons. The outputs of the final output neurons of the neural net accomplish the task, such as recognizing an object in an image.

To find the output of the neuron, first we take the weighted sum of all the inputs, weighted by the weights of the connections from the inputs to the neuron. We add a bias term to this sum. This weighted sum is sometimes called the activation. This weighted sum is then passed through a (usually nonlinear) activation function to produce the output. The initial inputs are external data, such as images and documents. The ultimate outputs accomplish the task, such as recognizing an object in an image

2. Connections and weights:

The network consists of connections, each connection providing the output of one neuron as an input to another neuron. Each connection is assigned a weight that represents its relative importance. A given neuron can have multiple input and output connections.

3. Propagation function:

The propagation function computes the input to a neuron from the outputs of its predecessor neurons and their connections as a weighted sum. A bias term can be added to the result of the propagation

Neural Networks and Deep Learning:

Logistic Regression: The learning algorithm used when the output label y in supervised learning problems are either 0 or 1.

■ Cost function: For calculating the difference between the output of the model and output label y.

■ Gradient Descent: Minimizes the cost by updating weight and bias.

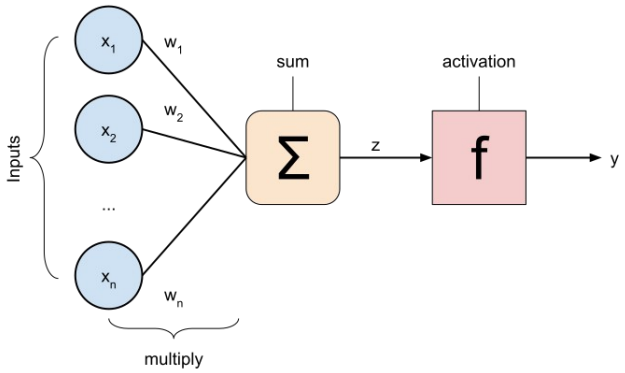

Figure : Represents Single Neuron

3. System Model

Our System model / Project Architecture is well described by given below figure. In

order for our model to work Successfully and to obtain desired result following assumptions are made : the image given for processing should be of PNG, JPEG, JPG format only, the image should contain only text, text should be English Alphabet A-Z and a-z, no overlapping of text is allowed.

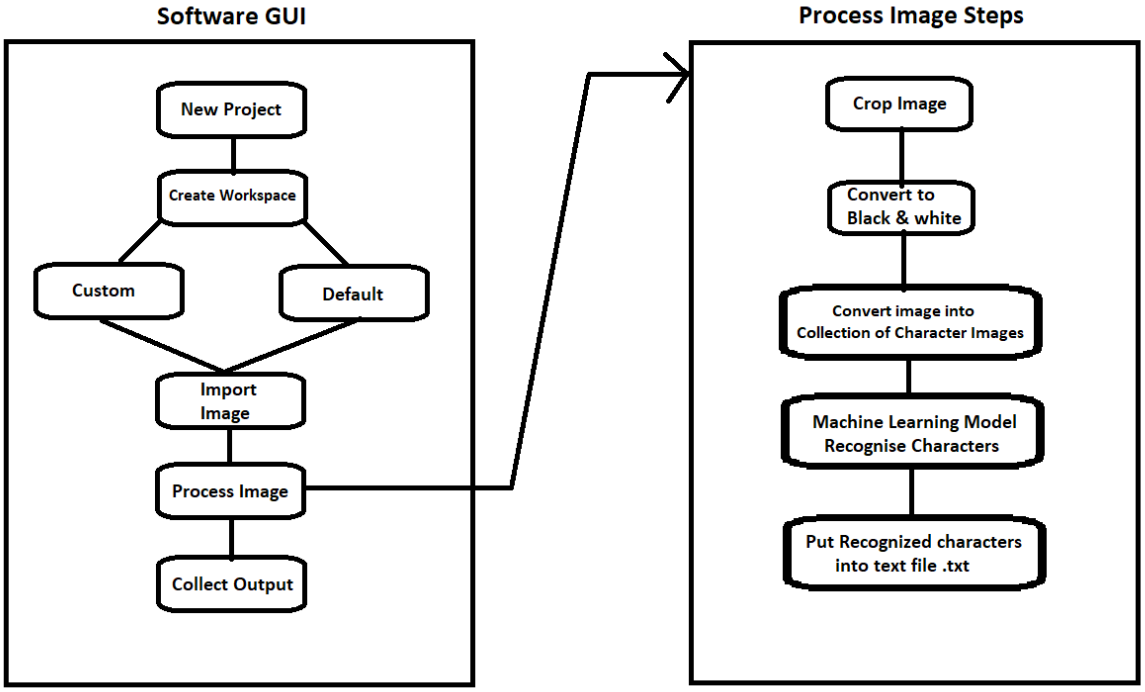

Figure : Represents System Model/Project Architecture

4. Problem Statement

Our project intends to make easy or simplify the process of digitization which in general has wide scope. More specifically our software's task is to take the image from the user and extract the all the information contained in it and export it to an editable text file.

