Mask R-CNN[1]

- 扩展方向\未读完\想法

  1、Instance seg; human pose (key points detection); bounding box;

  2、coco 比赛主页

  3、代码即将开源，而第三方代码尚未完善。

  4、为什么不用显著物体切分，这样会感觉更加节省内存

  效果略差一丢丢

  5、要看文章 12 来理解 mask loss

  6、查看文章 27 中 FPN 的实现

  7、查看文章 19 和 40 中的网络

  8、难道生成的 mask 不需要和原图一样大小的吗

  需要

  9、Table 2b 解释中的 multinomial loss 指什么

  10、其实最关键的在于，借助 AlignPool，操作于 RoI 的操作，其实完全等价于在原图 crop 出 RoI，然后通过一个深层网络做 segmentation 的效果，这里所说的等价是指不像 RoIPool 那样，位置出现偏差。但还不完全正确，因为它依旧使用 bin，因此实际上依旧是不等价的

一、介绍

- 目标：Instance segmentation is challenging because it requires the correct detection of all objects in an image while precisely segmenting each instance.

  object detection: to classify individual objects and localize each using a bounding box.

  Semantic seg: to classify each pixel into a fixed set of categories without differentiating object isnstance.

- 创新点阐述

  1、A branch for predicting segmentation mask on each RoI in parallel with existing for classification and bounding box regression.

  2、Align Pooling

  3、Decouple mask and class prediction，切分和分类完全解耦

  4、承认 Mask-RCNN 只是 faster 的一种扩展，但如何正确的扩展是很有技巧的

  5、as fast as FCIS

二、相关工作

- 发展脉络

  早期工作 → Deep Mask (seg → Faster) → Dai et a (类似本文) → Li et al (FCIS) 硬伤明显，可参看图 5

三、Mask R-CNN

- 网络框架

  Faster: RPN → candidate object bounding box → RoI pooling → feature → 分类\回归框

  Mask:

  Mask Representation, km^2; per-pixel sigmoid; L_{mask} is only defined to the k-th mask.

  RoIPool: 7*7 bin with max or average pooling

  RoIAlign: use x/16 instead of [x/16];

  1、 draw the box on the feature map;

  2、 draw the bin

  3、 for each bin, calculate the bin feature using bilinear interpolation to compute the exact values of the input

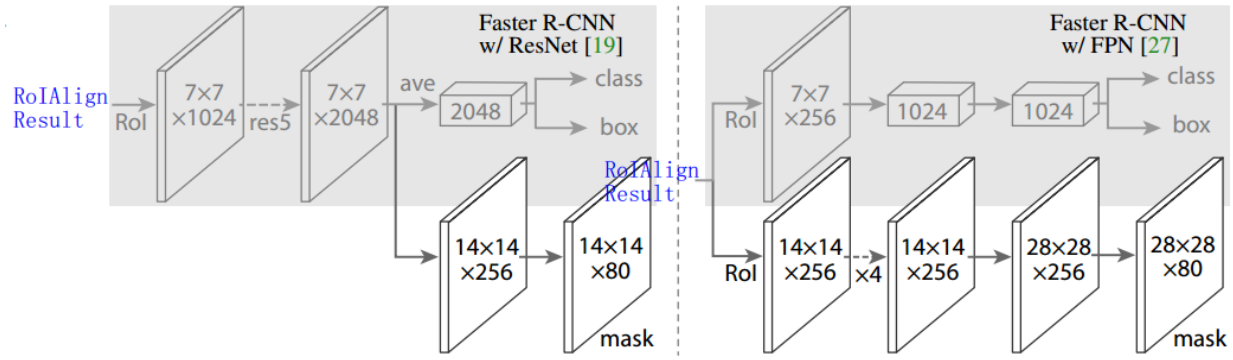
features at four regularly sampled locations.

Backbones:

ResNet; ResNeXt; network-depth-features

Feature Pyramid Network(FPN)

Network **head:**



四、实验

标准 AP averaged over IoU thresholds

$AP_{50}, AP_{75}, AP_S, AP_M, AP_L$ is AP at different scales.

