

CHAPTER 3

RESEARCH METHODOLOGY

There are several steps taken in working on this project.

1. Identification Problems and Literature Study

The first step of the research is to use OCR (optical character recognition) to save documents from picture of the building or some place within name into a file format that allows for easier document storage and for the process of reprocessing documents. research was obtained from the journals regarding problems of text recognition and image classification, which were sourced from google scholar articles from 2006 to 2019. The journal discusses the implementation of google OCR (optical character recognition) on several devices such as cellphones or computers through image processing on OCR (optical character recognition).

2. Structure algorithm

“EAST” is an Efficient and Accurate Scene Text detection. It can be detect the bounding boxes where having the text with CV2, the next step is to recognize text. The algorithm uses OpenCV EAST model for text detection and tesseract for text recognition. The capability of the Tesseract was mostly limited to structured text data. It would perform quite poorly in unstructured text with significant noise. So the images to show was worked by text detection with the EAST method and text recognition with Tesseract.

3. Pre-processing image

First we resize the image to 50 image size and make it same, next step is recognize the category of test and train, after that giving them labeling image for examples (X, and Y). X_train and Y_test containing (3000X, 2000Y) images, then labeled to get the results in the form of a Pickle (Pickle is used for serializing and

de-serializing Python object structures, also called marshalling or flattening). When we pre processing image dataset, there are 2 parts. The first of this part is used to train and the second part is being used for the test part. The image train section has 3000 images and in the test section has 2000 images. then make the image into a grayscale color and convert the image into an array of values containing 3 columns and 3 rows, and to make it simple we do normalize the data with / 225, then the image will be the original size and saved as binary format in the pickle library to be used. Labeling data will be stored in "X_train and Y_test".

4. Implementation algorithm

In the pre processing image, a feature extraction image which aims to find where the most important parts of each image will be trained by the algorithm. The image extractor is in the form of batch normalization which aims to make the image train have the same height and width, a convolutional layer which has a conv2d (convolutional layer), max-pooling here aims to find the most important part of each pixel in the train image, using relu activation (linear rectifier). In this convolutional layer, it will use 4 layers, then the train image on the test that has passed the extraction feature will be processed to carry out the labeling process by comparing the image on the train and the image being tested with (X_train x_test, Y_train y_test). The results of this labeling will be stored and loaded back into the algorithm to test the algorithm model. X_train and Y_test go to algorithm, the first step is Stem Extractor featrure uses 3 convo with different layers, the first convo layer uses 128, the second convo layer uses 256 layers, and the third uses the Convo layer 512. After that perform the Merging Branch feature with 4 different convo layers with each layer. using maxpooling, the first layer uses convo layer 256, then maxpooling with a kernel size 2X2, then the process returns to the convo with 128 layers then also maxpooling again with the same kernel 2X2 and after that the process is repeated again to the convo with 64 layers for maxpooling and Finally, the process of entering the convo uses 32 layers. After that, do the fit

model process with the dense layer with relu activation (Rectified Linear Unit), then enter the compile and calculate the loss with binary crossentropy and apply the Adam optimizer and run the train as much as epoch = 10, after that enter the fit model to save the model train using the Keras library.



