Efficient and accurate arbitrary-shaped text detection with pixel aggregation network (PAN)

ICCV 2019

Kim Bochan

2020. 07. 19 Mon

Scene Text Detection

Figure 8. Detection results on CTW1500.

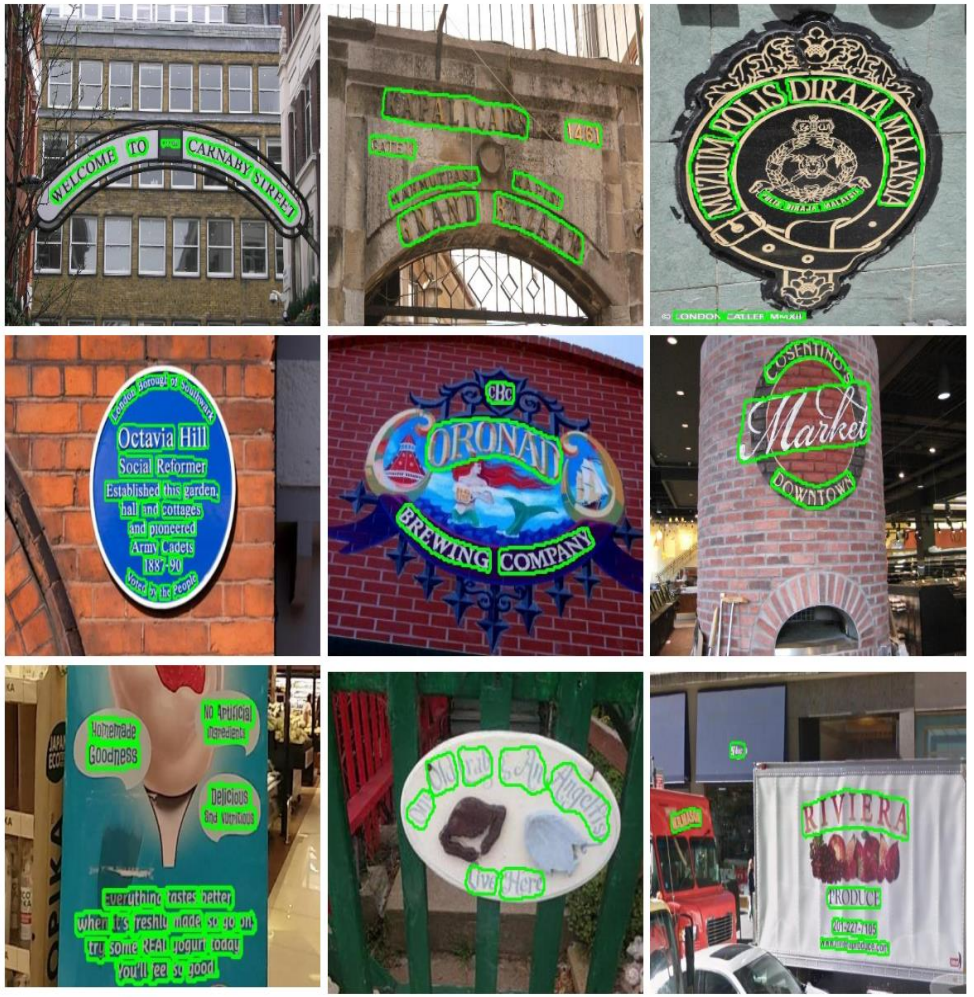


Figure 9. Detection results on Total-Text.

