SCENE TEXT DETECTION

Computer Vision

WHAT WE'LL DISCUSS

1. Introduction
2. Challenge
3. Method
4. Data overview
5. Data augmentation
6. Model
7. Result

1. Introduction

In recent years, reading text in scene images has become an active research area, due to its wide practical applications such as image/video understanding, visual search, automatic driving, and blind auxiliary.

2.Challenge

Natural scene images may be blurred, noisy, and in low quality and in severe weather, or multi-oriented(rotated/ curved).

3. Method

We decide to build 2 Deep Learning-based end-to-end scene text detection model

1.SHAPE ROBUST TEXT DETECTION WITH PROGRESSIVE SCALE EXPANSION NETWORK

2.REAL-TIME SCENE TEXT DETECTION WITH DIFFERENTIABLE BINARIZATION

3.EAST: AN EFFICIENT AND ACCURATE SCENE TEXT DETECTOR

4. Data overview

- Training data: 2000 images provided by VinAI
- Testing data: 500 images provided by BK.AI
- The ground truth is given as separate text files (one per image) where each line specifies the coordinates of one word’s bounding box and its transcription in a comma-separated format.
- “Do Not Care” regions are indicated in the ground truth with a transcription of “###”


```
262,21,312,15,325,73,257,70,###
99,86,409,89,399,182,93,186,CHÂU
64,205,143,204,145,254,63,254,CHO
153,194,251,195,249,252,154,252,THUÊ
72,275,159,273,159,317,75,320,XE
172,271,300,270,303,318,173,324,ĐẠP
315,261,415,260,415,311,311,314,ĐÔI
153,327,192,327,193,348,155,350,ĐT:
197,327,340,322,339,345,194,350,0977.030977
```

Data visualize

5. DATA AUGMENTATION

Original image

CONTRAST

Camera sensors have many imperfections and tunable settings

GRID

More suitable for STD than some object detection augmentation like CutOut, CutMix and MixUp

COLOR

SHADOW

Scene text may be captured under different weather conditions

RAIN

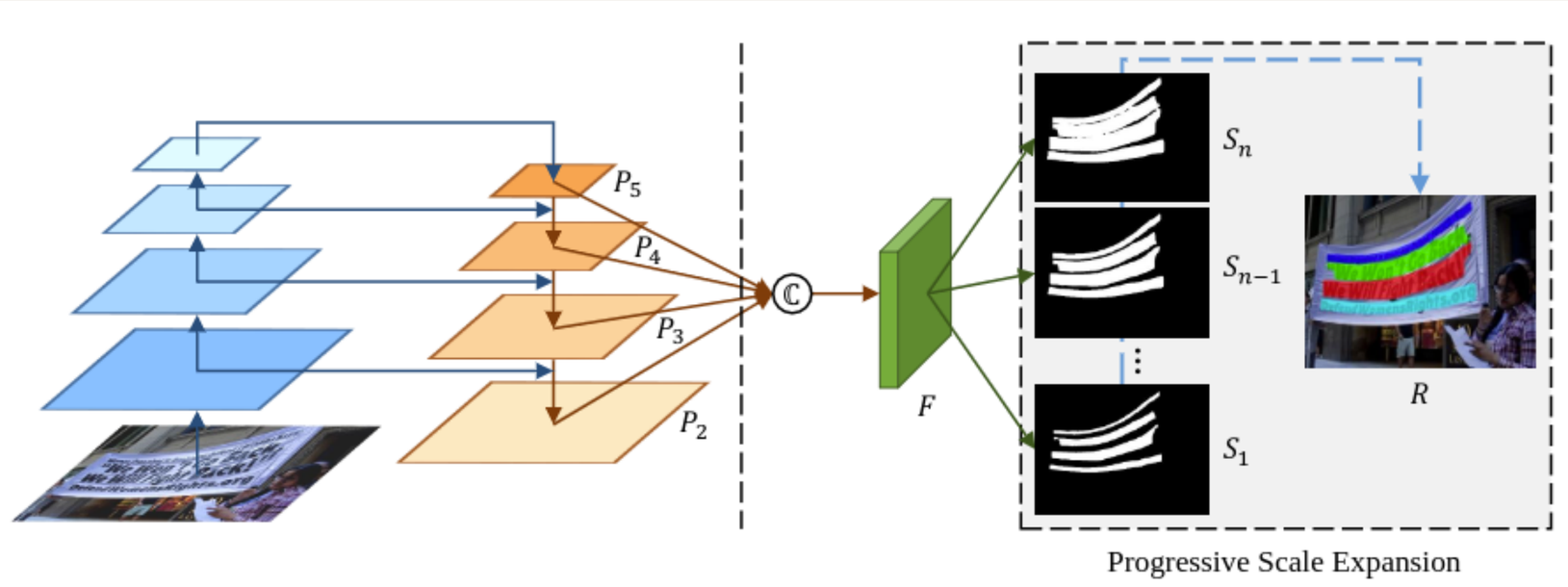
Scene text may be captured under different weather conditions

ROTATION AND SCALE

6. MODEL

6.1. PSE

Pipeline



- The left part is implemented from FPN.
- The right part denotes the feature fusion and the progressive scale expansion algorithm

