TEXT DETECTION IN IMAGES

SHAHANA.V.M

DATE:

EMAIL: ayshashahana006@gmail.com

CONTENT

* Abstract

* Objective

* Technology Used

* Libraries And Packages Used

* Algorithm

* Data Collection

* Result

* Conclusion

ABSTRACT

Extracting text from images is a very popular task in the operations units of the business (extracting information from invoices and receipts) as well as in other areas. Text recognition in images is an active research area which attempts to develop a computer application with the ability to automatically read the text from images. Nowadays there is a huge demand of storing the information available on paper documents in to a computer readable form for later use. In this paper, we have reviewed and analysed a method for text recognition from images. Detecting text from an image is an important prerequisite for the content based image analysis process. To understand the contents of an image or the valuable information, there is need of analysing the text appears in it. Various methods have been proposed over past years for text detection and extraction from different types of images, like scene image, born digital image and document image. In this paper, we describe the existing methods of text detection. OCR (Optical Character Recognition) is an electronic computer-based approach to convert images of text i machine-encoded text, which can then be extracted and used in text format.

OBJECTIVE

The objective of this project is to create a machine learning model to detect text from images. Extracting text from images has found numerous applications. Some of the applications are Passport recognition, automatic number plate recognition, converting handwritten texts to digital text, converting typed text to digital text, etc.

TECHNOLOGY USED

PYTHON

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small- and large-scale projects. Python is open-source, so it is free to use, modify and distribute the python source code. Python is a high-level programming language that has English-like syntax making it easier to read and understand the code also the standard library of python is very big, that any and all function needed for a project can be found minimizing the use of external libraries. Different python packages like numpy, pandas, matplotlib, opencv will used in the project.

MACHINE LEARNING

Machine learning is the study of computer algorithms that can improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to identify without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications where it is difficult to develop conventional algorithms to perform the needed tasks. The model which detects text from images will be built using machine learning algorithm

LIBRARIES AND PACKAGES USED

NUMPY

numpy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high -level mathematical functions to operate on these arrays. It is an open-source software and has many contributors.

MATPLOTLIB

matplotlib is an amazing visualization library in Python for 2D plots of arrays. It is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader scipy stack. It was introduced by John Hunter in the year 2002. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. It consists of several plots like line, bar, scatter, histogram etc.

OPENCV

opencv(Open source computer vision) is a library of programming functions mainly aimed at real-time computer vision. It helps to process an image and apply various functions like resizing image, pixel manipulations, object detection, etc.

TESSERACT

tesseract is an open source optical character recognition (OCR) platform for python that also serves as a wrapper for the Tesseract-OCR Engine. It can read and recognize text in images and is commonly used in python ocr image to text use cases.

ALGORITHM

I used **Optical Character Recognition (OCR)** to create the text detection model. It is the process that converts an image of text into a machine-readable text format. A scanner reads documents and converts them to binary data. The OCR software analyses the scanned image and classifies the light areas as background and the dark areas as text.

The OCR software first cleans the image and removes errors to prepare it for reading. These are some of its cleaning techniques:

- Tilting the scanned document slightly to fix alignment issues during the scan.
- Despeckling or removing any digital image spots or smoothing the edges of text images.
- Cleaning up boxes and lines in the image.
- Script recognition for multi-language OCR technology.

OpenCV's text detector implementation of EAST is quite robust, capable of localizing text even when it's blurred, reflective, or partially obscured. We call the algorithm "EAST" because it's an: Efficient and Accurate Scene Text detection pipeline.

EAST is a very robust method for text detection. It is worth mentioning as it is only a text detection method. It can find horizontal and rotated bounding boxes. It can be used in combination with any text recognition method. The text detection pipeline has excluded redundant and intermediate steps and only has two stages. One utilizes the fully convolutional network to directly produce word or text-line level prediction. The produced predictions which could be rotated rectangles or quadrangles are further processed through the non-maximum-suppression step to yield the final output.

OCR process consists of several steps, starting with importing images and ending with exporting the detected results.
Here, I am working with essential packages. OpenCV package uses the EAST model for text detection. The tesseract package is for recognizing text in the bounding box detected for the text.

1.Image Processing

After loading the packages, we have to give the location of the image to be read. Then save the original image and shape. Set new height and width to default 320 by using argument dictionary. Calculate the ratio between original and new image for both height and weight. This ratio will be used to translate bounding box location on the original image. Resize the original image to new dimensions. Calculate the ratio between original image and new image for both height and width. We need to create a 4-D input blob for feeding the image to the network. This is done using the blobFromImage function. The arguments we pass through the function are:

1. The first argument is the image itself
2. The second argument specifies the scaling of each pixel value. In this case, it is not required. Thus we keep it as 1.
3. The default input to the network is 320×320. So, we need to specify this while creating the blob. You can experiment with any other input dimension also.