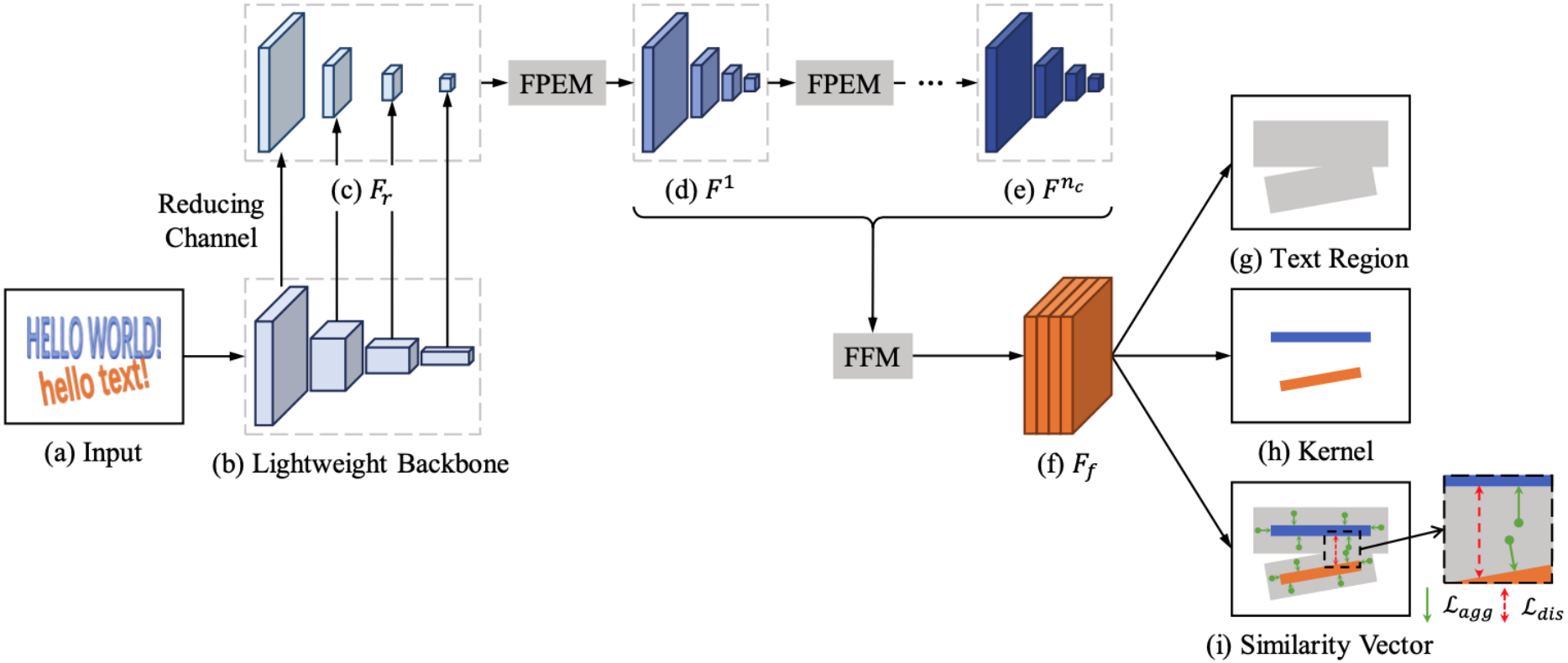
FPS

Method	Ext	R	P	F	FPS
SegLink* [28]	-	23.8	30.3	26.7	-
EAST* [49]	-	36.2	50.0	42.0	-
Lyu *et al.*[24]	✓	55.0	69.0	61.3	-
TextSnake [23]	✓	74.5	82.7	78.4	-
MSR [43]	✓	74.8	83.8	79.0	4.3
PSENet [33]	-	75.1	81.8	78.3	3.9
PSENet [33]	✓	78.0	84.0	80.9	3.9
Wang *et al.*[35]	-	76.2	80.9	78.5	-
TextDragon [6]	✓	74.2	84.5	79.0	-
TextField [41]	✓	79.9	81.2	80.6	6
PAN [34]	-	79.4	88.0	83.5	**39.6**
LOMO [46]	✓	75.7	**88.6**	81.6	4.4
LOMO† [46]	✓	79.3	87.6	83.3	-
CRAFT [2]	✓	79.9	87.6	83.6	-
Ours	-	**83.9**	86.9	**85.4**	3.8

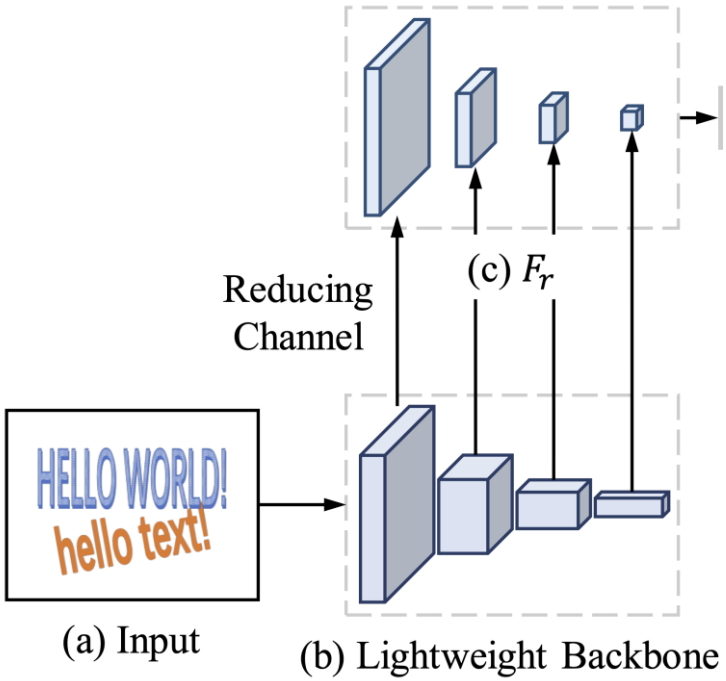
Method	Ext	R	P	F	FPS
EAST [49]	✓	73.5	83.6	78.2	13.2
Liao et al. [15]	✓	79.0	85.6	82.2	6.5
Lyu et al. [25]	✓	70.7	**94.1**	80.7	3.6
FOTS [20]	✓	82.0	88.8	85.3	7.8
PixelLink [4]	-	81.7	82.9	82.3	7.3
MSR [43]	✓	78.4	86.6	82.3	4.3
PSENet [33]	-	79.7	81.5	80.6	1.6
PSENet [33]	✓	84.5	86.9	85.7	1.6
PAN [34]	-	77.8	82.9	80.3	**26.1**
TextDragon [6]	✓	81.8	84.8	83.1	-
LOMO [46]	✓	83.5	91.3	87.2	3.4
TextField* [41]	✓	83.9	84.3	84.1	1.8
Liu *et al.* [21]	✓	83.8	89.4	86.5	-
Tian *et al.* [32]	✓	85.0	88.3	86.6	3
CRAFT [2]	✓	84.3	89.8	86.9	-
Wang *et al.* [35]	-	83.3	90.4	86.8	-
Wang *et al.*† [35]	-	86.0	89.2	**87.6**	-
Ours	-	**86.1**	87.6	86.9	3.5

