Master Mechatronics
Computer Vision Project Report

Scene-Text-Recognition of Signboards using Tesseract

Author:
Nishanth Nandakumar
Mtr. No:

Supervisor:
Prof.Dr.Stephan Elser

July 3, 2020

Abstract

The main idea of this project is to convert images to text. In this project, we mainly consider signboards, as it is more difficult for the OCR models to detect and recognize the characters in an unstructured image. The EAST model and Tesseract are being used for text detection and text recognition respectively. The model is used to convert images part of the KAIST scene text dataset and the results obtained are evaluated and discussed.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

For many years we humans have been working on building machines which will demonstrate better intelligent behavior and thus started the work in the field of Artificial Intelligence. As per the definition of Artificial Intelligence by Elaine Rich [Ric83],

Artificial Intelligence is that the study of making computers perform tasks that are performed better by humans at the instant.

Based on this we are able to say that we humans are good at reading and understanding textual data from our surroundings. This project aims at understanding how images can be converted into textual data providing the agent's ability to read and understand texts. This process requires three steps text detection, text recognition, and natural language processing. During this project, we restrict ourselves to text detection and text recognition, and as future scope of work natural language processing is implemented. There exist many models that may easily perform text detection and text recognition i.e Optical Character Recognition (OCR) on images. During this project, the EAST model [Zho+17] is employed for text detection, and also the Tesseract engine [Smi07] is employed for text recognition. Usually, the Tesseract engine is in a position to perform both the tasks of text detection and text recognition. Since we have an interest in scene text recognition of signboards within the natural environment the images are unstructured and Tesseract performs poorly on such images. This is often due to various challenges that occur in these unstructured images like Image/Sensor noise, different viewing angles that don't seem to be parallel to the text, different lighting conditions, blur within the images due to usage of cheap digital cameras and mobile phone cameras, and finally, we may have images with sub-par resolution due to differences in cameras. So, the idea is to use the EAST model to detect the text within the image which performs well on unstructured images and to extract the ROIs which is then used by Tesseract to recognize the characters/text within the image [Ros18]. See Figure 1.1.

The models are then evaluated using the KAIST scene text dataset based on the metrics and also the results are tabulated to understand the performance of the models used. The codes used and also the method of implementation is provided within the GitHub repository [Nan20].

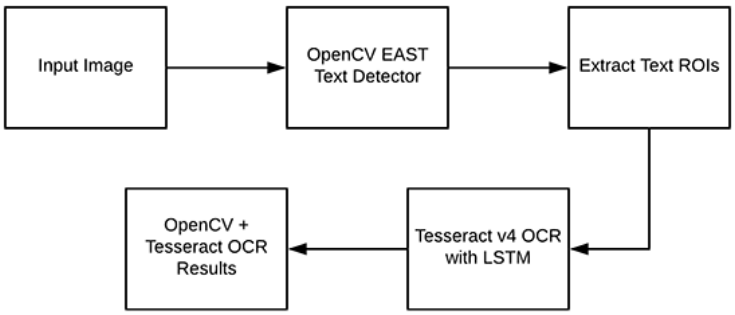


Figure 1.1: The pipeline showing EAST and Tesseract models; Source: [Ros18]

1.1 Models

In this project, we use two neural network models Efficient and Accuracy Scene Text detection(EAST) pipeline for text detection and utilize the Tesseract engine for text recognition. A brief description of these two models is mentioned in this section.

1.1.1 Tesseract

Tesseract is an Optical Character Recognition engine which was developed by Hewlett-Packard within the 1980s and is licensed under Apache license [Wik20b]. The current version of Tesseract 4.0 has been upgraded by introducing deep learning neural networks (LSTM) which have improved its performance compared to the versions which were supported by conventional methods. Tesseract may be installed on all operating systems, for this project it is installed on Ubuntu 18.0 and it can be called within a python script by importing pytesseract. For installation steps and therefore the dependencies refer [ZS20].