Progressive Scale Expansion Algorithm

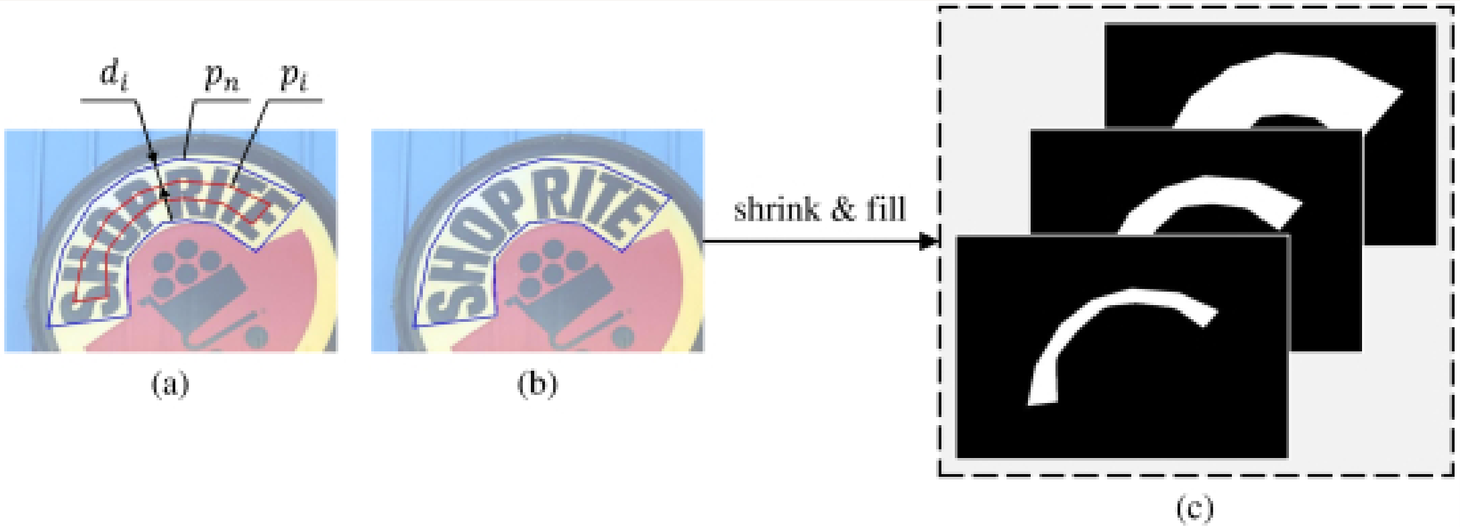
Algorithm 1 Scale Expansion Algorithm

Require: Kernels: C, Segmentation Result: S_i
Ensure: Scale Expanded Kernels: E

function EXPANSION(C, S_i)
  $T \leftarrow \emptyset$; $P \leftarrow \emptyset$; $Q \leftarrow \emptyset$
  for each $c_i \in C$ **do**
    $T \leftarrow T \cup \{(p, label) \mid (p, label) \in c_i\}$
    $P \leftarrow P \cup \{p \mid (p, label) \in c_i\}$
    Enqueue(Q, c_i) // push all the elements in c_i into Q
  end for
  while $Q \neq \emptyset$ **do**
    $(p, label) \leftarrow$ **Dequeue**(Q) // pop the first element of Q
    if $\exists q \in$ **Neighbor**(p) and $q \notin P$ and $S_i[q] =$ True **then**
      $T \leftarrow T \cup \{(q, label)\}$; $P \leftarrow P \cup \{q\}$
      Enqueue(Q, $(q, label)$) // push the element $(q, label)$ into Q
    end if
  end while
  $E =$ **GroupByLabel**(T)
  return E
end function

Label Generation

SENet produces segmentation results (e.g. S1, S2, ..., Sn) with different kernel scales. Therefore, it requires the corresponding ground truths with different kernel scales as well during training.



Loss Function

For learning PSENet, the loss function can be formulated as:

$$L = \lambda L_c + (1 - \lambda) L_s$$

where L_c and L_s represent the losses for the complete text instances and the shrunk ones respectively, and λ balances the importance between L_c and L_s.

Implementation Details

The backbone of PSENet is implemented from FPN. We firstly get four 256 channels feature maps (i.e. P_2, P_3, P_4, P_5) from the backbone. To further combine the semantic features from low to high levels, we fuse the four feature maps to get feature map F with 1024 channels via the function C(·) as:

$$F = C(P_2, P_3, P_4, P_5) = P_2||Up_{\times 2}(P_3)||Up_{\times 4}(P_4)||Up_{\times 8}(P_5)$$

6.2. DBNET

Motivation

- Segmentation based scene text detection has attracted a lot of attention recentlY
- However, most segmentation-based methods require complex post-processing for grouping the pixel-level prediction results into detected text instances, resulting in a considerable time cost in the inference procedure

Main idea

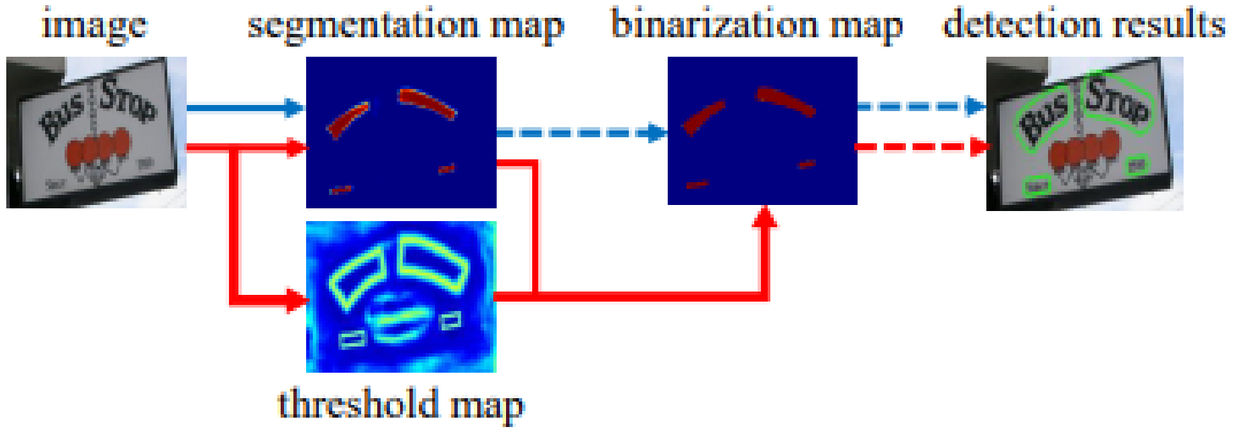


Figure 2: Traditional pipeline (blue flow) and our pipeline (red flow). Dashed arrows are the inference only operators; solid arrows indicate differentiable operators in both training and inference.