4. We also specify the mean that should be subtracted from each image since this was used while training the model. The mean used is (123.68, 116.78, 103.94).
5. The next argument is whether we want to swap the R and B channels. This is required since OpenCV uses BGR format and Tensorflow uses RGB format.
6. The last argument is whether we want to crop the image and take the center crop. We specify False in this case

2.Loading pre-trained EAST model and defining output layers

To detect text, we have to load the pre-trained EAST model for text detection. A function load the network into memory. It automatically detects configuration and framework based on file name specified. In our case, it is a pb file and thus, it will assume that a Tensorflow Network is to be loaded. We would like to get two outputs from the EAST model.

1. The first layer is our output sigmoid activation which gives us the probability of a region containing text or not.
2. The second layer is the output feature map that represents the “geometry” of the image

We’ll be able to use this geometry to derive the bounding box coordinates of the text in the input image.

3.Forward pass the image through EAST model

To get the desired output layers, we have to forward pass the blob from the image. We get the output by passing the input image through the network. The output consists of two parts: scores and geometry. Forward pass the image through blob to get the desired output layers. Then we have to set new value for the layer output blob.

4.Function to decode bounding box from EAST model prediction

First we returns the bounding box and probability score if it is more than minimum confidence. Then grab the number of rows and columns from the scores volume. Then initialize our set of bounding box rectangles and corresponding confidence scores.
We start off by grabbing the dimensions of the scores and then initializing two lists:

1. Boxes: Stores the bounding box (x, y)-coordinates for text regions.
2. Confidence values: Stores the probability associated with each of the bounding boxes in boxes.

Loop over the number of rows and extract the scores (probabilities), followed by the geometrical data used to derive potential bounding box coordinates that surround text. Extract our scores and geometry data for the current row.
Loop over the number of columns. For every row, we begin looping over the columns. We need to filter out weak text detections by ignoring areas that do not have sufficiently high probability. The EAST text detector naturally reduces volume size as the image passes through the network. Our volume size is actually 4x smaller than our input image so we multiply by four to bring the coordinates back into respect of our original image and compute the offset factor.

Then it extract the rotation angle from the prediction and compute sine and cosine. Then use the geometry volume to get the height and width of the bounding box. Compute both the starting and ending (x, y)-coordinates for the text prediction bounding box. Add the bounding box coordinates and probability score to our respective lists. It returns a bounding box and probability score if it is more than minimum confidence. Then it loop over rows and number of columns. Then it extracts the rotation angle for the prediction and compute the sine and cosine. It use the geometric volume to get the dimensions of the bounding box. Then it compute start and end for the text prediction box and return bounding boxes and associated confidence value.

5.Getting final bounding boxes after non max suppression

Non Maximum Suppression is a class of algorithms to select one entity (e.g., bounding boxes) out of many overlapping entities. We can choose the selection criteria to arrive at the desired results. The criteria are most commonly some form of probability number and some form of overlap measure.

6.Generating list with bounding box coordinates and recognized text in the boxes

Initialize the list of results. Loop over the bounding boxes to find the coordinate of bounding boxes. Scale the coordinates based on the respective ratios in order to reflect bounding box on the original image. Extract the region of interest. Hence configuration setting to convert image to string. This will recognize the text from the image of bounding box. It append box coordinate and associated text to the list of results.

7.Display image with bounding box and recognized text

Moving over the results and display on the image. At first it displays the text detected by Tesseract. Then it displays the text by removing anything non-ASCII and handles characters one by one and would still use one space per character replaced. Strip method removes any leading (spaces at the beginning) and trailing (spaces at the end) characters (space is the default leading character. Then it will display the text and shows the original image.

DATA COLLECTION

Data is mainly collected from internet. Relevant data can be obtained from websites like kaggle. Data is also collected from Google trends and Goggle forms.