1.1.2 EAST

EAST is a deep neural pipeline that is introduced in the paper [Zho+17]. It consists of two stages: a Fully Convolutional neural network(FCN) and also the Non-Maximum Suppression(NMS) merging stage which is used for text detection. The Fully convolutional neural network produces word or text-line predictions which exclude the redundant and slow intermediate steps which are employed in most of the present methods for text detection. This has resulted in significantly increasing the performance providing fast and accurate text detection. The NMS is used to yield the final results and the rotated rectangles or quadrangles of the predicted text are provided as input to it. The trained EAST model is obtainable as open-source by OpenCV this is provided as a part of the git hub repository [Nan20] for straightforward access.

1.2 Dataset

The evaluation of the models used is conducted using the KAIST Scene Text Database[KL11]. This database consists of about 3000 images in Korean and English languages but about 400 images were used during the evaluation which belongs to the English language. The images are captured employing a high-resolution digital camera and a low-resolution mobile phone in numerous environments. The environments are mainly classified into indoor and outdoor which are further divided based on the lighting conditions: day, night, artificial lighting conditions.

1.3 Evaluation Metrics

The evaluation of models is conducted based on standard metrics, so it's important for us to know these metrics. In this report as we use two models EAST for text detection and Tesseract for text recognition, two metrics Intersection Over Union and Character metrics are used to evaluate the models respectively.

1.3.1 Intersection Over Union

Also referred to as the Jaccard box, IOU is extensively used for evaluating object detection which can be used for text detection also because of the similarity within the concepts. The EAST model provides us the ROIs for the text detected using which a bounding box is drawn around the text within the image. The KAIST dataset provides ground truth bounding boxes for all the images. Referring the blog page [Aid19], IOU is applied to check the overlapping of those two bounding boxes which end up within the range of 0 to 1, which is given by the subsequent formulae,

$$IOU^{truth}_{predict} = \frac{Truth \cap Predict}{Truct \cup Predict} \tag{1.1}$$

The IOU calculated is employed to get the True Positives(TP), False Positives (FP), and False Negatives(FN) based on the threshold which is 75% during this report. All the resultant IOU's greater than the brink are considered to be True Positive and also the ones below the threshold False Positive. In all the images the model fails to detect the bounding box are classified as False Negative. These are later used to calculate the precision and recall of the EAST model using the subsequent formulae,

$$Precision = \frac{TP}{TP + FP} = \frac{TrueObjectDetection}{AllDetectedBoxes} \tag{1.2}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TrueObjectDetection}{AllGroundTruthBoxes} \tag{1.3}$$

Precision is the probability of matching the model predicted bounding boxes to the ground truth bounding boxes provided by the dataset. The recall is that the sensitivity of the model which measures the probability of ground truth texts being correctly identified. The accuracy of the test is obtained using F_1 Score which

is that the harmonic mean value of precision and recall and is calculated using [Wik20a],

$$F_1 Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{1.4}$$

1.3.2 Character Metric

This metric is referred from the paper [KB18], it relies on the Levenshtein distance which calculates the minimum number of modifications required to correct the output of the OCR to match the initial text within the image. The Levenshtein distance is calculated using the modification operations which are Character Insertion(C_{ins}) is that the number of characters inserted, Character Deletion(C_{del}) is that the number of characters deleted, and Character Substitution(C_{sub}) is that the number of characters substituted to correct the OCR output to match with the text within the image. These are then considered for calculating the total character error(C_{error}) which is that the changes to be made to the OCR output text so it's similar to the text within the image,

$$C_{error} = C_{ins} + C_{del} + C_{sub} \tag{1.5}$$

The number of correct characters($C_{correct}$) generated by the OCR is obtained by,

$$C_{correct} = C_{aln} - C_{error} \tag{1.6}$$

where C_{aln} is the total number of characters present within the image.
Using this we will calculate the character precision and character recall,

$$C_{precision} = \frac{C_{correct}}{C_H} \tag{1.7}$$

$$C_{recall} = \frac{C_{correct}}{C_R} \tag{1.8}$$

where C_H and C_R are the total numbers of characters within the OCR output and also the text present within the image respectively. The accuracy of the test is additionally obtained by using the F_1 Scores which are calculated using equation 1.4.