- Insert the binarization operation into a segmentation network for joint optimization.
- The threshold value at every place of an image can be adaptively predicted

=> fully distinguish the pixels from the foreground and BACKGROUND

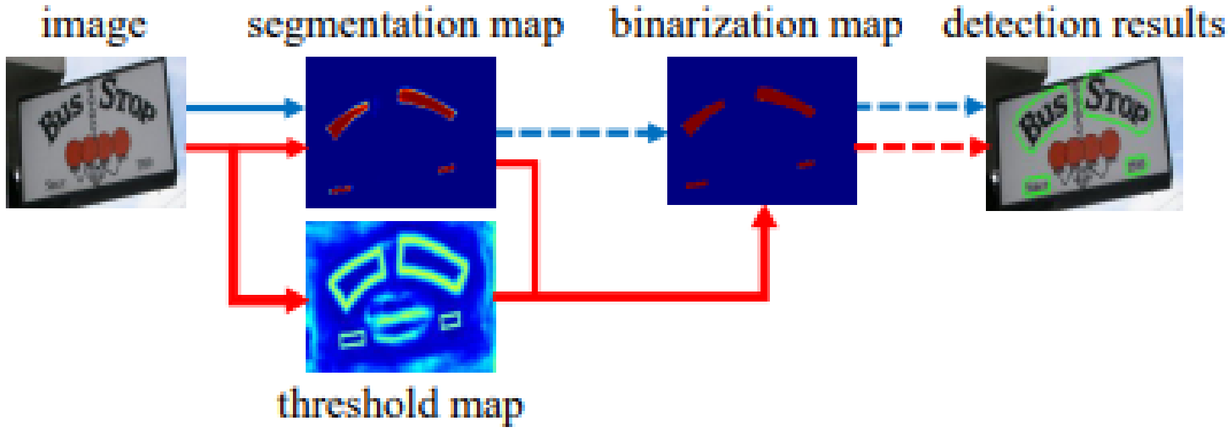
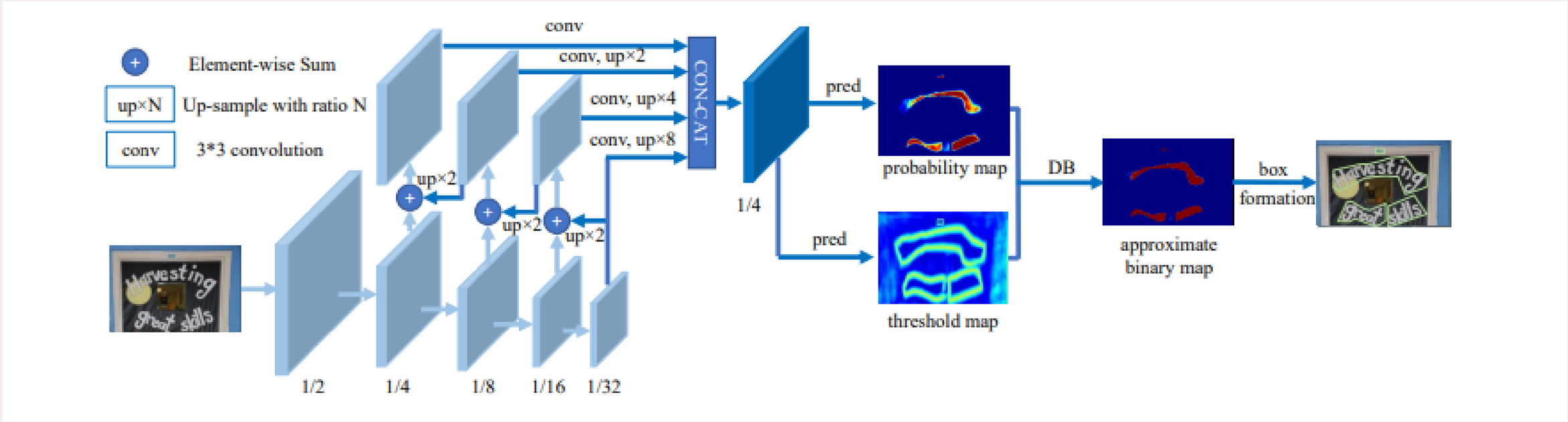


Figure 2: Traditional pipeline (blue flow) and our pipeline (red flow). Dashed arrows are the inference only operators; solid arrows indicate differentiable operators in both training and inference.

The major contribution in this paper is the proposed DB module that is differentiable, which makes the process of binarization end-to-end trainable in a CNN

Model architecture



DETAIL EXPLANATION

BINARIZATION

STANDARD BINARIZATION

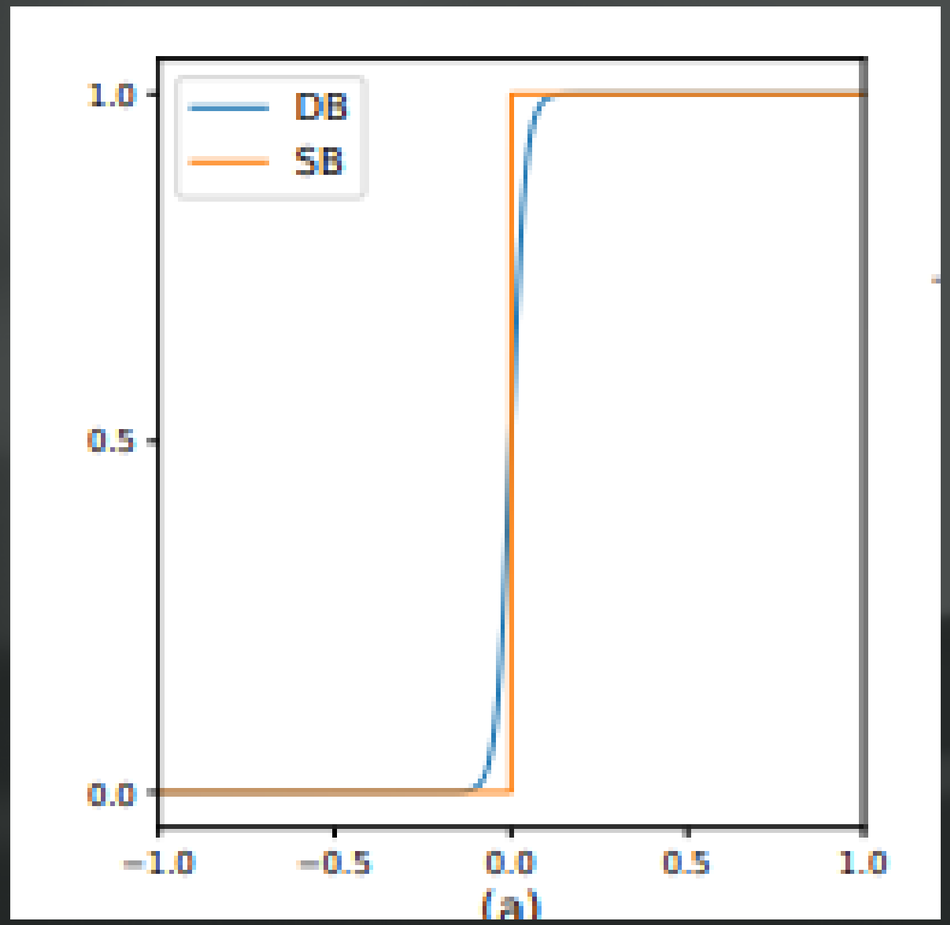
$$B_{i,j} = \begin{cases} 1 & \text{if } P_{i,j} >= t, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