5. Solution

The solution of the problem statement resulted into a software which is divided into five phases:

1. Graphical User Interface (GUI) - Any software, despite how resourceful and faster it is at executing the desired task, will be in the little interest of the End User if that software is difficult and not convenient to use. This phenomenon can be observed with wide popularity of Windows operating system over the Linux based operating systems. The user's convenience can be fulfilled by the equipment of the software with an easy to use Graphical User Interface (GUI). Hence we have created a simple and easy to use GUI for End User to use our software/project who can be of any background and not necessarily from an IT background.

2. Cropping Image - As we know, cropping is one of the most basic photo manipulation processes, and it is carried out to remove an

unwanted object or irrelevant noise from the periphery of a photograph, to change its aspect ratio, or to improve the overall composition. The image may contain unwanted sections, through the process of cutting we can eliminate this section. The unwanted sections in the image can hamper the process of black and white image conversion which is the next step therefore it's better to eliminate those sections in the initial phase itself.

3. Converting Colorful Image into Black and White - Furthermore, the next phase in the model is related to black and white image conversion. The process of predicting characters from the images becomes a lot easier if the images are in black and white instead of colour. The subsequent phase of character cutting embeds a technique that only works with B/W images. This phase is directly correlated to all the other phases because the B/W image that is generated through this phase acts as input to all the next phases.

4. Converting Image to Collection of separate Character Images - The next phase is to recognize characters from images which consist of text. To locate a character we can go by scanning images and locating characters.

5. Machine Learning Prediction Model Recognize Characters - The final phase uses a classification model with 26 classes of small alphabets ‘a’ to ‘z’. For training, we used a dataset containing ~26k images of 26 classes. This classification model classifies each image created by the previous phase and saves it in a document.

6. Analysis

As we discussed in solution part the methodology we implied to develop our project is divided into five phases and methods used for development of five respective phases have both quantitative and qualitative reasoning behind them. For phase one the graphical user interface development we used python programming language and many of its libraries in particular Tkinter is used most. The using of python as platform for GUI development finds its reason in python being very rich in built-in and third party libraries which comes in handy the make better software in short span of time which is reliable. The phase two is cropping the unwanted sections of the image. This is necessary to save the computational power which otherwise will be wasted in processing the parts of image that are not required. For phase three converting images to Black and White, the built in method in python to convert images to Black and White are static. They basically cant be used if image has sections with varying brightness intensities. So there was a need of a converter that analyses the brightness level of the images and carry out the process of conversion accordingly. The converter that we have implemented sits right in to overcome the drawbacks of the static built-in methods. The images that are captured in normal indoor conditions and processed through this converter turn out to be better. Although this converter needs some tuning to avoid noise if the images are captured in low light condition. Phase four is turning image into collection of character images for that we tried to built CNN to predict word position from images, for this we first used YOLO algorithm. But YOLO algorithm needed larger datasets and higher computation power. We were limited by both of this constraints so we designed a method which can be able to extract words and subsequently characters from it. This method uses low computational power. In

this method we are not building CNN so we were able to override both computation and power constraints. The last phase viz phase five is recognizing the characters from their images to do that we made for models using the same training set for each model and trained each model for different amounts of epochs. Our aim for making four models was to find the most accurate model to be used in our project. After training each model on different datasets we choose the one which provided maximum accuracy.

7. Simulation and Experimentation

Cropping Image - To process the mouse click events we define the click_and_crop
The callback function is a mouse event that happens, OpenCV will relay the pertinent details to our click_and_crop function.