数据集 COCO

Training: 80k train; 35k val(trainval35k);

tesing: 5k val(minival)

disclosed: 官方测试集

主要实验结果

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
COCO 2015 winner	MNC [10]	ResNet-101-C4	24.6	44.3	24.8	4.7	25.9	43.6
COCO 2016 winner	FCIS [26] +OHEM	ResNet-101-C5-dilated	29.2	49.5	-	7.1	31.3	50.0
	FCIS+++ [26] +OHEM	ResNet-101-C5-dilated	33.6	54.5	-	-	-	-
	Mask R-CNN	ResNet-101-C4	33.1	54.9	34.8	12.1	35.6	51.1
	Mask R-CNN	ResNet-101-FPN	35.7	58.0	37.8	15.5	38.1	52.4
	Mask R-CNN	ResNeXt-101-FPN	**37.1**	**60.0**	**39.4**	**16.9**	**39.9**	**53.5**

Table 1. **Instance segmentation** *mask* AP on COCO `test-dev`. MNC [10] and FCIS [26] are the winners of the COCO 2015 and 2016 segmentation challenges, respectively. Without bells and whistles, Mask R-CNN outperforms the more complex FCIS+++, which includes multi-scale train/test, horizontal flip test, and OHEM [35]. All entries are *single-model* results.

Ablation Experiments

net-depth-features	AP	AP_{50}	AP_{75}
ResNet-50-C4	30.3	51.2	31.5
ResNet-101-C4	32.7	54.2	34.3
ResNet-50-FPN	33.6	55.2	35.3
ResNet-101-FPN	35.4	57.3	37.5
ResNeXt-101-FPN	**36.7**	**59.5**	**38.9**

(a) **Backbone Architecture**: Better backbones bring expected gains: deeper networks do better, FPN outperforms C4 features, and ResNeXt improves on ResNet.

	AP	AP_{50}	AP_{75}
softmax	24.8	44.1	25.1
sigmoid	**30.3**	**51.2**	**31.5**
	+5.5	*+7.1*	*+6.4*

(b) **Multinomial *vs.* Independent Masks** (ResNet-50-C4): *Decoupling* via per-class binary masks (sigmoid) gives large gains over multinomial masks (softmax).

	align?	bilinear?	agg.	AP	AP_{50}	AP_{75}
RoIPool [12]			max	26.9	48.8	26.4
RoIWarp [10]		✓	max	27.2	49.2	27.1
		✓	ave	27.1	48.9	27.1
RoIAlign	✓	✓	max	**30.2**	**51.0**	**31.8**
	✓	✓	ave	**30.3**	**51.2**	**31.5**

(c) **RoIAlign** (ResNet-50-C4): Mask results with various RoI layers. Our RoIAlign layer improves AP by ~3 points and AP_{75} by ~5 points. Using proper alignment is the only factor that contributes to the large gap between RoI layers.

	AP	AP_{50}	AP_{75}	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}
RoIPool	23.6	46.5	21.6	28.2	52.7	26.9
RoIAlign	**30.9**	**51.8**	**32.1**	**34.0**	**55.3**	**36.4**
	+7.3	*+ 5.3*	*+10.5*	*+5.8*	*+2.6*	*+9.5*

(d) **RoIAlign** (ResNet-50-**C5**, *stride 32*): Mask-level and box-level AP using *large-stride* features. Misalignments are more severe than with stride-16 features (Table 2c), resulting in massive accuracy gaps.

	mask branch	AP	AP_{50}	AP_{75}
MLP	fc: 1024→1024→80·28^2	31.5	53.7	32.8
MLP	fc: 1024→1024→1024→80·28^2	31.5	54.0	32.6
FCN	conv: 256→256→256→256→256→80	**33.6**	**55.2**	**35.3**

(e) **Mask Branch** (ResNet-50-FPN): Fully convolutional networks (FCN) *vs.* multi-layer perceptrons (MLP, fully-connected) for mask prediction. FCNs improve results as they take advantage of explicitly encoding spatial layout.

Table 2. **Ablations** for Mask R-CNN. We train on `trainval35k`, test on `minival`, and report *mask* AP unless otherwise noted.