Architecture

(Light)Backbone +
Feature Pyramid Enhancement Module(FPEM) +
Feature Fusion Module(FFM) +
Pixel Aggregation(PA)



Architecture – backbone

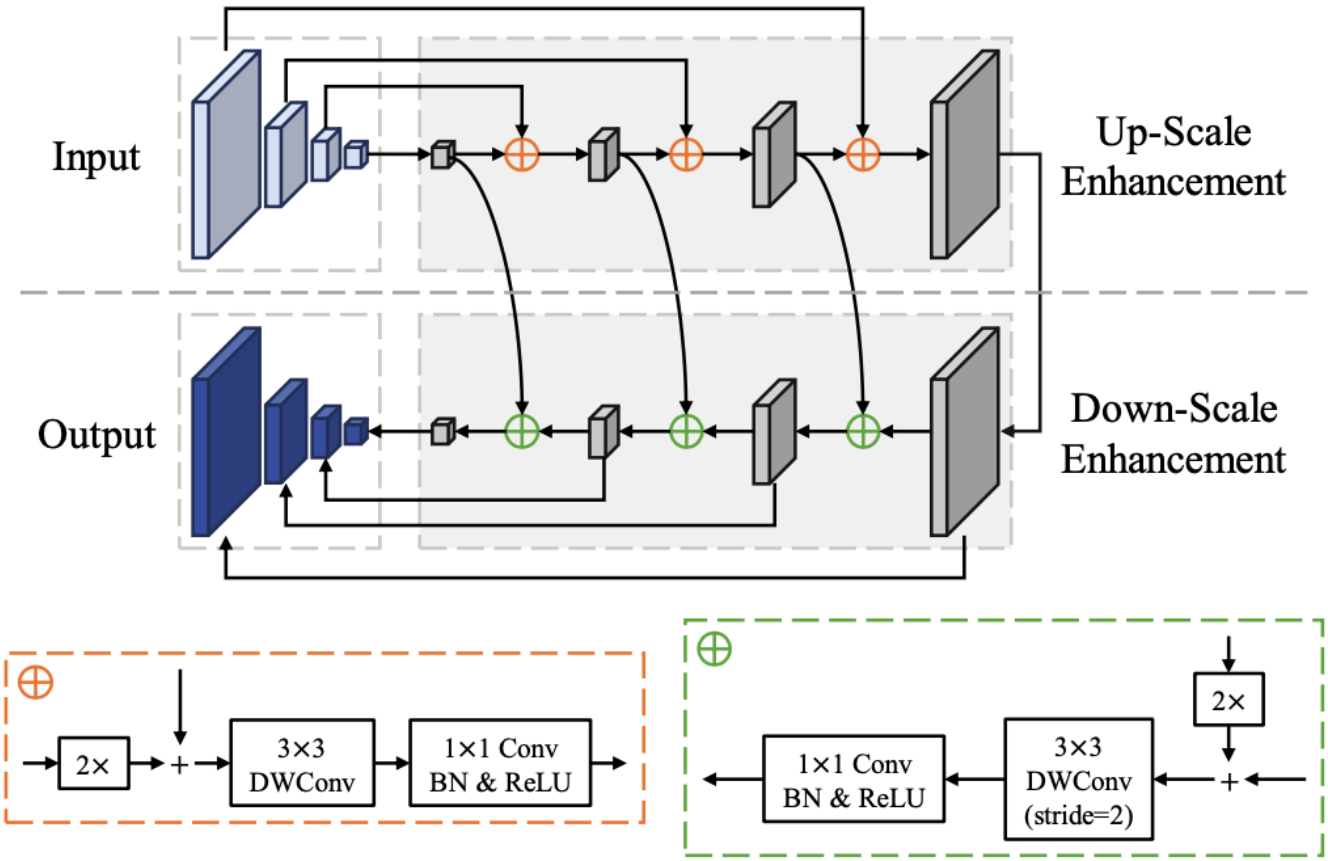


1. Resnet18

2. 4 feature maps generated by conv2, conv3, conv4, and conv5 stages of backbone (4, 8, 16, 32 strides)

