ISSN NO: 2249-3034

A System for Curve Text Detection using Polygon- Faster-RCNN

Dr DINESH KUMAR

[1] Research Scholar, Institute of Computer Science and Information Science,
Srinivas University, Mangalore, India,
[2] Professor, Institute of Computer Science and Information Science,
Srinivas University, Mangalore, Karnataka, India,

Abstract

At present, text orientation is not diverse enough in the existing scene text datasets. For instance, text with curve-orientation has close to zero existence in them and thus receivedminimal attention from the community. It is the first properly scaled scene dataset that features three different text orientations: horizontal, multi-oriented, and curve-oriented. In addition, several studies regarding other important elements such as the practicality and quality of ground-truth, evaluation protocol, insights of curved text, and the annotation process are presented in this work as well. These elements are found to be as important asthe images and ground-truth to facilitate a new research direction. In addition, Polygon-Faster-RCNN, a text detection baseline, has been proposed as the contribution of this work. It has demonstrated its ability in detecting text in all kinds of orientations.

1. Introduction

Human communicates, not only verbally but often visually through written text, stationary signs, fancy billboards, banners, posters, etc. The constant need of conveying information in the human society produces an abundant existence of text in every corner of theworld. Human leverages the existence of text in natural scenes to drive, find out the cafehe or she wants to go, understand the context of the piece of art on the wall, be aware of the sales happening in certain stores on the street, the application list goes on and on. The abundant of such information in our everyday scenes calls for the incorporation ofit in the design of intelligence systems. Figure 1 shows two natural images filled with scene text. By giving machine the ability to understand text in the wild, we could havea smarter navigation system, convenient shoot-and-translate mobile application, efficientimage retrieval system, etc. All of these applications depend on the progress of the scenetext understanding research. Both scene text detection and scene text recognition are two active research topics under the umbrella of scene text understanding. Given a natural scene image, the goal ofscene text detection is to determine the existence of text, and return the location if it is present. W. Huang et al. [1]; Y. Liu & Jin [2]; X.-C. Yin et al. [3,4]; Zhang et al. [5]; Zhou et al. [6] Meanwhile, the current attempts at scene text recognition problem assumes pre-cropped text image patches as input, aims to recognize the text in it (i.e. output in human language) (Jaderberg et al. [6]; B. Shi et al. [7] ; T. Wang et al. [8]). Like many research fields, the scene text

detection challenge has several common datasets for researchers to benchmark their algorithms. Over the years, tougher scene text datasets were launched to better represent text instance in real-word scenarios. For instance, MSRA-TD500 (Yao et al. [9]) were launched to introduce the problem of detecting text in arbritary orientation, the images of ICDAR2015 were captured withoutprior efforts in focusing, which is more likely the case in compared to well-captured pho-tos in terms of application, the large scale COCO-Text (Andreas et al. [10]) captured the wide scene diversity of the real-world, etc. Despite the emergence of multiple datasets, text in curve orientation, in spite of its commonness in real-world scenes, went under the radar – it has close to zero existence in the existing scene text datasets. Without the motivation of a proper dataset, scene text detection algorithm with curved text in consideration is rarely seen. This would prevent scene text detection system in achieving robust performance against text of all orientations in the real-world application. In the meanwhile, there is a concrete sign that the community is summoning the curved text data – multiple state-of-the-art performingmulti-oriented scene text detectors have reported failure in detecting curved text (B. Shi et al. [11]; Xing et al. [12]; X.-C. Yin et al. [13]; Zhang et al. [14]).

(a)

(b)

Figure 1: Image taken in (a): Time Square, New York, and (b): Jalan Tun Razak,Kuala Lumpur. Texts are an integral part of everyday scenes in the human society.

2. Related Work

Text detection solutions with multi-oriented text in consideration was rare before the introduction of MSRA-TD500 in 2012. In the past, the main challenge of multi-oriented text has always been the design of complex hueristics in grouping character candidates into a text-line candidates (or word). Most of the works (Epshtein et al. [15], Kim et al.[16]; Liao et al. [17]; Neumann & Matas [18]; Pan et al. [19], Yi & Tian [20,21]; X. Yin et al. [22]. X.-C. Yin et al. [23] were designed with the assumption oftext are horizontal oriented. X.-C. Yin et al. [24] algorithm was one of the advanced multi-oriented text detection works in prior to the CNN era. The work adopted a coarse-to-fine approach in forming text-line candidates as illustrated in Figure 2. Firstly, it

groups character candidates into clusters based on their color, stroke width, and compact-ness similarity. Then, the character pairs with similar orientations will be grouped into finer clusters. Lastly, text-line candidates with different orientations will be separated andproduce the final detection outputs. Zhang et al. [25] s solution is another work that has similar rule-based approach in spotting multi-oriented text line candidates. Upon extract-ing text regions and character candidates with FCN and MSER respectively, the algorithmdetermines the best angle offset, θ that will hit the most character blocks within the same text region. As illustrated in Figure 3(c) the connecting top two lines), the spotted text region could potentially contains several text line candidates. The algorithm then use thegeometrical information of each character blocks to separate them into different clusters,indicating the potential of multiple text-line candidates (Figure (3(d-f)) Another FCN was employed to remove false positives among these candidates for the final detection stage. The entire pipeline can be seen in Figure 3. Such rule-based approaches suffer from three major drawbacks: i) complicated heuristic designs, ii) aggregated errors frommultiple stages, and iii) restricted circumstances coverage. These shortcomingsare the reasons why they are hard to be adapted for future work. The top-down approaches are well-received by the community today. Vast majority of the proposal-based and single-network-based solutions directly predict the location oftext regions as the detection output. This line of works formulate the detection task as a regression problem, expecting their networks to learn how to regress the encoded bound-ing box *(x, y, w, h)* from extracted feature maps. Faster-RCNN and SSD does not fit the shape of multi-oriented text[9]. W. He et al. [26], Jiang et al. [27]; Liao et al. [28]; Y. Liu & Jin [29]; Ma et al. [30]; Xing et al. [31]; Zhou et al. [32] works can all broadlygrouped into this category. All of the aforementioned works have demonstrated the potential of scaling up either Faster-RCNN or SSD kind of framework to regress more parameters than just the conventional *(x, y, w, h)*. Apart from that, the direct text region spotting mechanism eliminated the needs of heuristic designs and multiple stage of processess, which are usually error prone. Attracted by these attributes, the proposed methodof this work is based on the Faster-RCNN framework. Specifically, the effort is focusedon the limitation of the existing axis-aligned bounding box, rotational bounding box, andquadrilateral bounding region in detecting curved text instances. Text detection algorithm with curved text in consideration is rarely seen. Risnumawan etal. [33]'s system is one of the unconventional cases. Like many other scene text detec-tors before the CNN era, Risnumawan et al. [34] designed a symmetry-based feature extraction technique to spot for character candidates in natural images. The novelty of this system is the design of an eclipse growing technique that is capable of connecting neighbouring character candidates with no orientation restrictions. Figure 4 illustratestwo of the grouping examples. The eclipse growing process starts at a random pixel in the feature space, and search through all adjacent pixels for spotted character candidates(obtained in previous stages). The expansion will be terminated when there is no more character candidate in the nearby pixel. The algorithm has demonstrated its effectivenesson the CUTE80 dataset with several notable rooms for improvement: i) the entire system consists of many complicated hueristics (to extract features, generate text-line candidates,and false positives removal) which are prone to errors, ii) rotated bounding box output for- mat which does not fit the curved text instances tightly, and iii) slow execution speed – 16.1s per image. In contrast, Yuliang et al. [35] s CTD (Curved Text Detector) is one of the modernscene text detectors that was designed with curved text in consideration. It was built on top the Faster-RCNN architecture, trained to regress *polygon bounding region* in order to detect and bind text of