Detection Result

	backbone	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{bb}_{S}	AP^{bb}_{M}	AP^{bb}_{L}
Faster R-CNN+++ [19]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [27]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [21]	Inception-ResNet-v2 [37]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [36]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
Faster R-CNN, RoIAlign	ResNet-101-FPN	37.3	59.6	40.3	19.8	40.2	48.8
Mask R-CNN	ResNet-101-FPN	38.2	60.3	41.7	20.1	41.1	50.2
Mask R-CNN	ResNeXt-101-FPN	**39.8**	**62.3**	**43.4**	**22.1**	**43.2**	51.2

Table 3. **Object detection** *single-model* results (bounding box AP), *vs.* state-of-the-art on `test-dev`. Mask R-CNN using ResNet-101-FPN outperforms the base variants of all previous state-of-the-art models (the mask output is ignored in these experiments). The gains of Mask R-CNN over [27] come from using RoIAlign (+1.1 AP^{bb}), multitask training (+0.9 AP^{bb}), and ResNeXt-101 (+1.6 AP^{bb}).

Human Pose Estimation

the training target is an one-hot m*m binary mask where only a single pixel is labeled as foreground

	AP^{kp}	AP^{kp}_{50}	AP^{kp}_{75}	AP^{kp}_{M}	AP^{kp}_{L}
CMU-Pose+++ [6]	61.8	84.9	67.5	57.1	68.2
G-RMI [31]†	62.4	84.0	68.5	**59.1**	68.1
Mask R-CNN, keypoint-only	62.7	87.0	68.4	57.4	71.1
Mask R-CNN, keypoint & mask	**63.1**	**87.3**	**68.7**	57.8	**71.4**

Table 4. **Keypoint detection** AP on COCO `test-dev`. Ours (ResNet-50-FPN) is a single model that runs at 5 fps. CMU-Pose+++ [6] is the 2016 competition winner that uses multi-scale testing, post-processing with CPM [39], and filtering with an object detector, adding a cumulative ~5 points (clarified in personal communication). †: G-RMI was trained on COCO *plus* MPII [1] (25k images), using two models (Inception-ResNet-v2 + ResNet-101). As they use more data, this is *not* a direct comparison with Mask R-CNN.

	AP^{bb}_{person}	AP^{mask}_{person}	AP^{kp}
Faster R-CNN	52.5	-	-
Mask R-CNN, mask-only	**53.6**	**45.8**	-
Mask R-CNN, keypoint-only	50.7	-	64.2
Mask R-CNN, keypoint & mask	52.0	45.1	**64.7**

Table 5. **Multi-task learning** of box, mask, and keypoint about the *person* category, evaluated on `minival`. All entries are trained on the same data for fair comparisons. The backbone is ResNet-50-FPN. The entry with 64.2 AP on `minival` has 62.7 AP on `test-dev`. The entry with 64.7 AP on `minival` has 63.1 AP on `test-dev` (see Table 4).

	AP^{kp}	AP^{kp}_{50}	AP^{kp}_{75}	AP^{kp}_{M}	AP^{kp}_{L}
RoIPool	59.8	86.2	66.7	55.1	67.4
RoIAlign	**64.2**	**86.6**	**69.7**	**58.7**	**73.0**

Table 6. **RoIAlign *vs.* RoIPool** for keypoint detection on `minival`.

Cityscapes

	training data	AP [`val`]	AP	AP_{50}	person	rider	car	truck	bus	train	mcycle	bicycle
InstanceCut [23]	`fine` + `coarse`	15.8	13.0	27.9	10.0	8.0	23.7	14.0	19.5	15.2	9.3	4.7
DWT [4]	`fine`	19.8	15.6	30.0	15.1	11.7	32.9	17.1	20.4	15.0	7.9	4.9
SAIS [17]	`fine`	-	17.4	36.7	14.6	12.9	35.7	16.0	23.2	19.0	10.3	7.8
DIN [3]	`fine` + `coarse`	-	20.0	38.8	16.5	16.7	25.7	20.6	30.0	23.4	17.1	10.1
Mask R-CNN	`fine`	31.5	26.2	49.9	30.5	23.7	46.9	22.8	32.2	18.6	19.1	16.0
Mask R-CNN	`fine` + COCO	**36.4**	**32.0**	**58.1**	**34.8**	**27.0**	**49.1**	**30.1**	**40.9**	**30.9**	**24.1**	**18.7**

Table 7. Results on Cityscapes `val` ('AP [`val`]' column) and `test` (remaining columns) sets. Our method uses ResNet-50-FPN.