Chapter 2

Methods

2.1 Using Tesseract and EAST model

The models EAST and Tesseract are supported by OpenCV 4, the serialized EAST model is obtained from the GitHub repository of OpenCV and the Tesseract is installed within the python environment. Following the methods of the author in [Ros18], this report utilizes the strategy for text detection and recognition using the two models. The primary step includes the preprocessing of the image because the EAST model requires the image size to be within the multiples of 32 the images are resized to 320x320. The OpenCV's deep neural network module is used for preprocessing the images which include mean subtraction, scaling, and channel swapping. The mean subtraction is used to assist us with illumination changes within the input images.

Now that the image is preprocessed, we pass the image into the EAST model and extract the output features of two layers, one the feature map of the geometry map which provides us with the coordinates to the text within the image and the other an output sigmoid activation of the scores map which provides the probability of the text contained within the detected region. The regions with high probability are filtered based on the confidence score which is kept to be 0.5 as default. Then non-maximum suppression is applied to the bounding boxes to suppress any weak overlapping bounding boxes.

Now that we've got the text regions extracted from the images it's time to perform text recognition using Tesseract. The Tesseract parameters are set to utilize the LSTM neural network to detect single lines of text within the English language. Then the bounding boxes obtained are padded for better performance and these padded ROIs are passed to the tesseract model which provides us the output which is within the text format. Refer figure A.1 for better understanding where the bounding boxes and the text output is drawn on the images.

2.2 Evaluation of the models

The performance of the model is checked by evaluating it on the KAIST scene text dataset. The output of the model for the images is evaluated using the metrics Intersection Over Union for the EAST model and Character metric for Tesseract. The dataset consists of images captured in natural scenes of signboards. It also

provides a file in the .xml format with ground truths for the text within the image. But not all the text within the image was given the ground truth so this created an issue to use an algorithm for evaluation because the output of the model included all the textual data within the image as shown in figure A.12. Another issue faced was that the model recognized the individual words within the image separately but the ground truth provided the bounding boxes for these words combined as shown in figure A.10. So the ground truth and also the predicted output of the model had to be input by hand within the algorithm to evaluate the models.

The evaluation using the IOU metric requires the ground truth bounding box and the predicted bounding box which are available. These are provided as input to determine the input coordinates of the intersection rectangle and then the area of this rectangle is calculated. The area of union is simply the addition of the two areas of bounding boxes and therefore the intersection area obtained is subtracted to avoid double count of area. These are then accustomed to calculate the IOU using equation 1.1. The IOU calculated is used to calculate the precision and also the recall using equations 1.2 and 1.3 respectively. Finally the F_1 Score was calculated based on the precision and recall obtained using equation 1.4.

The character metric is applied to evaluate Tesseract. The output of the model was obtained as a text, but there have been many issues that were faced to write an algorithm to compare the result because of the mismatch of the ground truth and also the output text. One of the main issues was that the output contained more information compared to the ground truth, as within the dataset only some important texts are referenced and the other text is ignored. The ground truth provides the coordinates for every letter within the selected words but the model discussed was used to provide the coordinates of the entire word or text within the image. Due to these reasons, the evaluation was carried out manually. The output of the model and the ground truth were saved in an excel sheet and a comparison between the words was made to get the precision and recall of the model using the equations 1.6 and 1.7 respectively. The F_1 Score for the test conducted was calculated using the equation 1.4.

2.2.1 Performance on Digital Camera Images of Signboards

The KAIST scene dataset consists of images captured by Digital Camera and this section of the report provides a detailed report on the results obtained. The images captured using digital cameras are classified based on lighting conditions therefore the evaluation was carried separately for these classes. Table 2.1 provides us the precision and recall of all the classes together with the overall result for the images captured by digital cameras. Table 2.2 provides us precision and recall obtained using the character metric for images captured under different lighting conditions.

As we observe from Table 2.1 the EAST model's overall text detection precision and recall are 61% and 88% respectively. The precision of 60.5% is good for a text detector but as can be seen from the table the performance is nearly 70% or above for the images part of Shadow, Outdoor 1, Outdoor 2, and Indoor classes. This can be mainly due to a large number of images present within these classes compared to Light and Night which has resulted in poor performance of the model. The F_1 Score calculated of the test provides an accuracy rate of 71%.