all orientations tightly. The author empirically found that poly- gon with fourteen pairs of vertices is sufficient to cover most *line-level text instances* in natural images, hence extended the regression head of CTD to have thirty-two parame- ters (for x_{min}, y_{min}, x_{max}, y_{max}, w_1, h_1, ... w_{14}, h_{14}). In addition, the authors argued that the relationship between each polygon vertices is sequentially related, hence, employed a variant of Recurrent Neural Network (RNN) – Bidirectional Long Short Term Memory (BLSTM) to model the relationship between each of them. Meanwhile, the work followed the convention of Faster-RCNN (Ren et al. [36] and DMPNet (Y. Liu & Jin [37]) agreed that relative information (i.e., the *w*idth and *h*eight difference between each ver- tices to a reference point (x_{min}, y_{min})) are easier targets to be optimized in compared tothe absolute information of each vertices (i.e., the exact x and y values for each vertices).CTD inherited the strength of Faster-RCNN – the absence of complicated feature ex- traction design and multi-stage hueristic processes, and was proven to be both effective and efficient on the CTW1500 dataset. The proposed method is different with the said model in several ways: i) Polygon-FRCNN is modelled to detect text in word-level (in- line with the ICDAR2013, ICDAR2015, COCO-Text, MLT datasets), ii) consequently, it has lesser parameters in the regression head to be optimized (i.e., six in compared to fourteen vertices)[10] iii) no need for the intermediate BLSTM module (again, due to the smaller amount of vertices).

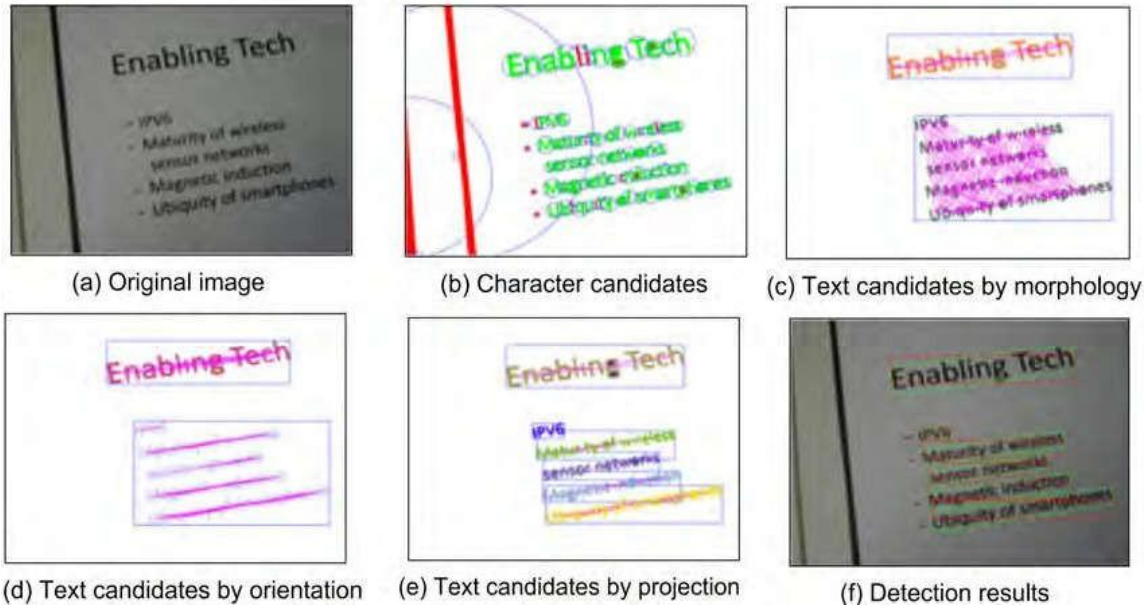


Figure 2: Overall pipeline of X.-C. Yin et al. [38] s approach. The multi-stage text-line generation process starts from (c) to (e).

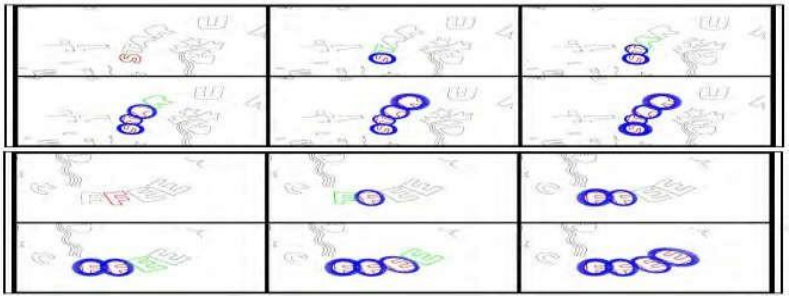


Figure 3: Examples of the eclipse growing scheme employed in Risnumawan et al. (2014)'s work.

ISSN NO: 2249-3034

3. Proposed Method

The objective of RPN is to propose regions of interest (ROIs) for the next stage of the network. The number of proposal is another hyper parameter of the Polygon-FRCNN model, as inherited from Faster-RCNN. Following the studies of Ren et al. [40] and J. Huang et al. [41], the proposed method is configured to produce 300 proposals for each input image. Both studies agreed that increasing the number of proposals does not improve the performance but would slow down the execution speed instead. The raw proposals are first filtered by the Non-Maximal Suppression (NMS) algorithm then clipped with their classification scores. The NMS algorithm is a straightforward way to eliminate overlapping proposal boxes, which is explained in the Algorithm. In the casewhere there are more than 300 proposal left after the NMS filtering process, the top-300proposals based on their classification scores will be retained for the next stage. Figure 5 visualizes the proposals produced by RPN upon feeding in an input image

Figure 5: Proposals ((a) - (c)) and detections ((d) - (f)) produced from RPN and PCN respectively at the end of different training iterations. (a) and (d): 0 iteration, (a) and (e): 100,000th iteration, (c) and (f): 200,000th iteration. The width of the bounding lines indicates the confidence level of each proposal or prediction, wider the line, higher the confidence.

3.1 Polygon-Faster-RCNN

Inspired by the success of (Jiang et al. [42]; Y. Liu & Jin [43]; Ma et al. [44]; Yuliang et al. [45]) in building scene text detectors on top of the Faster-RCNN architecture, the proposed method of this work, is too built based on Faster-RCNN. Polygon-FRCNN, is designed to detect text instances with the

polygon format. The main motive of modeling Polygon-FRCNN to produce polygon output is in line with the reason of employing polygon as the ground truth format in *Text detection* – to bind text of all orientations tightly. Figure 6 depicts the overall architecture of Polygon-FRCNN. Inherits from Faster-RCNN, Polygon-FRCNN has three major blocks: feature extraction, region proposal network, and polygon classifier network (originally box classifier network), all of which will be discussed in the following subsections.

3.2 Feature Extraction

The feature extraction part of a network is important as good features usually translate to good performance in computer vision problems. Polygon-FRCNN adopts the Inception-ResNet-v2 (Szegedy et al. [46]) model as the feature extractor of the network. The rationale behind the option are i) Inception-ResNet-v2 has demonstrated its strength in the studies of Szegedy et al. [47] and J. Huang et al. [48] ii) pre-trained weights (onthe MSCOCO (Lin et al. (2014)) dataset) were made available by J. Huang et al. [49],which eliminates the need of training the network from scratch that would take weeks andmany computing resources. The network architecture of Inception-ResNet-v2 can be seen in Figure 7. It combines the strength from both ResNet (K. He et al. [50]) and the Inception modules (Szegedy et al. [51]) which allows i) large number of layers withinthe network, and ii) faster running speed with parameter factorization. Both of these advantages made the construction and training of a very deep network possible, which is essential in extracting high quality features for the later regression and classification tasks.