5. Project Result

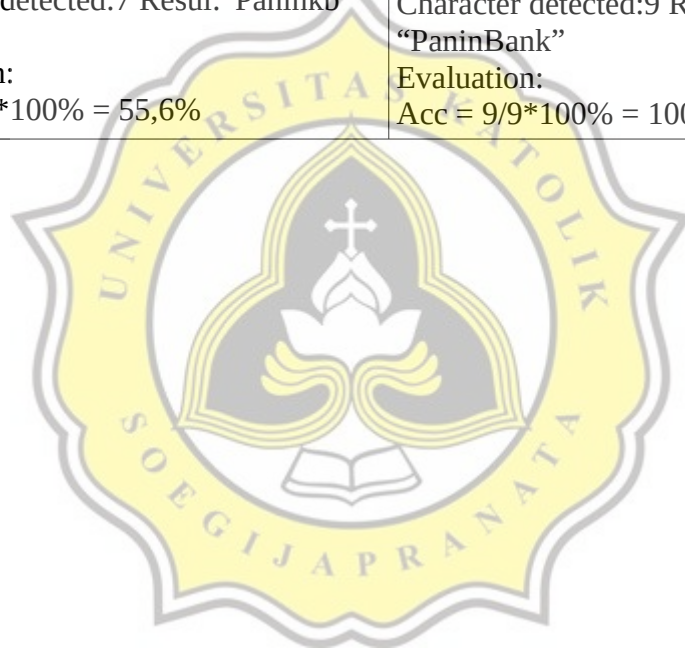
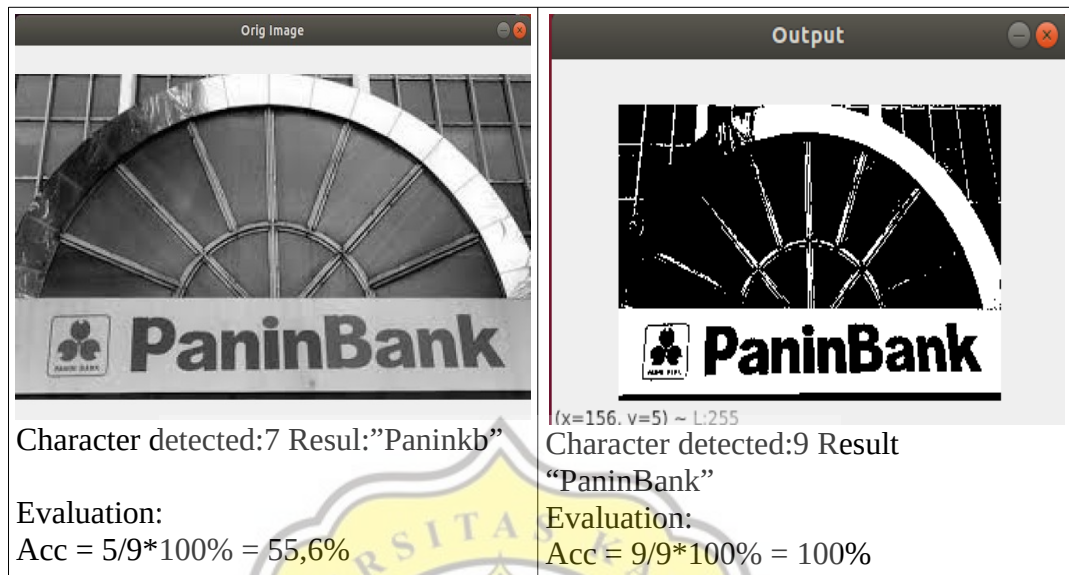
Text that has been detected will be saved and reloaded to recognize the character in the input image to recognizing the character and predicting the character. By using the library it will predict the characters and letters in the input image. And also use OpenCV as a bounding box maker or as an area box when detecting text in the image that inputed. Furthermore, using a geometry method based on height and width of the input image. There's two kind geometry, first RBOX geometry is detected using a 1x1 kernel (based on width, height, X coordinate, Y coordinate). The use of geometric RBOX is as a guide to the detected text area in the image, so that the detected text can be recognized by the algorithm. RBOX geometry has a detection character with an upright position such as writing or a series of alphabets with a standard or straight sideways position. Then Quad Geometry is detected using 1x1 kernel as well and based on (based on width, height, X coordinate, Y coordinate, X offset, Y offset, X angle, Y angle). Quad geometry is used to calculate angles using mathematical formulas, namely sin and cos. And Quad geometry is also usually used to detect the placement of various text in an image, such as writing in a curved model that is not upright and has many angles. To recognize text in images using the PYTesseract library and display the detected text in the image input.