Table 2.2 provides us the results for the evaluation of Tesseract based on charac-

Lighting Condition	Text Detection		
	Precision	Recall	F_1Score
Shadow	0.75	0.92	0.83
Light	0.41	0.85	0.55
Outdoor 1	0.72	0.93	0.81
Outdoor 2	0.78	0.96	0.86
Outdoor 3	0.59	0.83	0.70
Indoor	0.68	0.89	0.77
Night	0.31	0.79	0.45
Overall	**0.61**	**0.88**	**0.71**

Table 2.1: Performance of EAST model on Digital Camera Images of Dataset

Lighting Condition	Text Recognition		
	Precision	Recall	F_1 Score
Shadow	0.61	0.62	0.62
Light	0.60	0.59	0.59
Outdoor 1	0.71	0.71	0.71
Outdoor 2	0.76	0.77	0.77
Outdoor 3	0.59	0.60	0.60
Indoor	0.78	0.78	0.78
Night	0.63	0.61	0.62
Overall	**0.67**	**0.67**	**0.67**

Table 2.2: Performance of Tesseract on Digital Camera Images of Dataset

ter metric. We can observe that the overall precision and recall rate for text recognition is 67% and 67% respectively. As is observed the performance is well above 60% for all the images captured in numerous lighting conditions and the performance is the highest for images captured Indoor with a precision rate of 78% and recall rate of 78%. The F_1 scores obtained provides an accuracy rate of 67% for the text recognition of the model on the images captured using a digital camera.

2.2.2 Performance on Mobile Phone Camera Images of Dataset

The images captured by mobile phones from the KAIST scene text dataset are evaluated and this section provides us the results. The images are captured in numerous lighting conditions and are classified into Outdoor, indoor, and Light classes.

Lighting Condition	Text Detection		
	Precision	Recall	F_1 Score
Outdoor	0.25	0.69	0.38
Indoor	0.59	0.79	0.68
Light	0.56	0.89	0.69
Overall	**0.47**	**0.79**	**0.58**

Table 2.3: Performance of EAST model on Mobile Phone Camera Images of Dataset

Lighting Condition	Text Recognition		
	Precision	Recall	F_1 Score
Outdoor	0.55	0.53	0.54
Indoor	0.80	0.80	0.80
Light	0.86	0.86	0.86
Overall	**0.74**	**0.73**	**0.73**

Table 2.4: Performance of Tesseract on Mobile Phone Camera Images of Dataset

Table 2.3 provides us the result for evaluating text detection using IOU and therefore the precision and recall are calculated for every class. The overall precision rate is 53% and also the recall is 79%. The results for the lighting conditions also are within the 50% range and this is often because of the small size of those datasets. The performance could be better evaluated if the size of the dataset were large. The F_1 Score provides the accuracy which is 58% for the evaluation of text detection on the images captured using a mobile phone.

The OCR evaluation of the model is provided within the table 2.4 based on the character metrics. The precision and recall are over 80% for the Indoor and Light which is the highest result obtained so far. The overall precision rate is 74% and the recall rate is 73%. The F_1 score calculated provides an accuracy of 73% for the text recognition test conducted on the images captured using the mobile phones.

Chapter 3

Conclusion

The main objective of this project was natural scene text detection and this has been achieved using the EAST and Tesseract models. The two models provide good results after they are evaluated on the KAIST scene text database, but the models could have been better evaluated if we had a dataset with more images. This was a constraint as there are only a few datasets available for unstructured images. We should also keep in mind that no machine in the world is going to be able to perform at a 100% accuracy rate. All the set tasks during the proposal have been achieved, in this report, the preprocessing has been carried out using the OpenCV deep neural network instead of applying other methods as the neural networks provided the required output for the models to perform well on the images. Also, the comparison table between the performance of Tesseract and the models used in this report is not provided due to the tedious task of evaluation which was conducted on 400 images. As a future study, research would be conducted to enhance the performance of the two models, which may be done by training the models on the unstructured dataset which might give us better results. Also, we are able to reduce the padding provided which is able to improve the results as shown in figure A.14. A plethora of methods is available going forward to enhance the results of the two models at hand and to realize the task of converting images to text. A separate work was carried out to understand natural language processing and good results were obtained using the Stanza toolkit by Stanford University which can be combined with the EAST and Tesseract to develop an agent that has a capability to read from the natural environment.