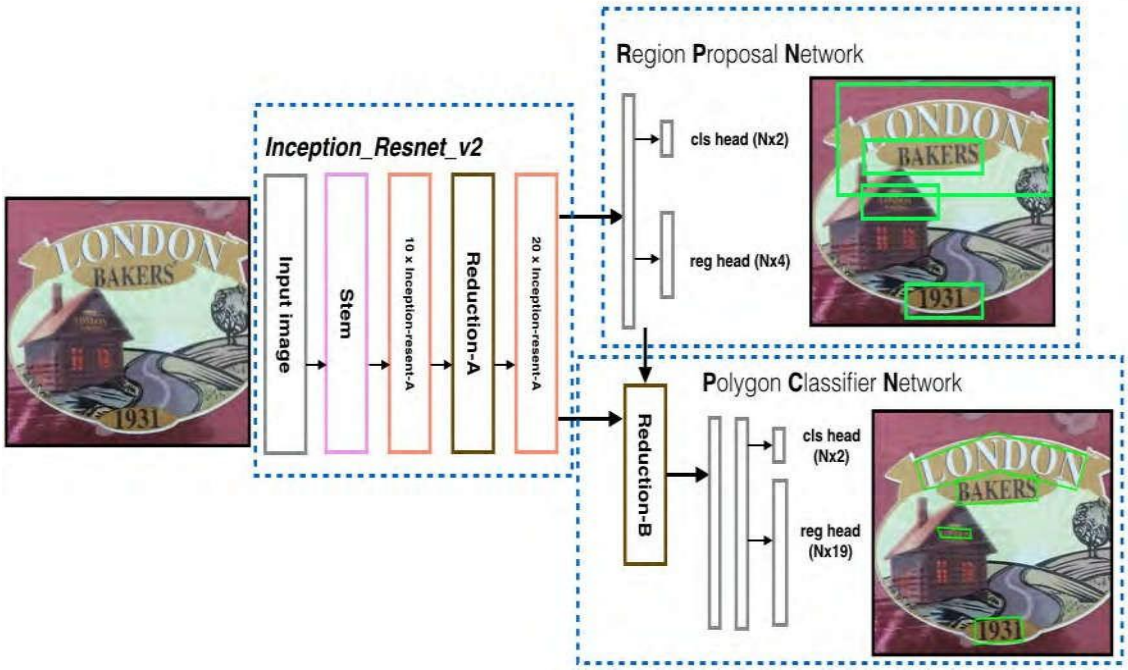


Figure 6: The overview of Polygon-FRCNN. It is based on Faster-RCNN with the modification in the second stage regression head.

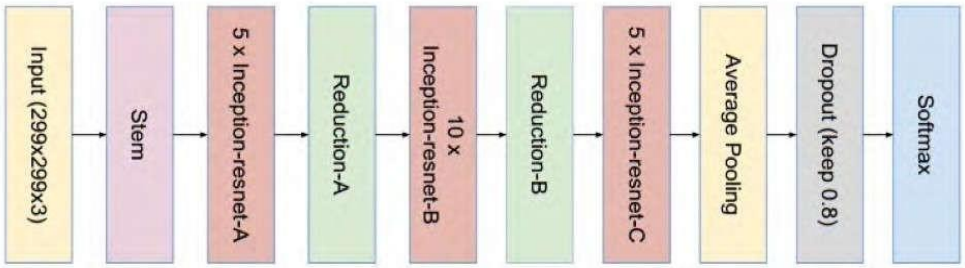
Table 1: Network details of the feature extractor (based on Inception-ResNet-v2).Note that *Mixed_5b*, *Reduction A*, and *Inception resnet B* are made up of the combination of convolution

and pooling operations (which can be referred to in Figure 2) with multiple sets of parameters (i.e. kernel size, padding, and stride). Hence, they are not tabulated for the sake of the consistency of the table.

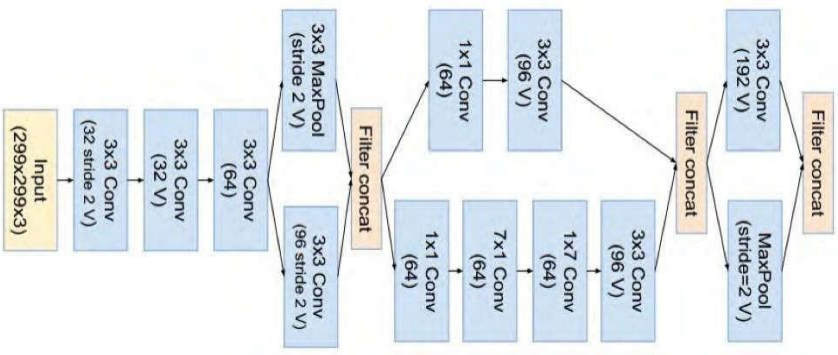
Layer/block	input size	kernel size, k ($h \times w \times c \times num\_filters$)	padding, p	stride, s	output size
Conv2d_1a	$299 \times 299 \times 3$	$3 \times 3 \times 3 \times 32$	0	2	$149 \times 149 \times 32$
Conv2d_2a	$149 \times 149 \times 32$	$3 \times 3 \times 32 \times 32$	0	1	$147 \times 147 \times 32$
Conv2d_2b	$147 \times 147 \times 32$	$3 \times 3 \times 32 \times 64$	0	1	$147 \times 147 \times 64$
MaxPool_3a	$147 \times 147 \times 64$	$3 \times 3 \times 64$	0	2	$73 \times 73 \times 64$
Conv2d_3b	$73 \times 73 \times 64$	$1 \times 1 \times 64 \times 80$	0	1	$73 \times 73 \times 80$
Conv2d_4a	$73 \times 73 \times 80$	$3 \times 3 \times 80 \times 192$	0	1	$71 \times 71 \times 192$
MaxPool_5a	$71 \times 71 \times 192$	$3 \times 3 \times 192$	0	2	$35 \times 35 \times 192$
Mixed_5b	$35 \times 35 \times 192$	–	–	–	$35 \times 35 \times 320$
Inception-resnet-A	$35 \times 35 \times 320$	–	–	–	$35 \times 35 \times 320$
Reduction-A	$35 \times 35 \times 320$	–	–	–	$35 \times 35 \times 1088$
Inception-resnet-B	$35 \times 35 \times 1088$	–	–	–	$17 \times 17 \times 1088$

The input image of the network is restricted to 600 - 1024 pixels, input image exceeding the range will be resized with the aspect ratio intact. At the end of the feature extraction module, a feature map of dimension ($h_{f\ eature} \times w_{f\ eature} \times 1088$) will be generated, where $h_{f\ eature}$ and $w_{f\ eature}$ are modelled with Equation 1, where dim_{in} and dim_{out} represent input and output dimension of feature maps respectively. Table 1 contains allthe details of the Inception-ResNet-v2 based feature extractor. Meanwhile, it provides a demo calculation based on Equation 1 assuming the input image has a spatial dimension of (299×299). The extracted feature maps will then be shared between the region proposal network and polygon classifier network.

$$dim_{out} = [(dim_{in} + 2p - k)/s] + 1 \quad (1)$$



(a)