where t is the predefined threshold and (i, j) indicates the coordinate point in the map.

DIFFERENTIABLE BINARIZATION

$$\hat{B}_{i,j} = \frac{1}{1 + e^{-k(P_{i,j} - T_{i,j})}}$$

where Bˆ is the approximate binary map; T is the adaptive threshold map learned from the network;



Binarization graph

DETAIL EXPLANATION

ADAPTIVE THRESHOLD

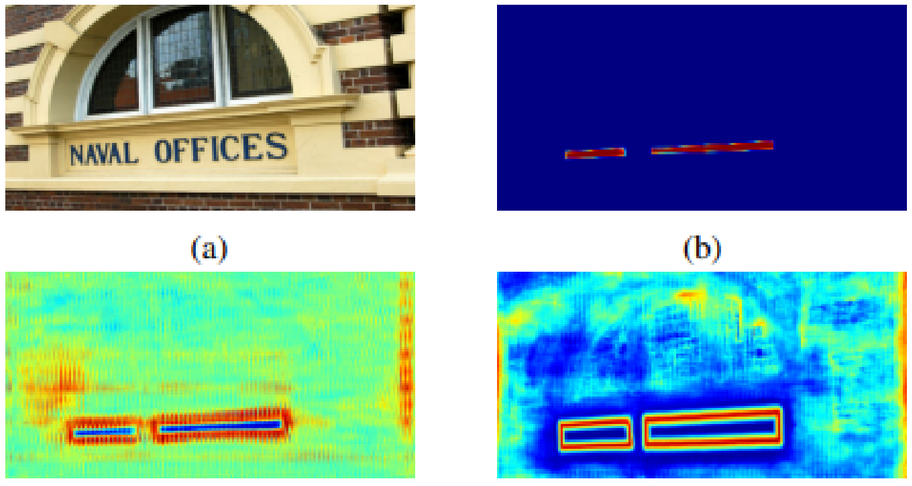


Figure 6: The threshold map with/without supervision. (a) Input image. (b) Probability map. (c) Threshold map without supervision. (d) Threshold map with supervision.

The threshold map would highlight the text border region

DETAIL EXPLANATION

DEFORMABLE CONVOLUTION

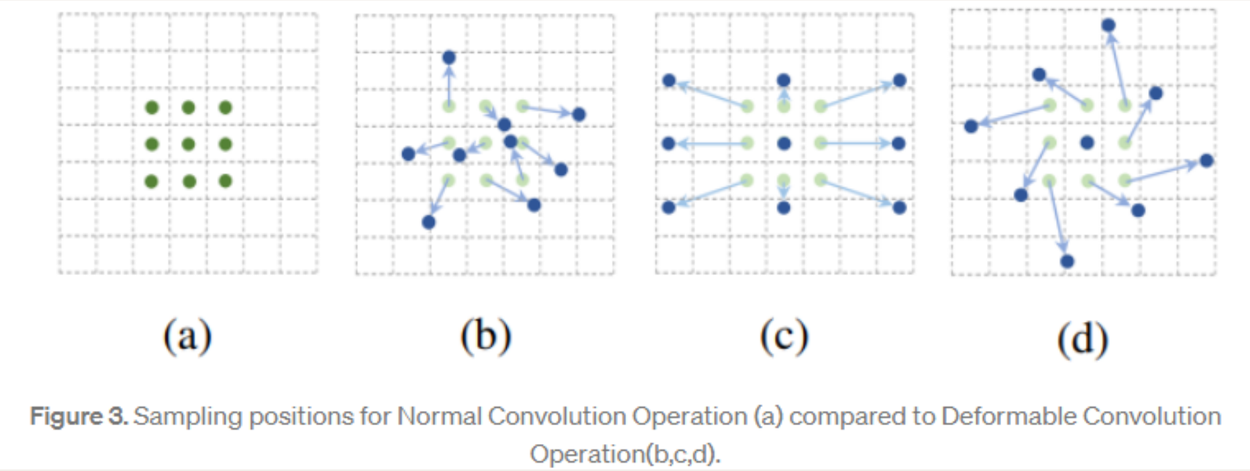


Figure 3. Sampling positions for Normal Convolution Operation (a) compared to Deformable Convolution Operation(b,c,d).