6. Evaluation

for the evaluation of our EAST model. Comparisons will be made with the original Based EAST. original EAST will be trained with our dataset, which contains Train 3000 and Test 2000 (from IIIT 5K word). After that the two models will each detect the same 5 images with various sizes to measure the accuracy based on the character of the image testing. Measuring accuracy will be done with the Accuracy formula :

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

East_model	East_text-detection
	
Character total:9	Character total:9





Character total:8



Character detected:8

Result:"MATAHARI"

Evaluation:

Acc = $8/8 * 100\% = 100\%$



Character total:8

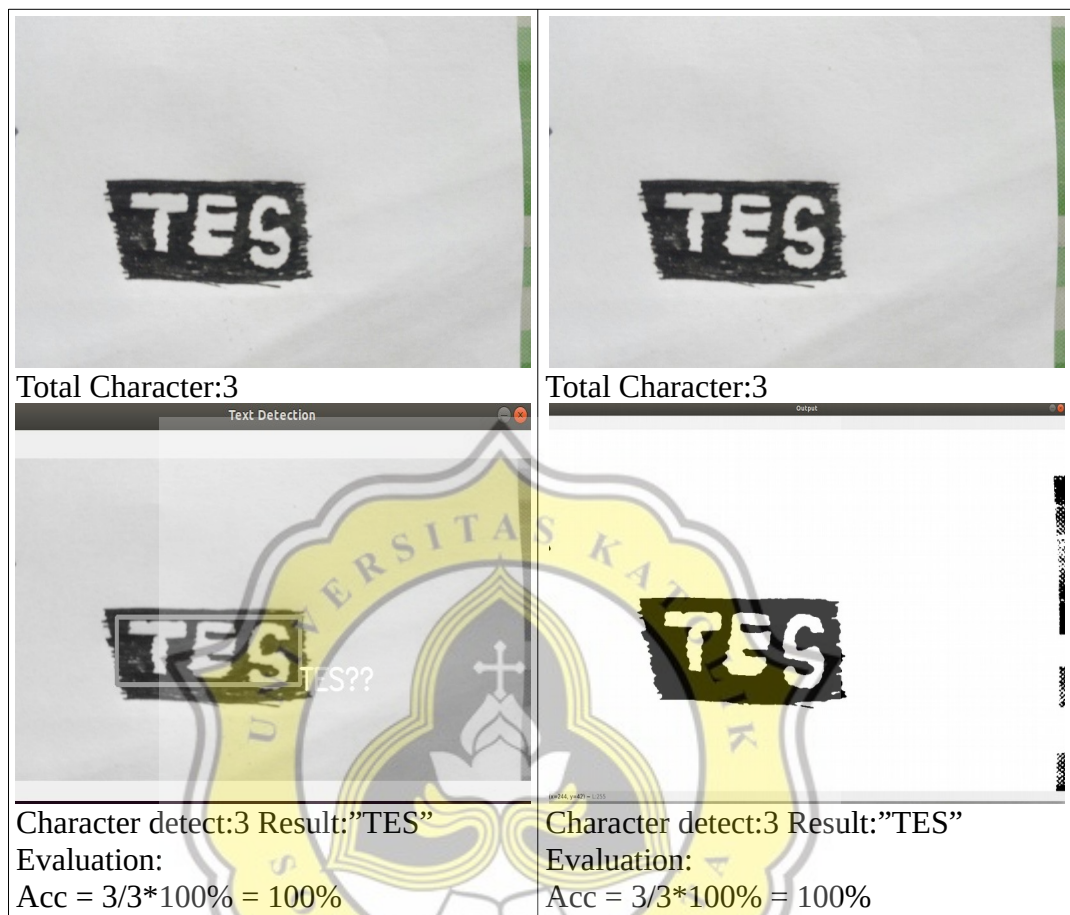


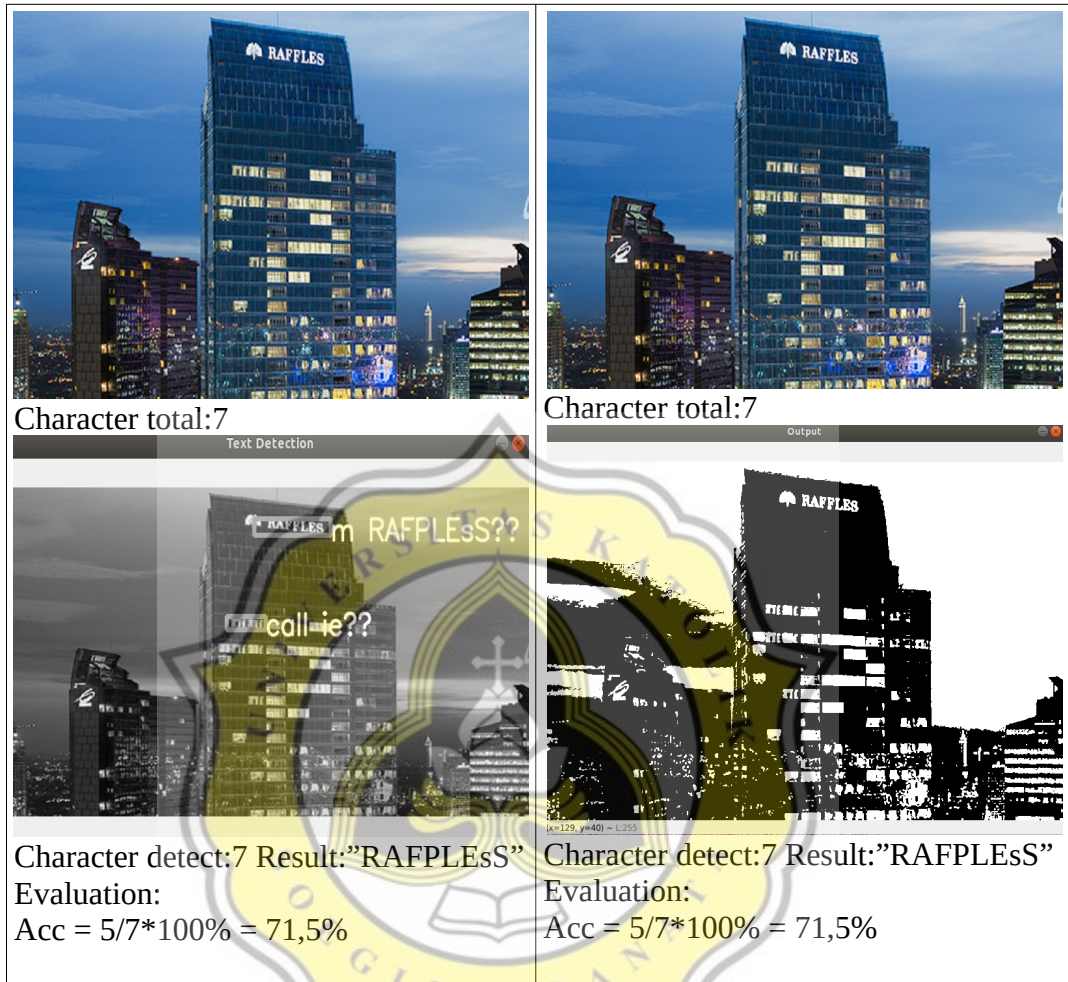
Character detected:3

Result:"eat"

Evaluation:

Acc = $3/8 * 100\% = 37,5\%$








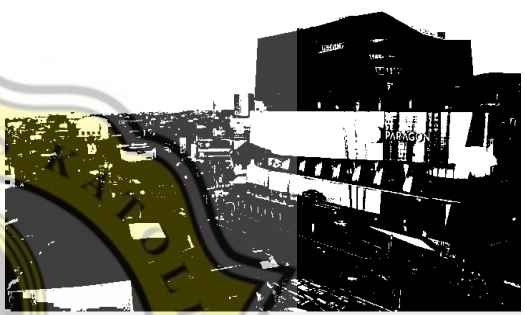
	
<p>Character total:7</p>	<p>Character total:7</p>
<p>Text Detection</p> 	<p>Output</p> 
<p>Character detect:6 Result:"PARAGGN" Evaluation: Acc = $6/7 \times 100\% = 85,8\%$</p>	<p>Character total:0 Result:"" Evaluation: Acc = $0/0 \times 100\% = 0\%$</p>

Table 3.1: Evaluation