ISSN NO: 2249-3034

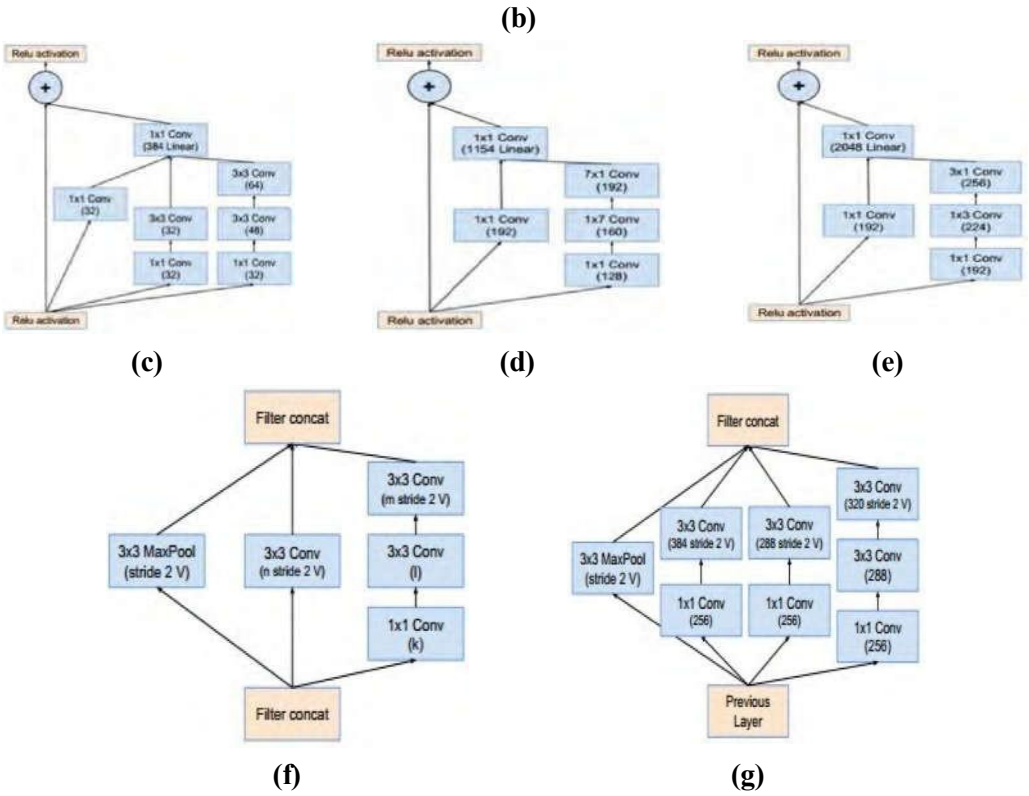


Figure 7: Overall architecture and its internal network details of Inception- ResNet-v2. (a): Overall architecture, (b): *'Stem'* block, (c): *'Inception-resnet- A'* block, (d): *'Inception-resnet-B'* block, (e): *'Inception-resnet-C'* block, (f): *'Reduction-A'* block, (g): *'Reduction-B'* block.

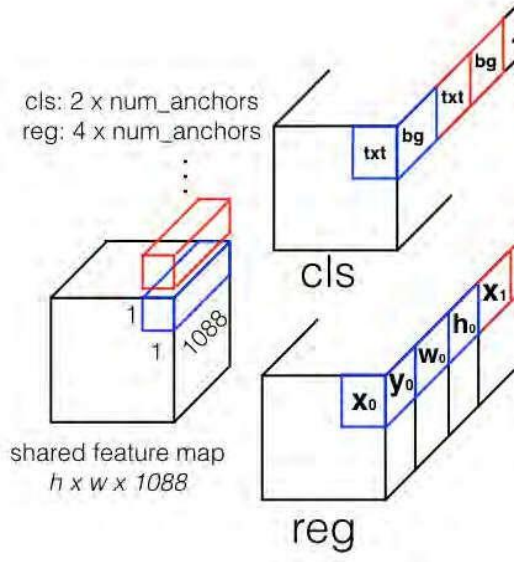


Figure 8: Classification (cls) and regression (reg) head of the Region Proposal Module.

3.3 Region Proposal Network

Upon transforming the input image to high dimensional feature maps (with the dimensions of ($h_{feature} \times w_{feature} \times 1088$)), a classification and regression head are attached tothe feature map with the objective of i) classifying whether text exists, and ii) regress itslocation in axis-aligned bounding box format.

Fundamentally, the classification 'head' is a convolutional operation with the filter size of $1 \times 1 \times 103.88 \times (num\_of\_anchors \times 2)$ over every pixel in the extracted feature maps. In other words, the filter will learn to predict an input-dependent size of two-layer probability maps, indicating the likelihood of each pixel being 'text' or 'background' region. The parameter *num_of_anchors* is associated with the design of anchors, which will be explained soon. Similarly, the re- gression 'head' is also a convolutional operation, but with the filter size of $1 \times 1 \times 1088 \times (num\_of\_anchors \times 4)$. The '4' here represents the encoded axis-aligned bounding box, *(x, y, w, h)*. Figure 8 illustrates both the classification and regression heads with severaldetails that will be referred to in the following part.

3.4 Polygon Classifier Network

The objective of Polygon Classifier Network (PCN) is to refine coarsely proposed boxesto produce precise polygons. The design of PCN is based on Box Classifier Network (BCN) from the original Faster-RCNN architecture. On top of the traditional bounding box regression head, PCN is modelled to regress polygon shaped bounding region as well.The lower level details to adapt such change are detailed in this section. As illustrated in Figure5, the PCN takes both the feature maps and proposal boxes as input. Firstly, theshared feature is cropped based on the proposed regions from the RPN. Since proposal boxes come in arbritrary size, a resize operation is required before feeding them into thefully-connected layers. Hence, all the cropped regions will be resized to 17x17 spatially,while retaining the third dimension size. Next, the cropped and resized proposal regionswill be fed into the average pooling layer, through the fully-connected layer to the final prediction layer. The final output is two vectors representing i) the confidence score of theparticular proposal being a text instance and ii) the combined parameters of a bounding box and N-vertex polygon bounding region. All of these operations are detailed in Figure 9 for better visualization as well.

Algorithm 2 Non-maximum-surpression algorithm for the proposal boxes filtering pro-cess.

Let the input proposal boxes be $proposal_n$
Let the input classification scores be $scores_n$
Let the processed proposal be *processedProposals*
Let iouThreshold = 0.6 Sort
$scores_i$ descendingly
Sort $proposal_i$ based on the *ID* of sorted $scores_i$
Remove $proposal_0$ from $proposal_i$
while length($proposal_i$) > 1 **do**
  for all all i such that $0 < i < n$ **do**
    $IoU = (proposal_0 \cap proposal_i)/(proposal_0 \cup proposal_i)$
    if IoU >= iouThreshold **then**
      Remove $proposal_i$ from $proposal_n$
    else
      Continue
    end if
    Append the *processedProposals* with $proposal_0$
  end for
end while
Append the *processedProposals* with the last proposal in $proposal_n$

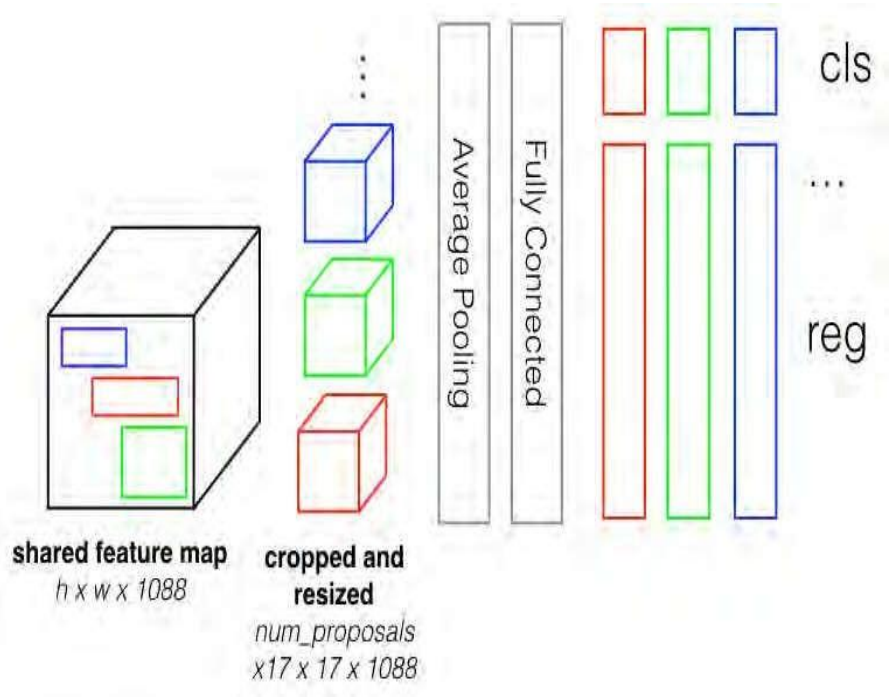


Figure 9: Architecture of Polygon Classifier Network.