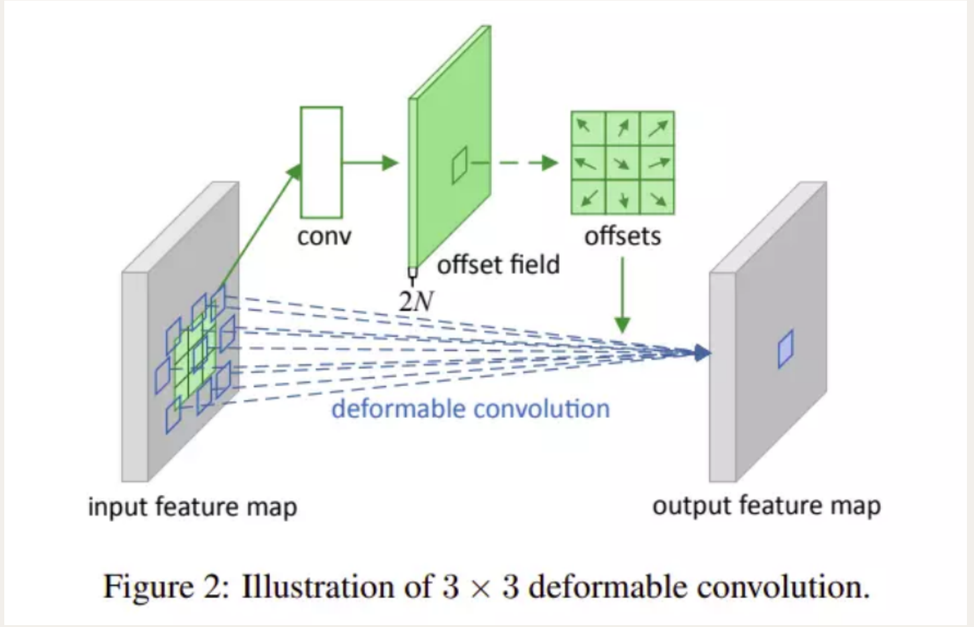


Figure 2: Illustration of 3×3 deformable convolution.

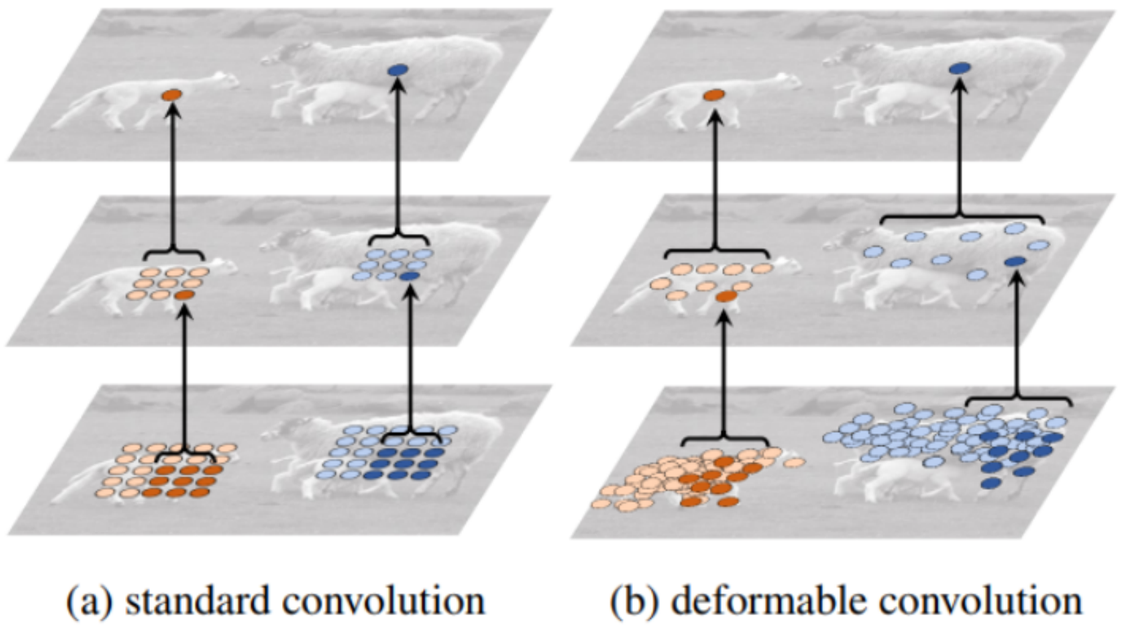


Figure 6. Illustration of fixed receptive field in standard convolution and adaptive receptive field in deformable convolution operation.

- In Deformable conv, the receptive field of a deep pixel is concentrated to the corresponding object
- How ever, in normal CNN, its rectangle receptive field also contain the little sheep