Appendix A

Supporting Information

(a) Text Recognition (b) Text Detection with IOU

Figure A.1: This is an image captured in natural light using a Digital camera. The left image provides us the output of the model inserted in the image. The right image shows the output of IOU calculation where the red box is predicted bounding box by EAST model and the green box is the ground truth bounding box.

(a) Text Recognition (b) Text Detection with IOU

Figure A.2: The left image provides us the output of the model inserted in the image. The right image shows the output of IOU calculation.

(a) Text Recognition (b) Text Detection with IOU

Figure A.3: The left image provides the output of the model. We can observe that the model is clearly able to recognize the letters within the image. But it clearly fails to recognize the number 7 in the background.

(a) Text Recognition (b) Text Detection with IOU

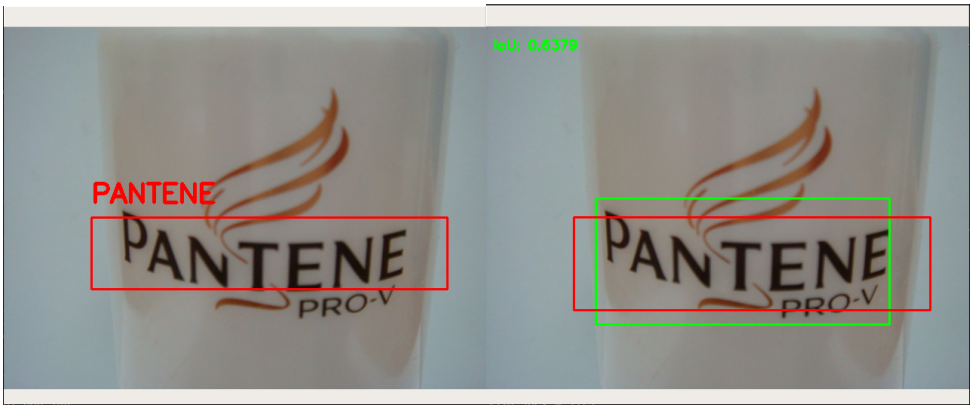
Figure A.4: Here we can observe the model performs well even with shadows and variations in the image. It is able to detect all the characters in the image as shown on the left. The IOU is 0.86 which is a better result.

(a) Text Recognition (b) Text Detection with IOU

Figure A.5: This image is captured at night using a digital camera. The image on the left provides the output of the model which recognizes all the characters. The image on the right provides the IOU which is 0.69.

(a) Text Recognition (b) Text Detection with IOU

Figure A.6: This image is captured outdoor using a digital camera. The model performance really well even with a lot of noise in the image. It is able to detect all the characters as on the left with an IOU of 0.77 as shown on the right.



(a) Text Recognition (b) Text Detection with IOU

Figure A.7: This is an image of a shampoo bottle. We can observe on the left the model performs well even if the text is present on a curved surface. But the model fails to recognize the PRO-V text present in the image. The left image shows the two bounding boxes with a relatively poor IOU of 0.63.

(a) Text Recognition (b) Text Detection with IOU

Figure A.8: We can observe the model performs well on the text with different font styles as in the left image. All the characters in the image are recognized by the model with an IOU of 0.817 as on the right.

(a) Text Recognition (b) Text Detection with IOU

Figure A.9: This image is captured indoor with artificial lighting. The model performs well but with a slight error as can be seen on the left. It inserts two extra characters to the extreme end of the word. The text detection performance is good with an IOU of 0.71 as shown on the right.

(a) Text Recognition (b) Text Detection with IOU

Figure A.10: This is an image captured using a mobile phone camera. The performance of the model is really good as can be seen on the right even with a lot of noise in the image. The IOU is relatively bad which is 0.637 as shown on the right.