RESULT

We successfully created machine learning model capable of detecting text from images. The output we get after detecting and recognizing text from images will be as follows:

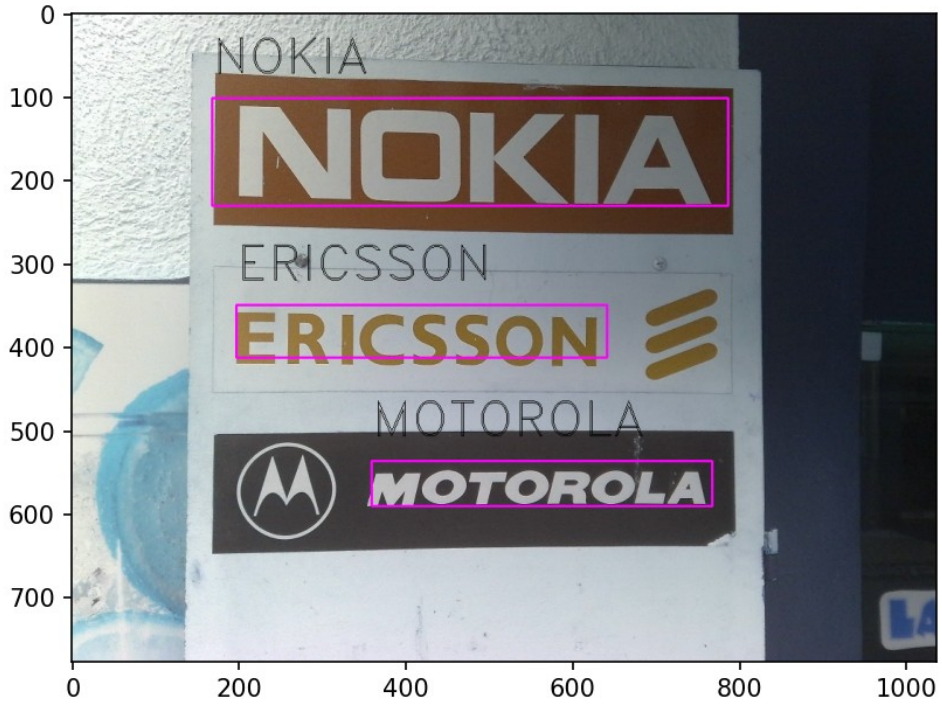
EXAMPLE-1

Detected Text is:

ERICSSON

MOTOROLA

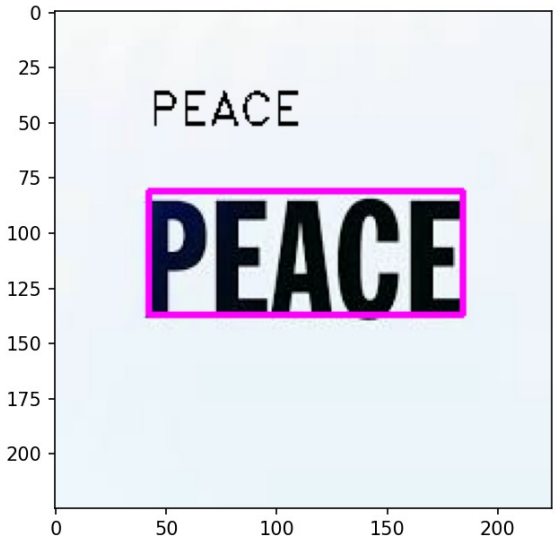
NOKIA



EXAMPLE-2

Detected text is:

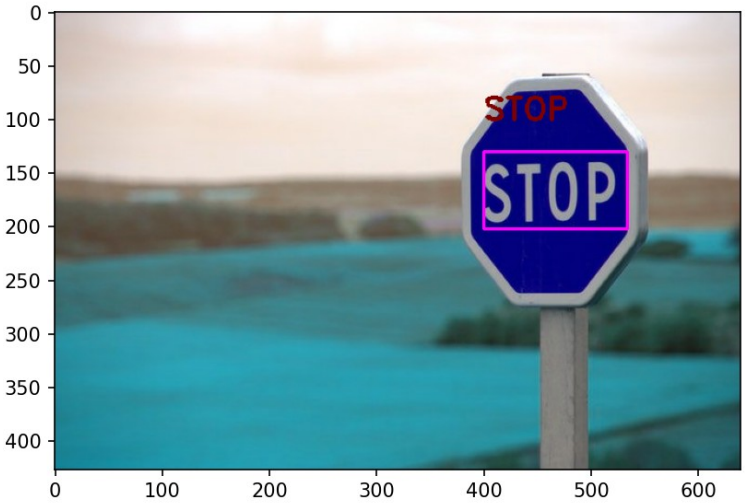
PEACE



EXAMPLE-3

Detected text is:

STOP



CONCLUSION

It was a wonderful learning experience for me while working on this project. The aim of this project was to create a machine learning model capable of detecting texts from images. This project help us to detect and recognize texts from any images. As clearly shown by the result of this model, it was able to detect text from images with different backgrounds, fonts, size and text orientations. We also see some undetected text also, but overall performance of this model is very good. The model which was created was able to detect and recognize texts from an inputted image with an accuracy of 85%. We can not expect the OCR model to be 100 % accurate. Still, we have achieved good results with the EAST model and Tesseract.