3. 1×1 conv to reduce the channel to 128

Architecture – FPEM

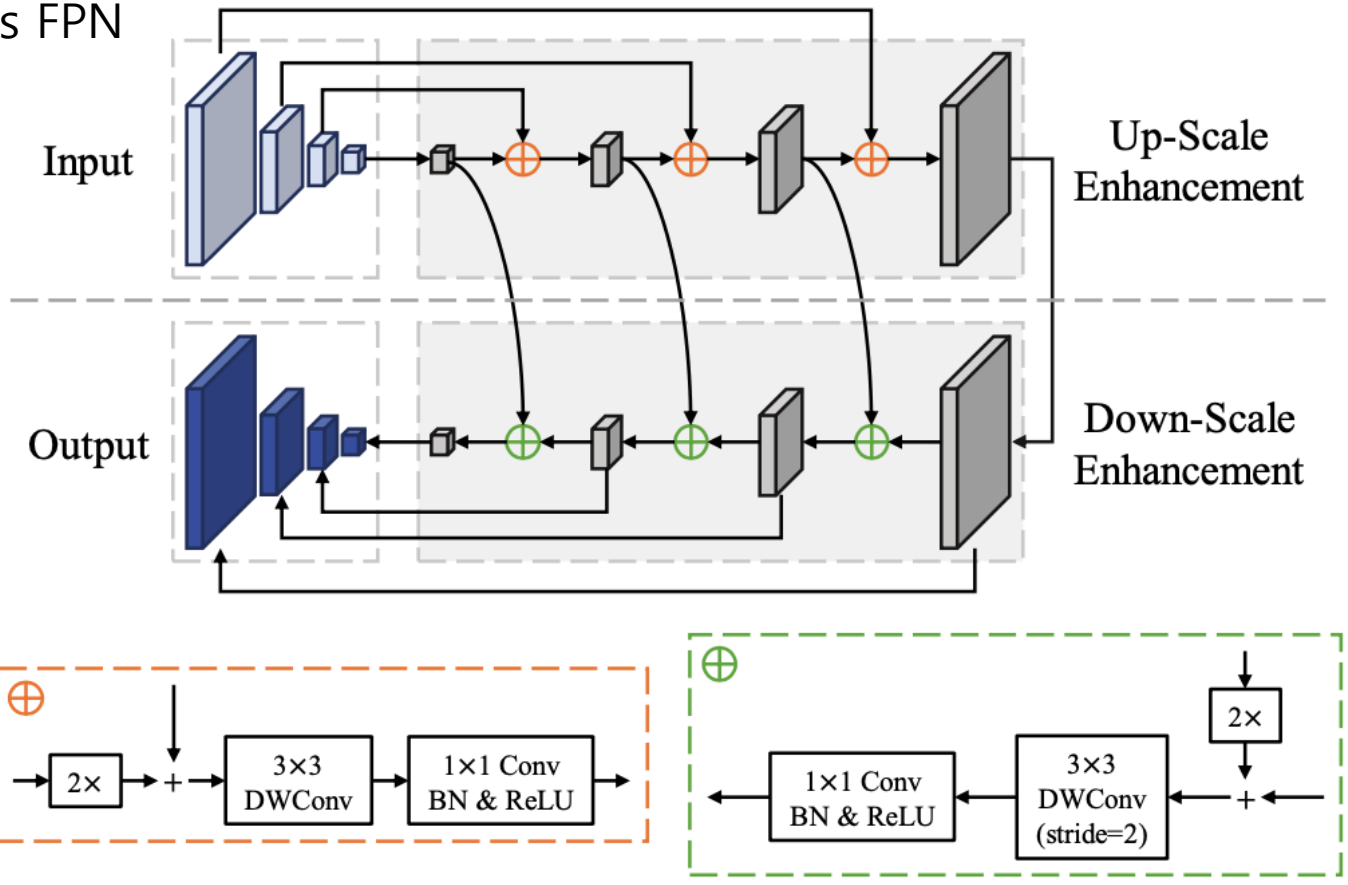


1. U-shaped with two phases, up-scale & down-scale enhancement
    1. enhancement is iteratively conducted on [4, 8, 16, 32] strides