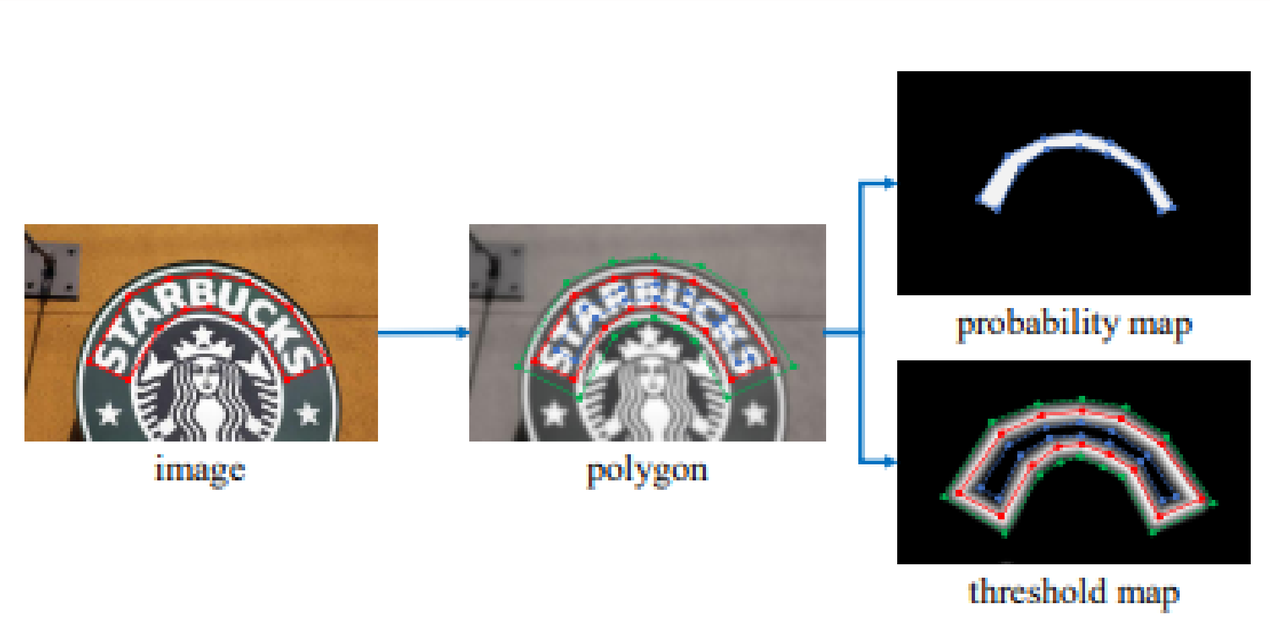
=> Bring ambiguity

The receptive field in CNN is the region of the input space that affects a particular unit of the network

Figure 7. Triplet Images depicting the adaptive receptive field for each object.

DETAIL EXPLANATION

LABEL GENERATION



- The positive area for probability map is generated by shrinking the polygon
- The label of the threshold map can be generated by computing the distance to the closest segment in G.

OPTIMIZATION

$$L = L_s + \alpha \times L_b + \beta \times L_t$$

Lt is the loss for the threshold map

$$L_t = \sum_{i \in R_d} |y_i^* - x_i^*|$$

Ls is the loss for the probability map
Lb is the loss for the binary map

$$L_s = L_b = \sum_{i \in S_l} y_i \log x_i + (1 - y_i) \log (1 - x_i)$$

6.3. EAST

Main Idea

EAST is a fast and accurate scene text detection method and consists of two stages:

1. It uses a complete convolutional network (FCN) model to directly generate pixel-based word or text line predictions

2. After generating text predictions (Rotate a rectangle or quad) and the output is sent to the non-maximum suppression to produce the final result.

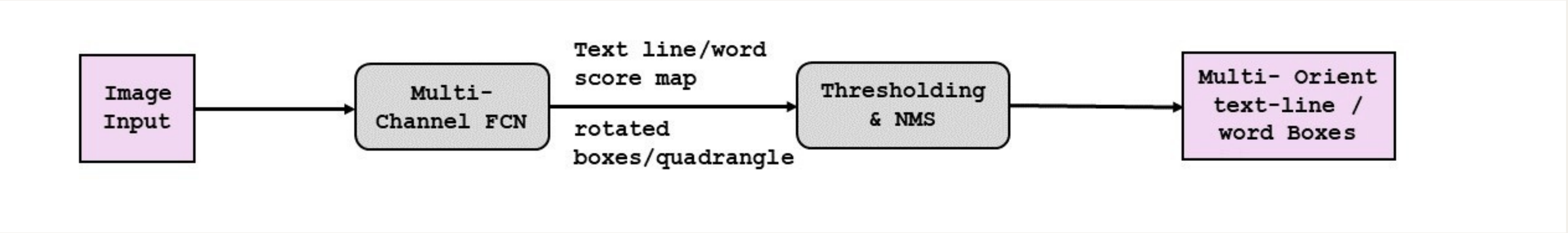
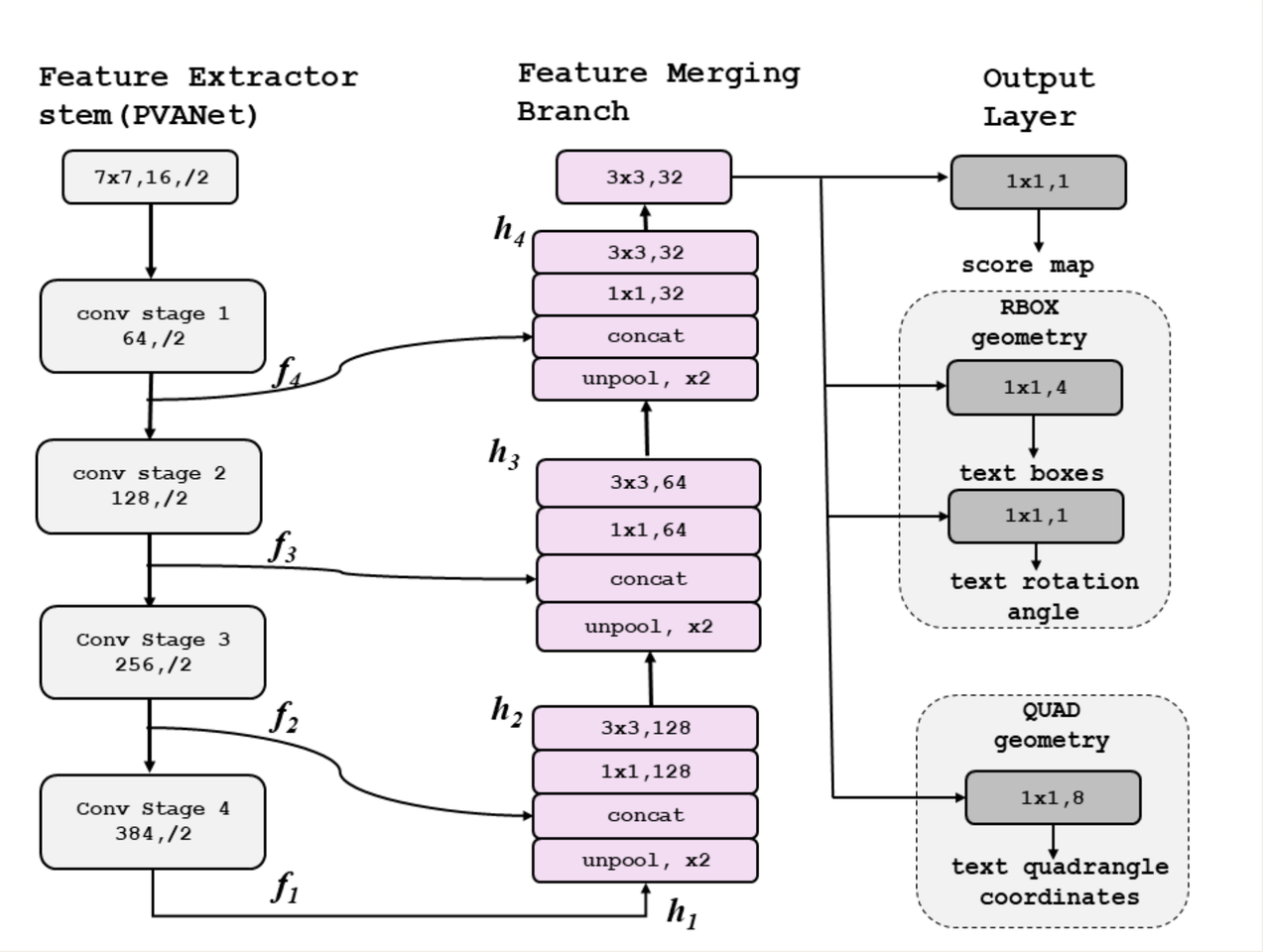