3.4.1 Encoded Polygon Regression Target

The convention regression target (x_m y_m, w, h)[3] used in Faster-RCNN, SSD, and YOLO only applies to axis-aligned rectangular boxes, but not polygon. Y. Liu & Jin (2017) introduced an encoding method for quadrilateral and claimed that relative information is easier to train instead of absolute values of quadrilateral vertices (i.e., $(x_0, y_0, x_1, y_1, x_2, y_2, x_3, y_3)$). In fact, most of the regression-based scene text detectors were modelled to regress one kind of encoded quadrilateral or another.

3.4.2 Encode

Decoding process is needed to convert the predictions to actual polygon vertices. Duringdecoding, all the middle points from second points onwards (i.e., $i = [1, N/2)$) will firstbe determined with Equation 2.

$$h(i) = abs(y(i) - y((N-1)-i)) w(i)1 = x(i) - x_{mid}(i)$$

$$w(i)2 = x((N-1)-i) - x_{mid}(i) \; For \; i = [0, N/2) \qquad (2)$$

Note that the encoding method is compatible to any polygon with *even* number vertices. However, for the sake of simplic- ity, all the explanation henceforth assume the 6-vertex polygon. Hence, utilizing the same example as earlier (i.e., taking the $\{x_0, y_0\}$ and $\{x_5, y_5\}$ pair as example):

1) $h_{(i)}$ is the height between y_0 and y_5,

2) $w_{(0)1}$ is the width between $x_{mid(0)}$ and x_0, and

3) $w_{(0)2}$ is the width between $x_{mid(0)}$ and x_5.

The reason for two width values ($w_{(i)1}$ and $w_{(i)2}$) is to ensure that the model is flexible enough to cover the cases of $x_i > x_{N-i-1}$ and $x_i < x_{N-i-1}$, which are pretty common in the shape of curved text. As a result, the polygon ground truth with the original form of (x_0, y_0, x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4, x_5, y_5) is encoded into fifteen relative information ($y_{mid}(0)$, $x_{mid}(0)$, $h(0)$, $w(0)1$, $w(0)2$, $dy_{mid}(1)$, $dx_{mid}(1)$, $h(1)$, $w(1)1$, $w(1)2$, $dy_{mid}(2)$, $dx_{mid}(2)$, $h_{(2)}$, $w_{(2)1}$, $w_{(2)2}$).

3.4.3 Decode

Decoding process is needed to convert the predictions to actual polygon vertices. Duringdecoding, all the middle points from second points onwards (i.e., $i = [1, N/2)$) will firstbe determined with Equation 3.

$$x_{mid}(i) = x_{mid}(i-1) + dx_{mid}(i) y_{mid}(i) = y_{mid}(i-1) + dy_{mid}(i)$$

$$For\ i = [1, N/2) \tag{3}$$

Then h_i, $w_{(i)1}$, and $w_{(i)2}$ will be used to transform these middle points to N vertices usingEquation 4-5.

$$x(i) = x_{mid}(i) - w(i)1$$

$$y(i) = y_{mid}(i) - h(i)/2 For\ i = [0, N/2) \tag{4}$$

$$x(i) = x_{mid}(i) + w(i)2$$

$$y(i) = y_{mid}(i) + h(i)/2 For\ i = [N/2, N) \tag{5}$$

$y_{(0)}$, $y_{(1)}$, and $y_{(2)}$ will always be the opposite of $y_{(5)}$, $y_{(4)}$, and $y_{(3)}$ respectively with the height values: $h_{(0)}$, $h_{(1)}$, and $h_{(2)}$ (likewise for $x_{(i)}$ with $w_{(i)1}$ and $w_{(i)2}$).

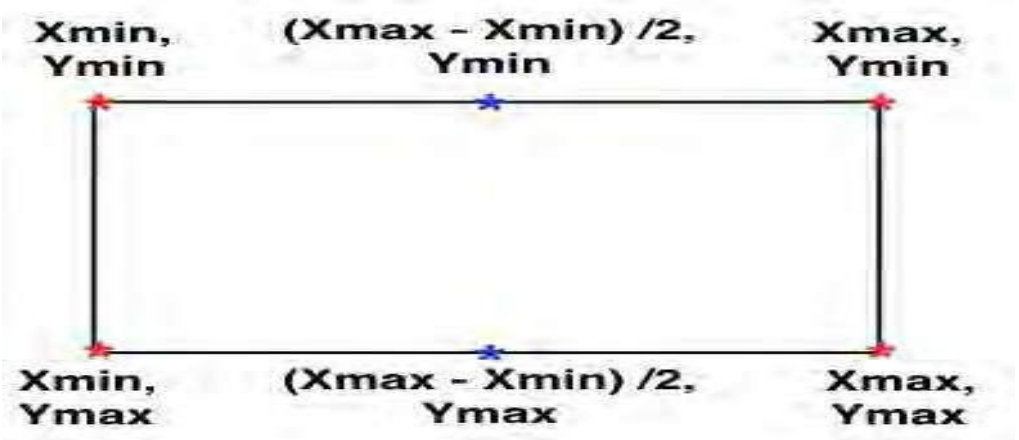
3.5 Matching between Axis-aligned Proposal Boxes and Ground-truth Polygons

The algorithms of identifying overlapping region that involves a polygon is complicated.They usually consist of heuristic rules and multiple processes which would slow down the execution time significantly. Such trait is not desirable in the design of a modern scene text detectoras minimal execution

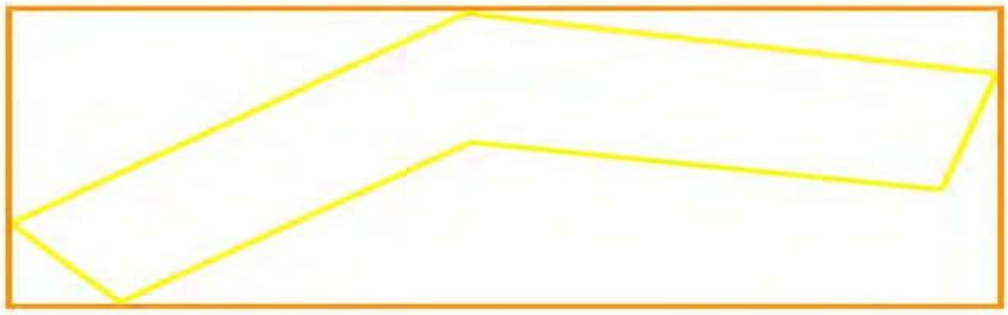
speed is often demanded. The design of Polygon-FRCNN sidestepsthis problem with an approximation solution: converting polygon to a circumscribed rectangle. Figure 10 shows the said circumscribed rectangle of a polygon. Consequently, the problem is simplified to just finding the overlapping region between two rectangles. Note that such conversion only applies to the matching process, the ground truths remainin polygon format for the following process.

3.5.1 Parameterization between Proposal and Ground-truth Polygons

Upon identifying the matching proposal and ground truth pairs, the next step is to parameterize them. Unlike the original parameterization equation as stated.



(a)



(b)

Figure 10: Methods employed in the PCN matching and parameterization process:(a): increasing the number of vertices of proposal box to match the 6-vertex polygonground truth. (b): converting polygon (yellow) to a circumscribed rectangle (orange) for the matching process.

Which was derived for encoded axis-aligned box only, this work extended it to fit the proposed encoded polygon. Firstly, the axis-aligned bounding box outputs (four ver-tices, (x_{min}, y_{min}, x_{max}, y_{max})) from the RPN need to be converted to *N*-vertex polygon sothat both have equal variables to be parameterized. The conversion process is a simple operation which adds interval points to the original axis-aligned proposal box. The said process can be visualized in Figure 4.6a, the original 4-vertex bounding box (representedby four red '*') is added with two middle points (represented by two blue '*'), which in turn became compatible with the 6-vertex polygon ground truth for the parametrization process as formulated in Equation 6.

$$
\begin{aligned}
tx_{mid}(0) &= (x_{mid}(0) - xa_{mid}(0))/wa \\
ty_{mid}(0) &= (y_{mid}(0) - ya_{mid}(0))/ha \\
t_h(i) &= \log(h_{(i)}/h_a(i)) \\
t_{w(i)1} &= (w(i)1 - wa_{(i)1})/wa_{(i)1} \\
t_{w(i)2} &= (w(i)2 - wa_{(i)2})/wa_{(i)2} \; For\; i = [0, N/2) \\
tdx_{mid}(j) &= (dx_{mid}(j) - dxa_{mid}(j))/wa \quad tdy_{mid}(j) = (dy_{mid}(j) - dya_{mid}(j))/ha \; For\; j = [1, N/2)
\end{aligned}
\tag{6}
$$

Where x and x_a represent the encoded predicted polygon (also the encoded ground truth polygon) and encoded anchor polygon respectively (same applies to y, h, w_1, w_2, dx, and dy).