2. Depthwise Separable Conv(DWConv)
    1. small computation overhead

Architecture - FPEM

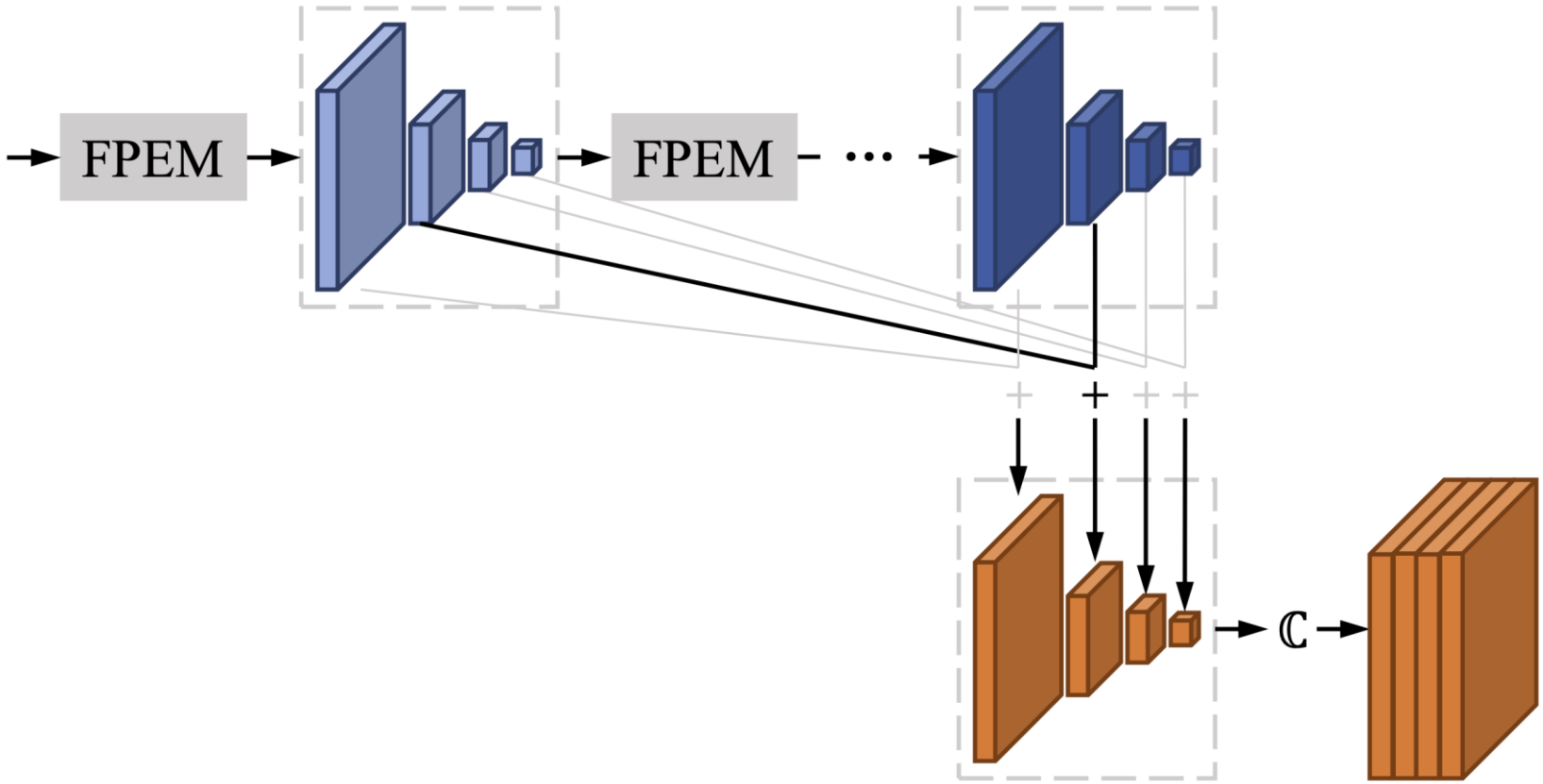
FPEM vs FPN



1. Fully Cascadable
    1. as n_c increased, feature map of different scales are fused
    2. receptive field more bigger

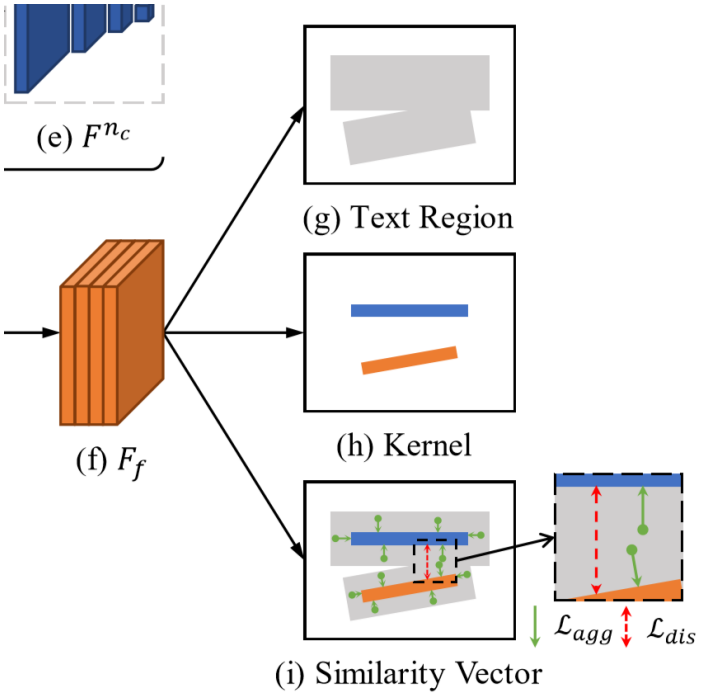
2. Computationally cheap
    1. 1/5 FLOPS of FPN