Figure 1: Pipeline

Network Architecture

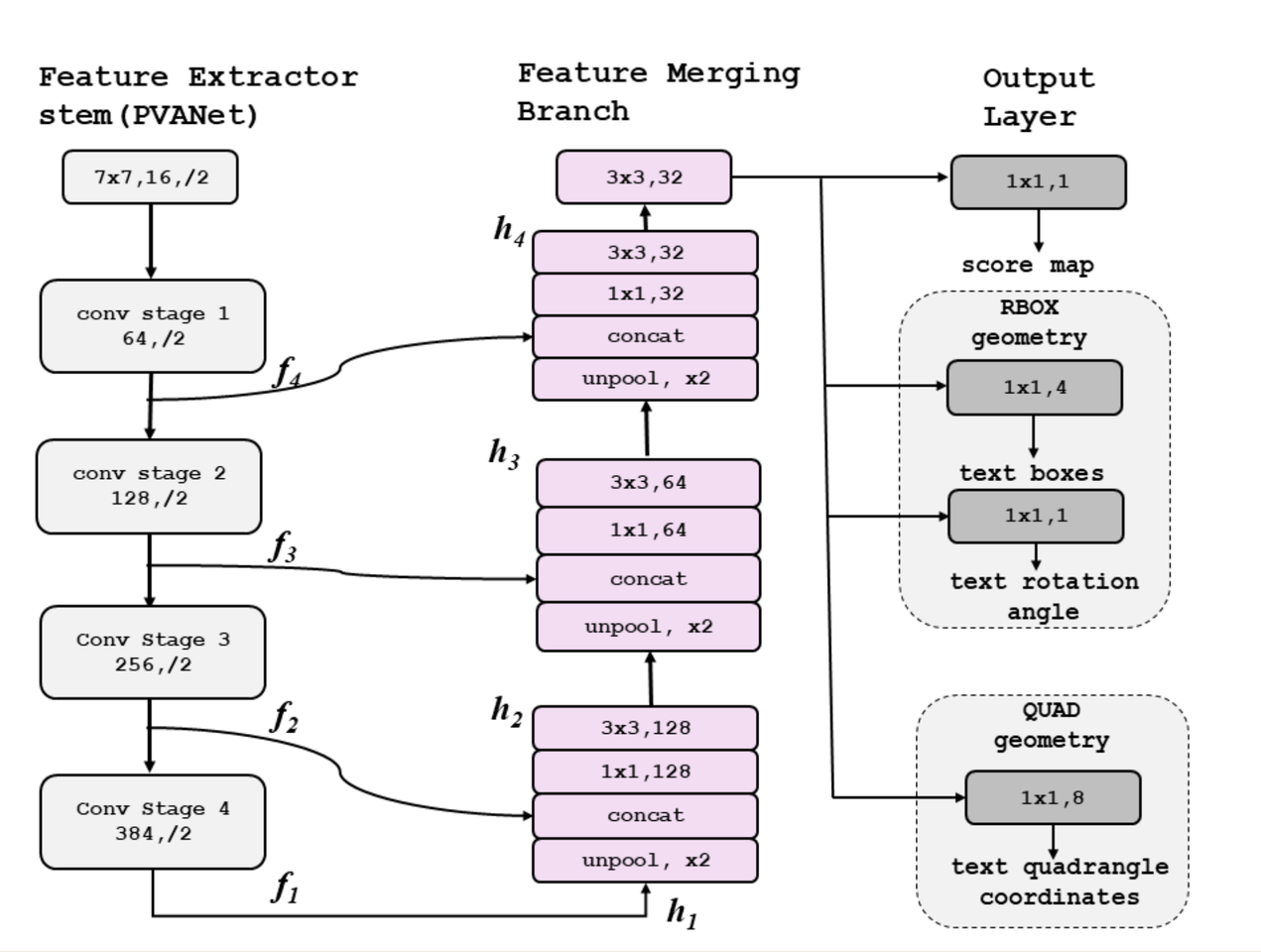
The model can be divided into three parts:

- Stem feature extraction
- Feature merging branches
- Output layer.



Feature Extractor

This part can be any convolutional neural network with convolutional layer and pooled layer interleaving pre-trained on Imagenet data for examples PVANet, VGG16 and RESNET50.