4. Implementation Details

Polygon-FRCNN was built on top of the Tensorflow object detection API (J. Huang et al. (2016)). All of the training and experiments were ran on a machine with Intel 6-core Xeon chip, 32GB of RAM, Nvidia Titan X Pascal Architecture GPU, and Ubuntu OS v14.04.

4.1 Variants of Polygon-FRCNN

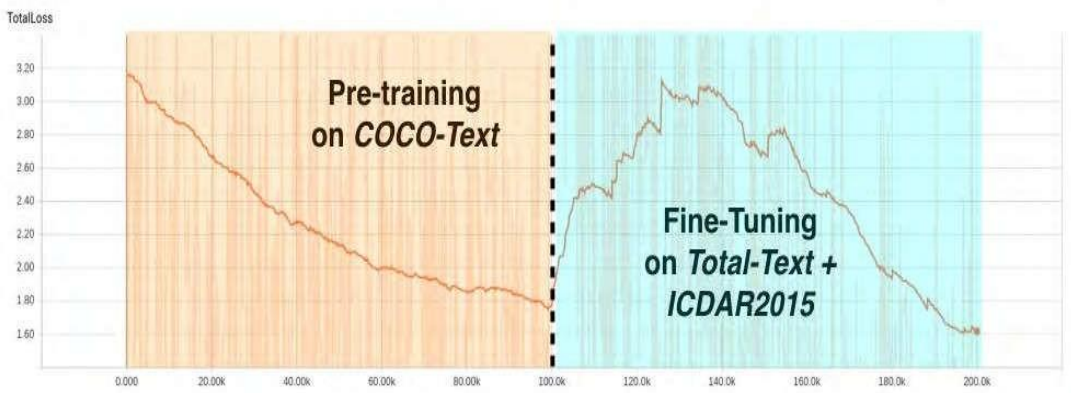
Polygon-FRCNN-3 is designed to produce six vertices polygon bounding region. The proposed encoding method splits six vertices into three opposing pairs, hence the notation '3'. The overview of Polygon-FRCNN-3 architecture can be seen in Figure 1. It is not perfect as some of the bigger curved text instances require more than six vertices to fit. In such cases, the bounding region will be looser in order to fit in the entire text region.

ISSN NO: 2249-3034

4.2 Training Process

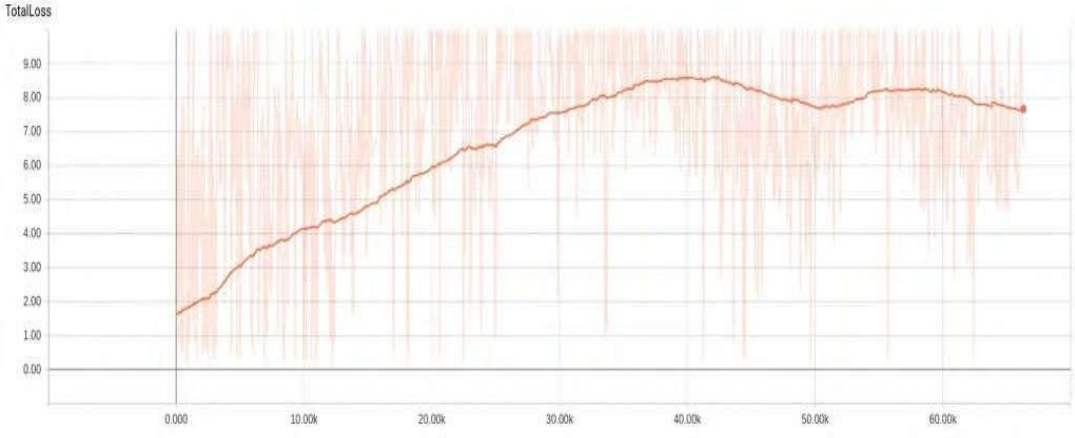
Pre-Training

The weights of Polygon-FRCNN was initialized with the ImageNet-pretrained weights.[4] Such initialization is important as it gives the network a general idea on what to lookfor since both ImageNet and scene text datasets are made up of images in natural scene. Additionally, training such a deep network from scratch is very time consuming (weeks of time), hence initializing a network with pre-trained model became a norm in the deeplearning community (Sharif R. et al. [51]; Yosinski et al. [52]). Upon initialization, Polygon-FRCNN was trained on COCO-Text for 100K iterations. It is a good practice to pre-train a deep network on a larger dataset first, before theactual trainingdata which is smaller in scale (*Total-Text* in this case). Doing so would pre-vent overfitting, and help the network to converge faster (Yosinski et al. [55]) during the actual training. Utilizing the extra annotation of COCO-Text, roughly 15K images with at least one legible text instance were retrieved for the pre-training process. Consistent with J. Huang et al. [53]'s study, the learning rate was set to 0.003 during the pre-training stage. After the pre-training process, Polygon-FRCNN was fine-tuned on the combination *Total- Text* and ICDAR2015 training set for another 100K iterations. The merging of dataset was done to increase the scale of training data, which was proven benefitial to the trainingof deep neural network models (LeCun et al. [54]). The combined dataset has 2000 training images in total, with 1000 images each from the training set of *Total-Text* and ICDAR2015. 255 images from the *Total-Text* training set was withhold to serve as the validation set, which was used to monitor the performance of the network during the training process, as plotted in Figure 4.7c. Yosinski et al. [56]'s work suggested to reduce the learning rate when fine-tuning a network on a much smaller scale dataset, hence it was decreased to 0.0003 for this stage of the training process. As shown in Figure 11 the loss increases abruptly at the 100K mark, when the fine-tuning process starts. That sudden increase was caused by the newly introduced training data, the network hasto adjust to the emergence of curved text training data. The loss started to decrease again around the 130K mark and reached a plateau around 195K iteration. The performance ofthe network on the validation set is seen to improve gradually until the 160K iteration mark, and saturated around 0.5-0.6 for the rest of the training.

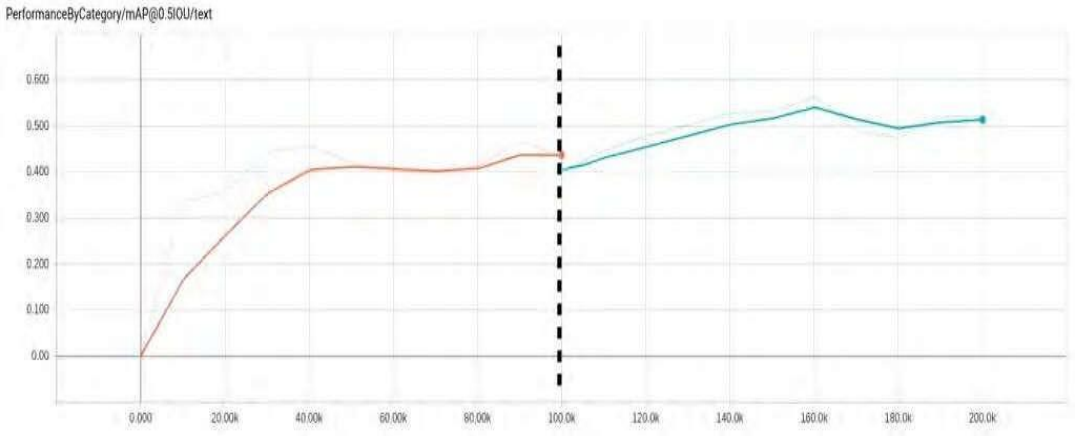


ISSN NO: 2249-3034

(a) Training loss of Polygon-FRCNN.



