19AIE205 Python for Machine Learning

Project Report – 2020

Submission Date: December 5, 2020

(Scene Text Detection using Opencv)

Submitted By:

NITHIN SYLESH

(AM.EN.U4AIE19044)

1. Project title

   Easy and effeciant Scene Text Detection using opencv and pytesseract

2. Problem definition/Abstract

   The detection and recognition of scene text from camera captured images Text detection is the process of detecting the text present in the image, followed by surrounding it with a rectangular bounding box. an image is segmented into multiple segments. Each segment is a connected component of pixels with similar characteristics. The statistical features of connected components are utilised to group them and form the text.

   In this work , I propose a simple yet powerful pipeline that yields fast and accurate text detection in natural scenes. The pipeline directly predicts words or text lines of arbitrary orientations and quadrilateral shapes in full images, eliminating unnecessary intermediate steps with a single neural network. The simplicity of this pipeline allows concentrating efforts on designing loss functions and neural network architecture

3. Datasets

   ICDAR 2015 is used in Challenge 4 of ICDAR 2015 Robust Reading Competition .It includes a total of 1500 pictures. These images are taken by Google Glass in an incidental way. Therefore text in the scene can be in arbitrary orientations, or suffer from motion blur and low resolution

   The **KAIST scene text dataset** comprises 3000 images captured in different environments, including outdoors and indoors scenes under different lighting conditions (clear day, night, strong artificial lights, etc). Images were captured either by the use of a high-resolution digital camera or a low-resolution mobile phone camera. All images have been resized to 640x480.

4. Data preprocessing

The EAST detector requires that your input image dimensions be multiples of 32, so if you choose to adjust your width and height values, make sure they are multiples of 32
Images accepted by the detector are of 720 p at maximum
Detection of images having resolution above 720p would lessen the accuracy

- Label Generation

Without loss of generality, we only consider the case where the geometry is a quadrangle. The positive area of the quadrangle on the score map is designed to be roughly a shrunk version of the original one, illustrated in Fig. 4 (a). For a quadrangle Q = {pi |i ∈ {1, 2, 3, 4}}, where pi = {xi , yi} are vertices on the quadrangle in clockwise order. To shrink Q, we first compute a reference length ri for each vertex pi as

ri = min(D(pi , p(i mod 4)+1),

D(pi , p((i+2) mod 4)+1))

where D(pi , pj) is the L2 distance between pi and pj . We first shrink the two longer edges of a quadrangle, and then the two shorter ones. For each pair of two opposing edges, we determine the “longer” pair by comparing the mean of their lengths. For each edge hpi , p(i mod 4)+1i, we shrink it by moving its two endpoints inward along the edge by 0.3ri and 0.3r(i mod 4)+1 respectively.

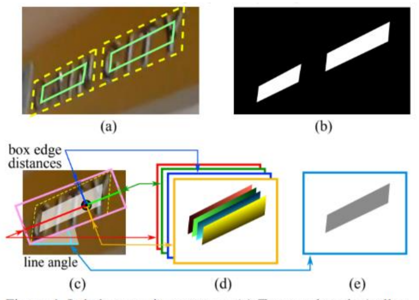


Figure 4. Label generation process: (a) Text quadrangle (yellow dashed) and the shrunk quadrangle (green solid); (b) Text score map; (c) RBOX geometry map generation; (d) 4 channels of distances of each pixel to rectangle boundaries; (e) Rotation angle.

For those datasets whose text regions are annotated in QUAD style (e.g., ICDAR 2015), we first generate a rotated rectangle that covers the region with minimal area. Then for each pixel which has positive score, we calculate its distances to the 4 boundaries of the text box, and put them to the 4 channels of RBOX ground truth. For the QUAD ground truth, the value of each pixel with positive score in the 8-channel geometry map is its coordinate shift from the 4 vertices of the quadrangle

- Loss Functions

The loss can be formulated as
$L = Ls + \lambda gLg$
where Ls and Lg represents the losses for the score map and the geometry, respectively, and λg weighs the importance between two losses. In our experiment, we set λg to 1

To facilitate a simpler training procedure, we use classbalanced cross-entropy introduced in, given by

$$Ls = \text{balanced-xent}(\hat{Y}, Y*) = -\beta Y* \log \hat{Y} - (1 - \beta)(1 - Y*) \log(1 - \hat{Y})$$

where $\hat{Y} = Fs$ is the prediction of the score map, and $Y*$ is the ground truth.
The parameter β is the balancing factor between positive and negative samples, given by