Architecture - FFM



1. Combine the corresponding-scale feature maps by element-wise addition
2. Feature maps after addition are upsampled and concatenated into a final feature map (4×128 channels)

Architecture - PA



Text regions lying closely are often overlapping
-> need to merge text region pixels to kernels

1. Distance between the **text pixel** and kernel of the same text instance should be small.
-> Aggregation Loss

2. **Kernels** of different text instances should maintain enough distance.
-> Discrimination Loss

Architecture - PA

1. Aggregation Loss

$$\mathcal{L}_{agg} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|T_i|}\sum_{p\in T_i} ln(\mathcal{D}(p, K_i) + 1)$$

$$\mathcal{D}(p, K_i) = max(\|\mathcal{F}(p) - \mathcal{G}(K_i)\| - \delta_{agg}, 0)^2$$ (δagg = 0.5)

N: number of text instances

T : text instance

p: pixel

K: kernel

F(p): similarity vector of pixel p

g(k): similarity vector of kernel k, calculated as $\sum_{q\in K_i}\mathcal{F}(q)/|K_i|$

Architecture – PA

2. Discrimination Loss

Try to keep the distance among the kernels not less than δdis (δdis = 3)

$$\mathcal{L}_{dis} = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} ln(\mathcal{D}(K_i, K_j) + 1),$$

$$\mathcal{D}(K_i, K_j) = max(\delta_{dis} - \|\mathcal{G}(K_i) - \mathcal{G}(K_j)\|, 0)^2.$$

Full Objective

$$\mathcal{L}_{tex} = 1 - \frac{2\sum_i P_{tex}(i) G_{tex}(i)}{\sum_i P_{tex}(i)^2 + \sum_i G_{tex}(i)^2},$$

$$\mathcal{L}_{ker} = 1 - \frac{2\sum_i P_{ker}(i) G_{ker}(i)}{\sum_i P_{ker}(i)^2 + \sum_i G_{ker}(i)^2},$$

$$\mathcal{L} = \mathcal{L}_{tex} + \alpha \mathcal{L}_{ker} + \beta(\mathcal{L}_{agg} + \mathcal{L}_{dis}),$$

1. Dice Loss
2. tex, ker : binary map
3. GT of kernel map is generated by shrinking original GT polygon
    1. methods followed by PSENet
4. Online Hard Example Mining(OHEM, ratio=3)
5. alpha=0.5, beta=0.25

Result Visualization

(a) Final Result (b) Predicted Text Region (c) Predicted Kernel (d) Predicted Similarity Vector

(e) CTW1500 (f) Total-Text (g) ICDAR 2015 (h) MSRA-TD500

1. Dimension of similarity vector is reduced to 3, 2 by PCA

Post Processing

(a) Final Result (b) Predicted Text Region (c) Predicted Kernel (d) Predicted Similarity Vector

(e) CTW1500 (f) Total-Text (g) ICDAR 2015 (h) MSRA-TD500

i) Finding the connected components in the kernels' segmentation result, and each connected component is a single kernel.

i) For each kernel K_i, merging its neighbor text pixel in predicted text regions while the Euclidean distance of their similarity vectors is less than d(=6).

i) Repeating step ii) until there is no eligible neighbor text pixel.

Training Details

1. Dimension of similarity vector = 4
2. SGD
3. Two training strategies
    1. From scratch(36K)
        1. lr = 10-3, "poly" lr strategy
    2. Pre-training on SynthText(50K) and finetuning(36K)
        1. lr = 10-3, weight decay = 5X10-4, Nesterov momentum = 0.99, He init
4. random scale, random horizontal flip, random rotation and random crop

Ablation Study - FPEM

Influence of the number of cascaded FPEMs

#FPEM	GFLOPS	ICDAR 2015		CTW1500	
		F	FPS	F	FPS
0	**42.17**	78.4	**33.7**	78.8	**49.7**
1	42.92	79.9	29.5	80.4	44.7
2	43.67	80.3	26.1	81.0	39.8
3	44.43	80.4	23.0	81.3	35.2
4	45.18	**80.5**	20.1	**81.5**	32.4

Table 1. The results of models with different number of cascaded FPEMs. "#FPEM" means the number of cascaded FPEMs. "F" means F-measure. The FLOPS are calculated for the input of $640 \times 640 \times 3$.