The experimentation of B/W code was carried through passing different images with different lighting conditions. The following are the results,

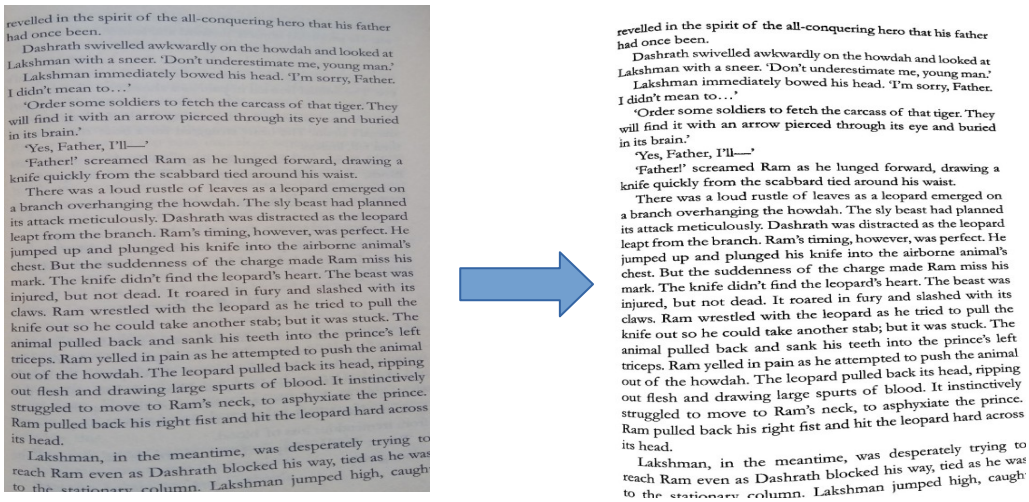

Figure : Represents Image converted to B/W

At first we tried to scale character image to 28*28 dimension. And trained Neural Network for this dimensions of image. The neural network was not able to extract the useful feature From image, so we generated more data set and trained our model on bigger data set, which was not enough. So we come up with another idea

-> We can scan image from left to right and ones we encounter Black Pixel, we can recursively move it to another blank image using flood fill algorithm.

By this two characters can be separated.

This solution was good enough for tackling this issue. The disadvantage or issue with this idea was that if any character is broken then it would consider it as two separate characters

The solution of this was to use good technique for converting RGB input image to Black and white. Because while converting RGB image to black and white, some character were breaking apart.

We made 4 models using the same training set for each model and trained each model for
different amounts of epochs. Our aim for making 4 models was to find the most accurate model to be used in our project. After training 4 models, we experimented with threshold values to increase the accuracy because if there were any images with multiple characters or any other symbol other than the characters trained to the model, in theory their prediction values

won’t be higher than the threshold value.

8. Conclusion

The convenience of using the software of the End User is of paramount importance which can be fulfilled with simple and easy to use Graphical User Interface (GUI). Although there are many ways for developing GUI the best/easiest way we found is using Python Programming Language which comes with an abundance of built-in and third party libraries.

The B/W image converter code has a hard time to deal with the “Low light and Bright light images”,So.the images need to be clicked in normal indoor lighting conditions and passed to the code.
Before giving an image to the code; we need to make sure that the image is properly cropped so that it helps to dampen the noise(unwanted black pixels).
From the technical front it can be concluded that the images with alpha values in the range (40 to 180) are good for the converter and can result in better outputs.

The classification model works well if proper images are provided to it, but still some inaccuracies may still persist while dealing with half visible characters, multiple characters in a single image, or plain inaccurate predictions like sometimes unable to
distinguish between ‘b’ and ‘h’ or ‘e’ and ‘o’ when these characters are distorted in shape.

9. References / Bibliography

- Deep learning Specialization by AndrewNg :
https://www.coursera.org/specializations/deep-learning
- Training set:
http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/
- Grokking Deep Learning - Andrew Trask
-
https://www.researchgate.net/figure/Commonly-used-activation-functions-a-Sigmoid-b-Tanh-c-ReLU-and-d-LReLU_fig3_335845675
-
https://www.researchgate.net/figure/Commonly-used-activation-functions-a-Sigmoid-b-Tanh-c-ReLU-and-d-LReLU_fig3_335845675
-
https://www.researchgate.net/figure/Commonly-used-activation-functions-a-Sigmoid-b-Tanh-c-ReLU-and-d-LReLU_fig3_335845675
-
https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6
- https://towardsdatascience.com/the-vanishing-exploding-gradient-problem-in-deep-neural-networks-191358470c11
- https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
- https://medium.com/@pechyonkin/key-deep-learning-architectures-lenet-5-6fc3c59e6f4
- https://neurohive.io/en/popular-networks/alexnet-imagenet-classification-with-deep-convolutional-neural-networks/
- https://neurohive.io/en/popular-networks/vgg16/