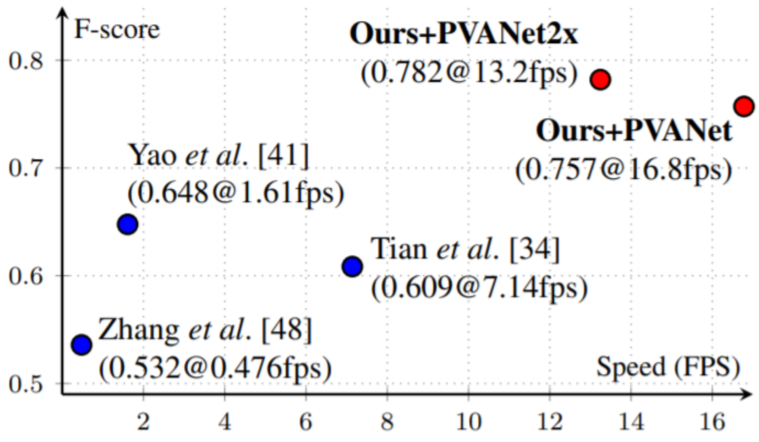
$$\beta = 1 - P\ y{*}{\in}Y{*}\ y * |Y{*}| .$$

This balanced cross-entropy loss is first adopted in text detection by Yao et al. as the objective function for score map prediction. We find it works well in practice.

- Training

The network is trained end-to-end using ADAM [18] optimizer. To speed up learning, we uniformly sample 512x512 crops from images to form a minibatch of size 24. Learning rate of ADAM starts from 1e-3, decays to one-tenth every 27300 minibatches, and stops at 1e-5. The network is trained until performance stops improving.

- Data visualization



Performance versus speed on ICDAR 2015 text localization. As can be seen, our algorithm significantly surpasses competitors in accuracy, whilst running very fast.
features are manually designed [5, 25, 40, 10, 26, 45] to capture the properties of scene text, while in deep learning based methods [3, 13, 11, 12, 7, 48] effective features are directly learned from training data. However, existing methods, either conventional or deep neural network based, mostly consist of several stages and components, which are probably sub-optimal and timeconsuming. Therefore, the accuracy and efficiency of such methods are still far from satisfactory.

5. Python packages

Imutils   - **Imutils** are a series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV and python 3

Numpy   - **NumPy** is a **Python** library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices.

Time   - **Python** has **defined** a module, “**time**” which allows us to handle various operations regarding **time**, its conversions and representations, Opencv - **OpenCV** (**Open** Source Computer Vision Library) is an **open** source computer vision and machine learning software library. **OpenCV** was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.

Pytesseract - **Python-tesseract** is an optical character recognition (OCR) tool for python. That is, it will recognize and “read” the text embedded in images. ... Additionally, if used as a script, **Python-tesseract** will print the recognized text instead of writing it to a file.

6. Algorithm

The algorithm used in my project is EAST(Easy And efficient Scene Text Detection) algorithm

The EAST pipeline is capable of predicting words and lines of text at arbitrary orientations on 720p images, and furthermore, can run at 13 FPS

Perhaps most importantly, since the deep learning model is end-to-end, it is possible to sidestep computationally expensive sub-algorithms that other text detectors typically apply, including candidate aggregation and word partitioning.

To build and train such a deep learning model, the EAST method utilizes novel, carefully designed loss functions.

A pre trained east detector model has been used in this project . The model is a protobuf file with extension .pb that contains the graph definition as well as the weights of the model. Thus the protobuf model is loaded into the memory

The algorithm follows the general design of DenseBox in which an image is fed into the FCN and multiple channels of pixel-level text score map and geometry are generated. One of the predicted channels is a score map whose pixel values are in the range of [0, 1]. The remaining channels represent geometries that encloses the word from the view of each pixel. The score stands for the confidence of the geometry shape predicted at the same location.