1. Satureaed after n_c = 2
2. Set default as n_c = 2

Ablation Study - FPEM

Effectiveness of FPEM

#FPEM	GFLOPS	ICDAR 2015		CTW1500	
		F	FPS	F	FPS
0	**42.17**	78.4	**33.7**	78.8	**49.7**
1	42.92	79.9	29.5	80.4	44.7
2	43.67	80.3	26.1	81.0	39.8
3	44.43	80.4	23.0	81.3	35.2
4	45.18	**80.5**	20.1	**81.5**	32.4

Method	ICDAR 2015		CTW1500	
	F	FPS	F	FPS
ResNet18 + 2 FPEMs + FFM	80.3	**26.1**	81.0	**39.8**
ResNet50 + PSPNet [56]	**80.5**	4.6	**81.1**	7.1

1. With FPEM, can make about 1.5% improvement on F-measure while bringing tiny extra computation
1. Can reach almost same performance with huge model
    1. 5 times faster
    2. Model size is only 12.25M

Ablation Study - FFM

Effectiveness of FFM

#	Backbone	Fuse	PA	ICDAR 2015		CTW1500	
				F	FPS	F	FPS
1	ResNet18	FFM	✓	80.3	26.1	81.0	39.8
2	ResNet18	-	✓	79.7	**26.2**	80.2	**40.0**
3	ResNet18	Concat	✓	80.4	22.3	81.2	35.9
4	ResNet18	FFM	-	79.3	26.1	79.8	39.9
5	ResNet50	FFM	✓	81.4	16.7	**81.6**	26.0
6	VGG16	FFM	✓	**81.9**	6.6	81.5	10.1

1. F-measure drop 0.6%-0.8% when the FFM is removed
2. Proposed FFM can achieve performance comparable to the direct concatenation

Ablation Study - PA

Effectiveness of PA

#	Backbone	Fuse	PA	ICDAR 2015		CTW1500	
				F	FPS	F	FPS
1	ResNet18	FFM	✓	80.3	26.1	81.0	39.8
2	ResNet18	-	✓	79.7	**26.2**	80.2	**40.0**
3	ResNet18	Concat	✓	80.4	22.3	81.2	35.9
4	ResNet18	FFM	-	79.3	26.1	79.8	39.9
5	ResNet50	FFM	✓	81.4	16.7	**81.6**	26.0
6	VGG16	FFM	✓	**81.9**	6.6	81.5	10.1

1. set β to 0
2. F-measure of the model without PA (Table #4) drops over 1%

Ablation Study - backbone

Effectiveness of backbone

#	Backbone	Fuse	PA	ICDAR 2015		CTW1500	
				F	FPS	F	FPS
1	ResNet18	FFM	✓	80.3	26.1	81.0	39.8
2	ResNet18	-	✓	79.7	**26.2**	80.2	**40.0**
3	ResNet18	Concat	✓	80.4	22.3	81.2	35.9
4	ResNet18	FFM	-	79.3	26.1	79.8	39.9
5	ResNet50	FFM	✓	81.4	16.7	**81.6**	26.0
6	VGG16	FFM	✓	**81.9**	6.6	81.5	10.1

1. ResNet50 and VGG16 can bring over 1% improvement on ICDAR 2015 and over 0.5% improvement on CTW1500.

2. The reduction of FPS brought by the heavy backbone is apparent

Results

CTW 1500 – Curved TD (Polygon)

Method	Ext.	Venue	CTW1500			
			P	R	F	FPS
CTPN* [47]	-	ECCV'16	60.4*	53.8*	56.9*	7.14
SegLink* [42]	-	CVPR'17	42.3*	40.0*	40.8*	10.7
EAST* [58]	-	CVPR'17	78.7*	49.1*	60.4*	21.2
CTD+TLOC [31]	-	ICDAR'18	77.4	69.8	73.4	13.3
PSENet-1s [24]	-	CVPR'19	80.6	75.6	78.0	3.9
PAN-320	-	-	82.2	72.6	77.1	**84.2**
PAN-512	-	-	83.8	77.1	80.3	58.1
PAN-640	-	-	84.6	77.7	81.0	39.8
TextSnake [35]	✓	ECCV'18	67.9	85.3	75.6	-
PSENet-1s [24]	✓	CVPR'19	84.8	79.7	82.2	3.9
PAN-320	✓	-	82.7	77.4	79.9	**84.2**
PAN-512	✓	-	85.5	**81.5**	83.5	58.1
PAN-640	✓	-	**86.4**	81.2	**83.7**	39.8

Results

Total Text – Curved TD (Polygon)

Method	Ext.	Venue	Total-Text			
			P	R	F	FPS
SegLink* [42]		CVPR'17	30.3*	23.8*	26.7*	-
EAST* [58]	-	CVPR'17	50.0*	36.2*	42.0*	-
DeconvNet [2]	-	ICDAR'18	33.0	40.0	36.0	-
PSENet-1s [24]	-	CVPR'19	81.8	75.1	78.3	3.9
PAN-320	-	-	84.0	71.3	77.1	**82.4**
PAN-512	-	-	86.7	78.4	82.4	57.1
PAN-640	-	-	88.0	79.4	83.5	39.6
TextSnake [35]	✓	ECCV'18	82.7	74.5	78.4	-
PSENet-1s [24]	✓	CVPR'19	84.0	78.0	80.9	3.9
SPCNet [50]	✓	AAAI'19	83.0	82.8	82.9	-
PAN-320	✓	-	85.6	75.0	79.9	**82.4**
PAN-512	✓	-	**89.4**	79.7	84.3	57.1
PAN-640	✓	-	89.3	**81.0**	**85.0**	39.6

Results

ICDAR 2015 – Arbitrary oriented dataset(most commonly used dataset)