(a) Text Recognition (b) Text Recognition (c) Text Detection with IOU

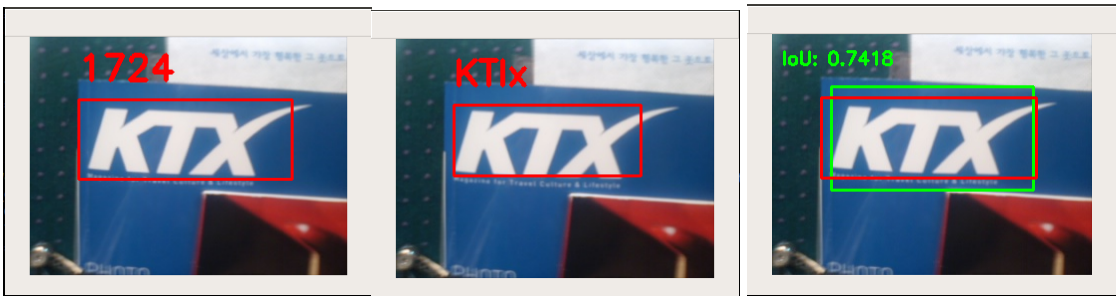
Figure A.11: The model detects two words in the image individually as shown in the left and the center images. But this is considered to be single text in the ground truth so the two predicted bounding boxes were combined to obtain the IOU as shown on the right.

(a) Text Recognition (b) Text Recognition (c) Text Detection with IOU

Figure A.12: This image also has the same issue as in the image A.10.The left and center images provide the text recognition of two words and the right image text detection and IOU.

(a) Text Recognition (b) Text Recognition (c) Text Detection with IOU

Figure A.13: Here we can observe the model fails to recognize the characters on the left image. But it recognizes the text in the background which is an issue present while evaluating the dataset.



(a) Text Recognition (b) Text Recognition (c) Text Detection with IOU

Figure A.14: The image on the left shows that the model performs badly with recognizing the text. But the performance is improved by reducing the padding from 0.05 to 0.001 as shown in the middle.

Bibliograpy

[Aid19] Manal El Aidouni. *Evaluating Object Detection Models: Guide to Performance Metrics*. 2019. URL: https://manalelaidouni.github.io/manalelaidouni.github.io/Evaluating-Object-Detection-Models-Guide-to-Performance-Metrics.html.

[KB18] Romain Karpinski and Abdel Belaid. "ZoneMapAlt: An alternative to the ZoneMap metric for zone segmentation and classification". In: *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*. IEEE. 2018, pp. 357–362.

[KL11] Prof. Jin Hyung Kim and Seonghun Lee. *KAIST Scene Text Database*. 2011. URL: http://www.iapr-tc11.org/mediawiki/index.php?title=KAIST_Scene_Text_Database.

[Nan20] Nishanth Nandakumar. *Scene-Text-Recognition-of-Signboards-using-Tesseract*. 2020. URL: https://github.com/nishanthnandakumar/Scene-Text-Recognition-of-Signboards-using-Tesseract.

[Ric83] Elaine Rich. *Artificial Intelligence*. McGraw-Hill, 1983.

[Ros18] Adrian Rosebrock. *OpenCV OCR and text recognition with Tesseract*. 2018. URL: https://www.pyimagesearch.com/2018/09/17/opencv-ocr-and-text-recognition-with-tesseract/.

[Smi07] Ray Smith. "An overview of the Tesseract OCR engine". In: *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*. Vol. 2. IEEE. 2007, pp. 629–633.

[Wik20a] Wikipedia. *F1 score — Wikipedia, The Free Encyclopedia*. http://en.wikipedia.org/w/index.php?title=F1%20score&oldid=963876982. [Online; accessed 03-July-2020]. 2020.

[Wik20b] Wikipedia. *Tesseract (software) — Wikipedia, The Free Encyclopedia*. 2020. URL: https://en.wikipedia.org/wiki/Tesseract_(software).

[Zho+17] Xinyu Zhou et al. "EAST: an efficient and accurate scene text detector". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2017, pp. 5551–5560.

[ZS20] Filip Zelic and Anuj Sable. *A comprehensive guide to OCR with Tesseract, OpenCV and Python*. 2020. URL: https://nanonets.com/blog/ocr-with-tesseract/.