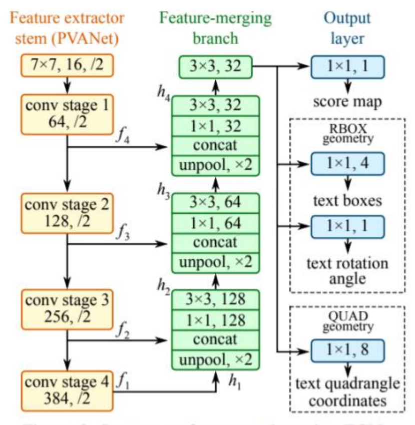
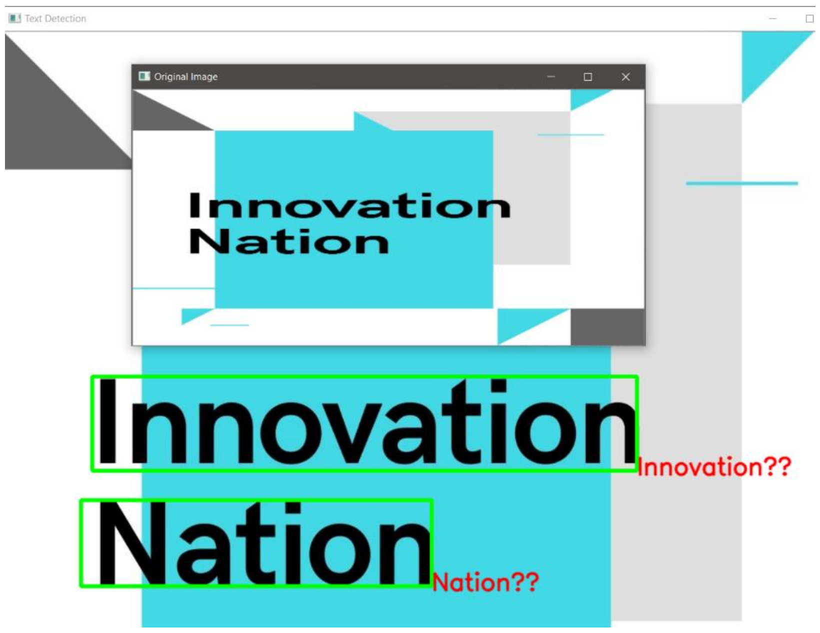


Figure 3. Structure of our text detection FCN.

7. Experimental result

It is able to handle various challenging scenarios, such as non-uniform illumination, low resolution, varying orientation and perspective distortion. As can be seen, the trained model produces highly accurate geometry maps and score map, in which detections of text instances in varying orientations are easily formed.

In ICDAR 2015, when images are fed at their original scale, the proposed method achieves an Fscore of 0.7820.



(output)

In KAISTdataset, the proposed algorithm result in higher accuracy , Fscore is 0.0614 while that in recall is 0.053, which confirm the advantage of the proposed algorithm, considering that KAIST datasetis the largest and most challenging benchmark to date.

The improvements of the proposed algorithm over previous methods prove that a simple text detection pipeline, which directly targets the final goal and eliminating redundant processes, can beat elaborated pipelines, even those integrated with large neural network models.

While the proposed method significantly outperforms state-of-the-art methods, the computation cost is kept very low, attributing to the simple and efficient pipeline.

8. Conclusion.

I have presented a scene text detector that directly produces word or line level predictions from full images with a single neural network. By incorporating proper loss functions, the detector can predict either rotated rectangles or quadrangles for text regions, depending on specific applications. The experiments on standard benchmarks confirm that the proposed algorithm substantially outperforms in terms of both accuracy and efficiency.

9. Reference

[1] Text Recognition Algorithm Independent Evaluation (TRAIT). http://www.nist.gov/itl/iad/ig/trait-2016.cfm. Accessed: 2015-11-1

[2] M. Busta, L. Neumann, and J. Matas. Fastext: Efficient unconstrained scene text detector. In Proc. of ICCV, 2015.

[3] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. J. Wu, and A. Y. Ng. Text detection and character recognition in scene images with unsupervised feature learning. In Proc. of ICDAR, 2011.

[4] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. FeiFei. Imagenet: A large-scale hierarchical image database. In Proc. of CVPR, 2009.

[5] B. Epshtein, E. Ofek, and Y. Wexler. Detecting text in natural scenes with stroke width transform. In Proc. of CVPR, 2010

[6] R. Girshick. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, pages 1440–1448, 2015.

[7] A. Gupta, A. Vedaldi, and A. Zisserman. Synthetic data for text localisation in natural images. arXiv preprint arXiv:1604.06646, 2016.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385, 2015.

[9] L. Huang, Y. Yang, Y. Deng, and Y. Yu. Densebox: Unifying landmark localization with end to end object detection. arXiv preprint arXiv:1509.04874, 2015.