Method	Ext.	Venue	ICDAR 2015			
			P	R	F	FPS
CTPN [47]	-	ECCV'16	74.2	51.6	60.9	7.1
EAST [58]	-	CVPR'17	83.6	73.5	78.2	13.2
RRPN [38]	-	TMM'18	82.0	73.0	77.0	-
DeepReg [17]	-	ICCV'17	82.0	80.0	81.0	-
PixelLink [3]	-	AAAI'18	82.9	81.7	82.3	7.3
PAN	-	-	82.9	77.8	80.3	**26.1**
SegLink [42]	✓	CVPR'17	73.1	76.8	75.0	-
SSTD [16]	✓	ICCV'17	80.2	73.9	76.9	7.7
WordSup [19]	✓	CVPR'17	79.3	77.0	78.2	-
Lyu et al. [37]	✓	CVPR'18	**94.1**	70.7	80.7	3.6
RRD [28]	✓	CVPR'18	85.6	79.0	82.2	6.5
MCN [32]	✓	CVPR'18	72.0	80.0	76.0	-
TextSnake [35]	✓	ECCV'18	84.9	80.4	82.6	1.1
PSENet-1s [24]	✓	CVPR'19	86.9	84.5	85.7	1.6
SPCNet [50]	✓	AAAI'19	88.7	**85.8**	**87.2**	-
PAN	✓	-	84.0	81.9	82.9	**26.1**

Results

MSRA-TD500 - Long straight TD (Quadrangle)

Method	Ext.	Venue	MSRA-TD500			
			P	R	F	FPS
EAST [58]	-	CVPR'17	87.3	67.4	76.1	13.2
RRPN [38]	-	TMM'18	82.0	68.0	74.0	-
DeepReg [17]	-	ICCV'17	77.0	70.0	74.0	1.1
PAN	-	-	80.7	77.3	78.9	**30.2**
SegLink [42]	✓	CVPR'17	86.0	70.0	77.0	8.9
PixelLink [3]	✓	AAAI'18	83.0	73.2	77.8	3.0
Lyu et al. [37]	✓	CVPR'18	87.6	76.2	81.5	5.7
RRD [28]	✓	CVPR'18	87.0	73.0	79.0	10
MCN [32]	✓	CVPR'18	**88.0**	79.0	83.0	-
TextSnake [35]	✓	ECCV'18	83.2	73.9	78.3	1.1
PAN	✓	-	84.4	**83.8**	**84.1**	**30.2**

Results

Time consumption comparison

Method	F	Time consumption (ms)			FPS
		Backbone	Head	Post	
PAN-320	77.10	**4.4**	**5.4**	**2.1**	**84.2**
PAN-512	80.32	6.4	7.3	3.5	58.1
PAN-640	**81.00**	9.8	10.1	5.2	39.8

1. Time costs of backbone and segmentation head are similar, and the time cost of post-processing is half of them.

1. Measured on 1 1080Ti GPU and 2.20GHz GPU with PyTorch

Results - robustness analysis

Cross-dataset results on word-level and line-level datasets

Annotation	Train Set→Test Set	P	R	F
Word	SynthText → ICDAR 2015	65.9	46.9	54.8
	SynthText → Total-Text	69.1	40.8	51.3
	ICDAR 2015 → Total-Text	72.0	57.8	64.1
	Total-Text → ICDAR 2015	77.6	65.5	71.1
Text Line	CTW1500 → MSRA-TD500	76.6	73.1	74.8
	MSRA-TD500 → CTW1500	82.4	69.1	75.2

1. still competitive

Results - robustness analysis

Results comparison on CTW1500 with other (high efficiency) segmentation methods

Methods	Ext.	F (%)	FPS
BiSeNet (ResNet18) [54]	-	78.8	25.9
CU-Net-2 (m=128, n=32) [46]	-	76.4	39.3
Ours (ResNet18 + 2FPEM + FFM)	-	**81.0**	**39.8**

1. They make prediction on 1/8 feature map

2. Our method enjoys obviously better accuracy (+2.2% and +4.6%) at the similar speed

개인적 소감

1. 속도 하나는 독보적이다

2. 속도를 중점적으로 focus하면서 성능도 최대한 지킬려고 노력한 느낌
   - 아키텍쳐 설계도 합리적이고 속도에 비해 성능도 나쁘지 않음

3. Training method(PA) 는 좀 애매한 느낌 (robust 할 수 있을까?)
   1) 가령 한 이미지에 text instance 가 엄청 많으면 4d-similarity vector 로 다 표현하게 학습할 수 있을까?
   2) text box를 binary segmentation 하는게 맞는건지 잘 모르겠음
   3) kernel을 정의하기 애매한 text나 이미지들도 wild 에선 존재할 듯

-> 속도는 빠르고 그 속도는 가성비 좋은 아키텍쳐 디자인에서 온 것 같으니 PA는 빼고 아키텍쳐만 한번 갖다 써볼까..

Q & A

감사합니다.