(b) Training loss of Polygon-FRCNN without pre-training it on COCO-Text.



(c) Performance of Polygon-FRCNN throughout the training

Figure 10 : Losses and performance of Polygon-FRCNN throughout the trainingprocess.

5. Experiments

Several experiments were carried out to evaluate the effectiveness of Polygon-FRCNN. This section describes all the experiment configurations, results, and some of the insightsdrawn upon them.

i. Datasets

Polygon-FRCNN was benchmarked on ICDAR2013, ICDAR2015, and *Total-Text*. IC- DAR2013 and ICDAR2015 were chosen to demonstrate the performance of Polygon- FRCNN on horizontal and multi-oriented text respectively.

ii. Evaluation Protocol

In order to ensure the legitimacy of the comparison carried out in this section, all the benchmarkings were done following the standard evaluation protocol and configuration of ICDAR2013 and ICDAR2015. Firstly, all the results on Table 2 were obtained using the DetEval protocol with the default thresholds (i.e. tr=0.8 and t p=0.4) (Wolf & Jolion (2006)). On the other hand, the Pascal VOC evaluation method is the official protocol for ICDAR2015. The results on Table 3 were recorded with the standard 0.5 IoU threshold. Both DetEval and Pascal VOC methods were made available for Total-Text. The threshold values for DetEval, tr and t p were configured to 0.7 and 0.6 respectively based on the findings. Meanwhile, for the Pascal VOC evaluation method wise, the standard 0.5 IoU threshold was chosen.

iii. Quantitative Result

Table 2 and 3 show that the performance of Polygon-FRCNN-3 (our best performingmodel) is on par with contemporary state-of-the-art works like Jiang et al. [57]; Liao et al. [58]; Y. Liu & Jin [59]; Ma et al. [60]; B. Shi et al. [61]. On the other hand, Table 4.4 tabulates the performance of Polygon-FRCNN on *Total-Text*. In order tomeasure the effectiveness Polygon-FRCNN againsts of text with different orientations, especially curved text, two subsets of *Text Detection* were created with the orientation annotation. The 'curved-set' in Table 4.4 consists of only curved text; while the 'non-curved-set' is made up of horizontal and multi-oriented text. There are three methods being comparedon *Text Detection*: Box-FRCNN, Polygon-FRCNN-3, and Polygon-FRCNN5. Box-FRCNN is the **axis-aligned box** output from the same network as Polygon-FRCNN-3, being com-pared to show the effectiveness of the **polygon** predicting Polygon-FRCNN-3. Mean- while, Polygon-FRCNN-5 is the scaled up version.

Table 4.2: Evaluation Result on ICDAR2013

Methods	Precision	Recall	F-measure
TextBoxes (Liao et al. (2017))	0.89	**0.83**	0.86
R^2CNN (Jiang et al. (2017))	**0.94**	**0.83**	**0.88**
Seglink (B. Shi et al. (2017))	0.88	**0.83**	0.85
RRPN (Ma et al. (2017))	0.90	0.72	0.80
DIRECT (W. He et al. (2017))	0.92	0.81	0.86
Polygon-FRCNN-3	0.89	0.81	0.85

As observed in Table 4, Polygon-FRCNN-3 outperforms its box predicting counterpart (Box-FRCNN) by 0.16 (0.12) in terms of F-measure overall (wholeset). Their performance differences became much more significant on the curved text only subset *Total-Text*, with a gap of 0.3 (0.2). This result

demonstrates that the design of the pro- posed method is effective in detecting curved text. The performance of Polygon-FRCNN-5 drops slightly by 0.04 (0.03) in terms of F-measure in compared to Polygon-FRCNN-3.The scaled up version is expected to bind curved text region smoothly with more vertices;however, the increase in number of vertices also reduces the error margin of each vertices,which causes the loss in performance. Another worthy observation is that the performance of the proposed method is loweron *Total-Text* than on both of the ICDAR2013 and ICDAR2015 datasets. Visualized comparison can be seen in Figure 11. This observation shows that *Total-Text* is more challenging than both of the well-received scene text datasets in the text community. Lastly, the execution speed of Polygon-FRCNN is 1.47s in average across all 300 testingimages in the proposed dataset.

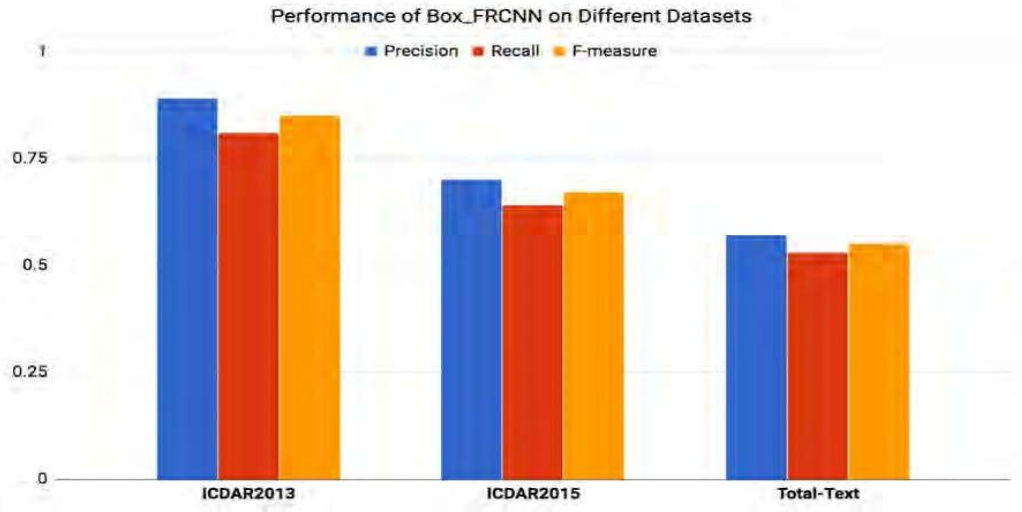


Figure 11: Performance of Box-FRCNN on different datasets. It achieves the best scores in terms of all three metrices on ICDAR2013, followed by ICDAR2015, loweston *Total-Text*.

Table 3: Evaluation Result on ICDAR2015

Methods	Precision	Recall	F-measure
DMPNet	0.73	0.68	0.70
R^2CNN	**0.86**	0.80	**0.83**
Seglink	0.73	0.77	0.75
RRPN	0.73	**0.82**	0.77
DIRECT	0.82	0.80	0.81
Polygon-FRCNN-3	0.76	0.67	0.72

iv. Qualitative Result

The detection examples of Polygon-FRCNN-3 can be seen in Figure 11, 12, 13, 14. The contrary examples of text instances in both Figure 4.10 and 4.11 illustrate that Polygon-FRCNN-3 is robust against text with different sizes, orientations and perpestive distortion.

Table 4: Evaluation of Polygon-FRCNN on Total-Text. All the models were evalu-ated on both DetEval and Pascal VOC (in the bracket) evaluation protocol. Legends:P = Precision, R

= Recall, F = F-measure.