Feature Merging Branch

$$g_i = \begin{cases} \text{unpool}(h_i) & \text{if} \quad i \leq 3 \\ \text{conv}_{3\times 3}(h_i) & \text{if} \quad i = 4 \end{cases}$$

$$h_i = \begin{cases} f_i & \text{if } i = 1 \\ \text{conv}_{3\times 3}(\text{conv}_{1\times 1}([g_{i-1}; f_i])) & \text{otherwise} \end{cases}$$

where,

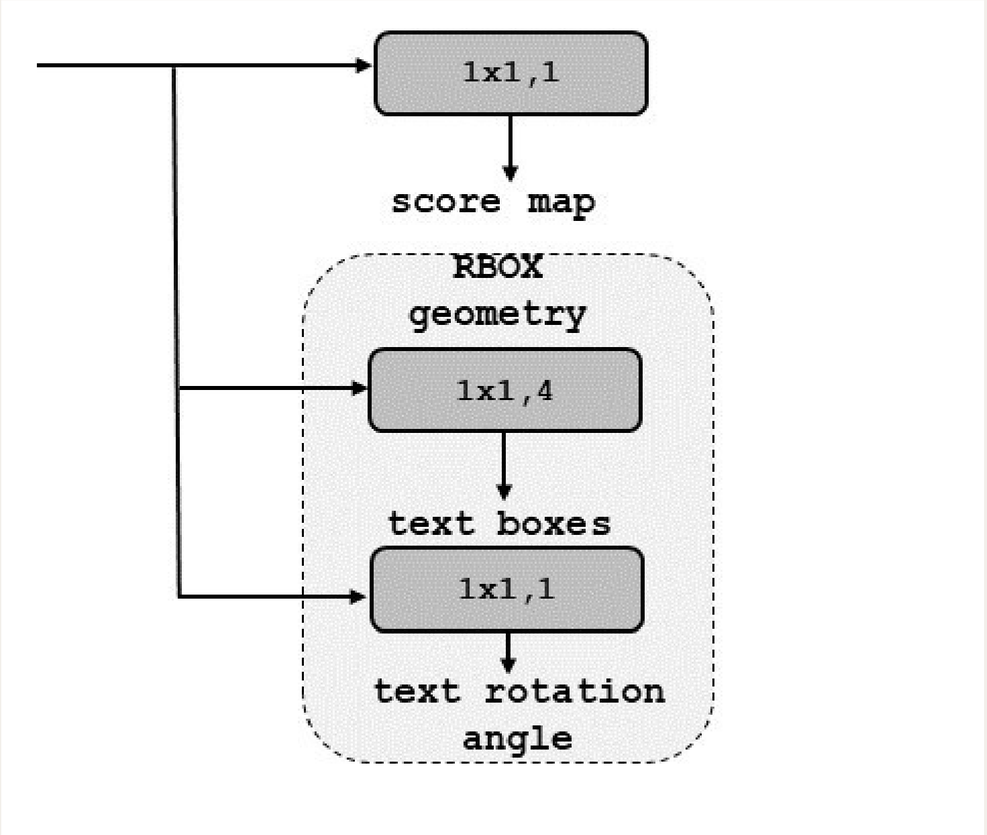
gi - an intermediate state and the basis of merging

h*i* - the merged feature map

RBOX

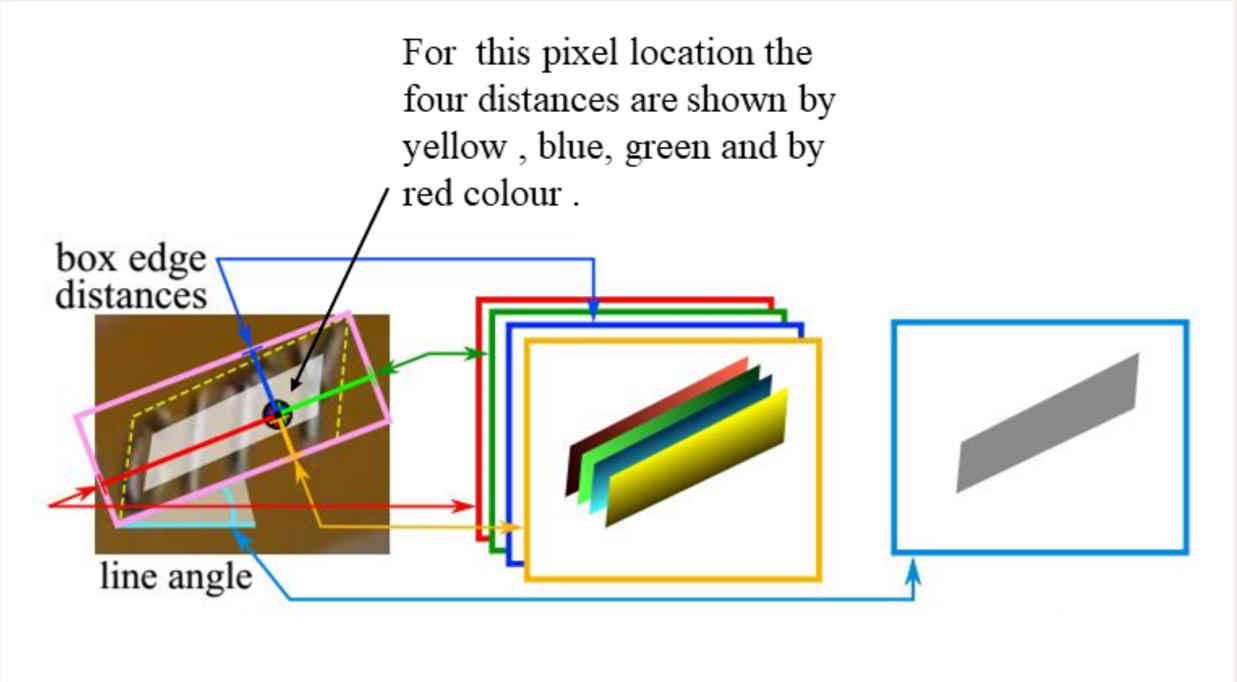
For RBOX, the geometry uses a four-channel axis-aligned bounding box (AABB) R and a channel rotation angle θ.

The formula of R is G.



RBOX

The four channels represent 4 distances, which are the distance from the pixel position to the rectangular boundaries and one channel for rotation angle



Loss Function

The loss can be formulated as

$$L = L_{\mathrm{s}} + \lambda_{\mathrm{g}} L_{\mathrm{g}}$$

where L_{s} and L_{g} represents the losses for the score map and the geometry, respectively, λ_{g} weighs the importance between two losses. In our experiment, we set λ_{g} to 1 .

Result

	F1-score	Precision	Recall
PSE	0.847	0.92	0.786
DBNet	0.931	0.922	0.926
EAST	0.84	0.814	0.868

Thanks for listening