Methods	wholeset		
	P	R	F
Box-FRCNN	0.50(0.57)	0.46(0.53)	0.48(0.55)
Polygon-FRCNN-3	**0.67**(**0.69**)	**0.61**(**0.64**)	**0.64**(**0.67**)
Polygon-FRCNN-5	0.64(0.69)	0.56(0.6)	0.6(0.64)
	curved-set		
Box-FRCNN	0.19(0.3)	0.17(0.29)	0.18(0.3)
Polygon-FRCNN-3	**0.51**(**0.52**)	**0.45**(**0.48**)	**0.48**(**0.50**)
Polygon-FRCNN-5	0.44(0.5)	0.38(0.44)	0.41(0.47)
	non-curved-set		
Box-FRCNN	0.50(0.51)	0.58(0.61)	0.54(0.55)
Polygon-FRCNN-3	0.63(0.64)	**0.59**(**0.62**)	**0.61**(**0.63**)
Polygon-FRCNN-5	**0.63**(**0.65**)	0.57(0.59)	0.59(0.62)

On top of that, Figure 12 shows that the polygon output of Polygon-FRCNN-3 is capable of binding text of all orientations tightly. Meanwhile, Figure 13 emphasizes theeffectiveness of polygon output in compared to box output when it comes to curved textinstances. Both of these visual examples validate the performance advantage of Polygon-FRCNN-3 against Box-FRCNN as tabulated in Table 4. On the other hand, Figure 14 visualizes the difference between Polygon-FRCNN-3 and the scaled up Polygon-FRCNN-5. The detection output of Polygon-FRCNN-5 is observed to bind curved text with larger curvature more tightly as expected. Lastly, Figure shows that the proposed baseline for Total-Text, Polygon-FRCNN- 3 is not perfect. Polygon has more vertices than axis-aligned bounding box and quadri- lateral (i.e., six for Polygon-FRCNN-3, in compared to two and four for axis-aligned bounding box and quadrilateral respectively), which makes it more prone to error. As observed in Figure 15, although the proposals were mostly right about the location of the text, but several errors in the vertices regression caused the polygon to miss the propertext region. Other failure cases include the shortcoming of Polygon-FRCNN-3 to bind text with larger curvature tightly (1st row, 2nd column), and complete miss detections (1st row, 3rd column).

ISSN NO: 2249-3034

Figure 12: Comparison between detection examples of Box-FRCNN and Polygon-FRCNN.

6. Conclusion

As a summary, Polygon-FRCNN is a scene text detector that was designed with curved text in consideration. It was built on top of the groundbreaking object detection frame- work, Faster-RCNN. Changes of design were made to equip it with the ability to producepolygon bounding region, which is capable of detecting curved text. One of the main changes is the introduction of the polygon encoding method to prepare for the network regression target. Multiple experiments were conducted to evaluate the performance of the proposed method. The benchmark result on ICDAR2013 and ICDAR2015 have shown that Polygon-FRCNN is on par with the state-of-the-art scene text detectors. Moreover, ithas proven to be a strong baseline for the proposed dataset, *Total-Text*. The effectiveness of polygon bounding output against text of all orientations is shown with the result break-down of the *Total-Text* dataset as well. Last but not least, the significance of the curved text data in the proposed dataset was presented as well.

ISSN NO: 2249-3034

Figure 13: Detection examples of Polygon-FRCNN-3 on ICDAR2013.

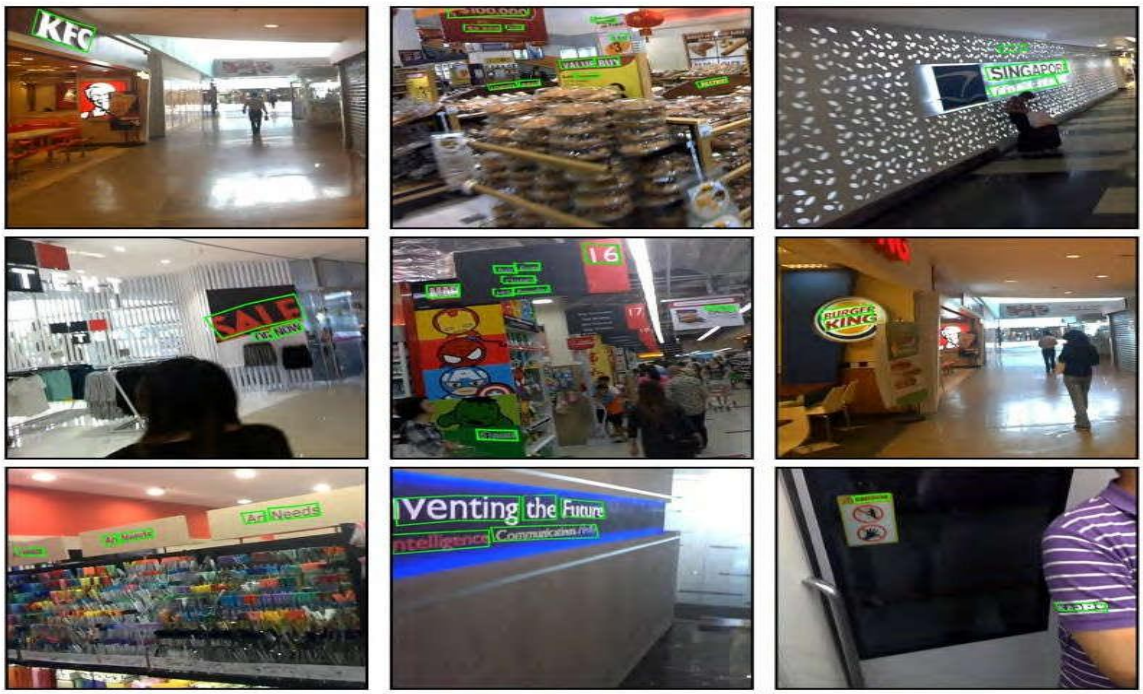


Figure 14: Detection examples of Polygon-FRCNN-3 on ICDAR2015.

ISSN NO: 2249-3034

Figure 15: Successful detection examples of Polygon-FRCNN-3 on Total-Text. Itshows that polygon shaped output is effective against text of all orientations.

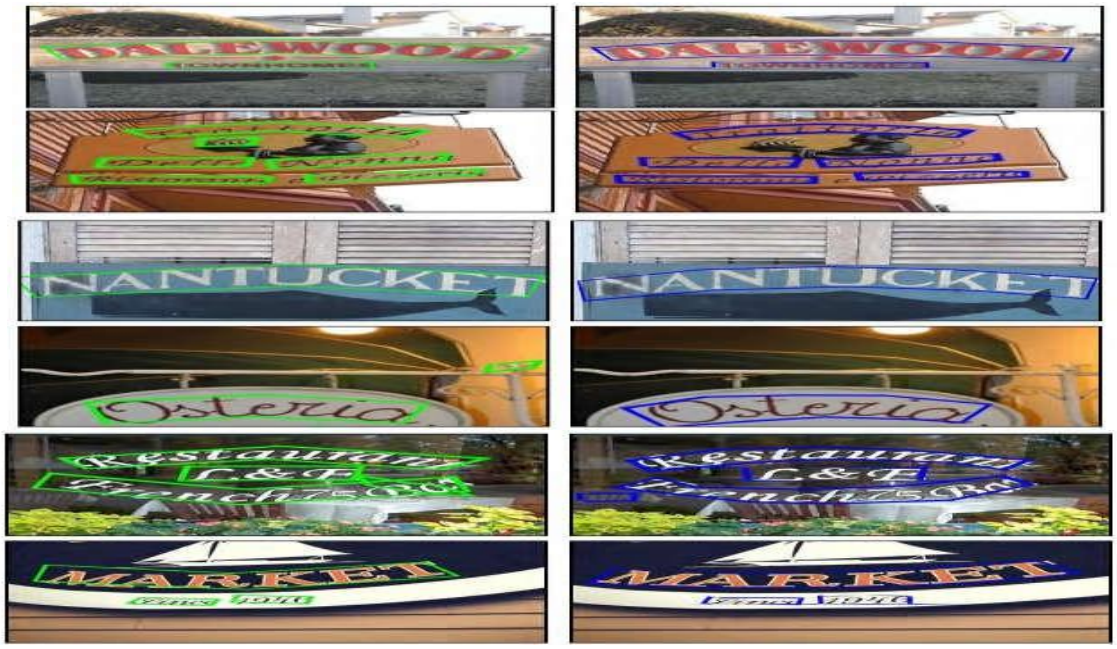


Figure 16: Left: Polygon-FRCNN-3; Right: Polygon-FRCNN-5. Polygon- FRCNN-5's output has more vertices, is able to bind curved text with larger cur- vature more tightly.