Fully Convolutional Instance-aware Semantic Segmentation FCIS[2]

- 扩展方向\未读完\想法

1、为什么不直接根据类别来进行分割→如果某个类别有多个实例呢？→那就必须一个个按顺序来

2、8-gpu training

3、代码中的 libs 的由来：faster r-cnn tensorflow 版本

- **创新点：**

  1、 Position-sensitive inside/outside score maps

  2、a novel joint formulation with no extra parameters to share the convolution representation and score maps for objection detection and segmentation

  3、per-RoI computation is simple, fast, and does not involve and waring or resizing operations.

- **Motivation**

  1、 Because convolution is translation invariant, the same pixel receives the same responses irrespective to the same responses. certain translation-invariant is required to solve the problem.

  2、 ROI distortion and fixed-size representation degrades the segmentation accuracy, especially for large objects.

  3、 Where more than 80% of the time is spent on the last per-ROI step. (MNC)

- **Baseline [5]**

  translation invariant score maps → position-sensitive score maps (translation variant)

  1、 blind to semantic categories and require a downstream network for detection

  2、 the object segmentation and detection sub-tasks are separated and the solution is not end-to-end

  3、 operate on square, fixed-size sliding window (224*224)

  4、 time-consuming image pyramid scanning → instances at different scales

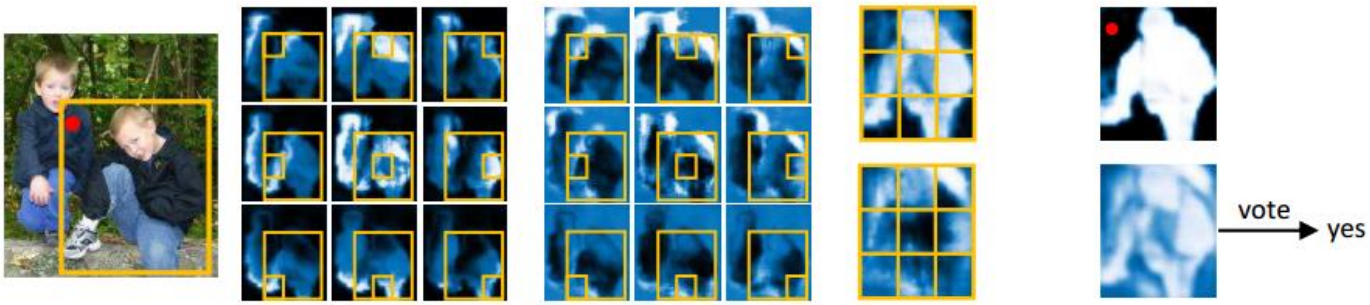
- **inside/outside score maps 的理解**

1、首先，可以看在训练的时候如何生成 target，观察下图，可以发现 inside maps 显示是使用 instance 之内的 9 个方位的标作为监督，而 outside maps 是使用 instance 之外，bbox 之内的 9 个方位进行监督。

2、更可怕的是，这些 score map 极有可能已经和原图的大小一致，因此 RPN 得到的框可以直接对这些 score map 进行操作

3、这些 feature map 其实是每一类生成一批的

4、RPN 得到的 bbox 有可能是跑完完整的 Faster 后得到的，如果只是 RPN 输出的，那会不会不够精细呢



- **使用到的各项技术的发展情况：**

1、Semantic segmentation

FCN → global context\multi-scale feature fusion\deconvolution→ with CRFs → with domain transform

2、Object segment proposal

MGG\Selective Search → DeepMask/SharpMask → [5]

3、instance-aware Semantic Segmentation

A segment proposal method + a classification task = SDS/CFM/MNC/MultiPathNet/iterative approach.

- 代码

My suggestion for this github :

Because mxnet and cuda is developing very rapidly. You need to update your cuda version to have this code run well. In my testing, only cuda 8 (0.64) is compatible with mxnet right now.

- 实验情况

1、Experiments on PASCAL VOL 2012

method	mAP^r@0.5 (%)	mAP^r@0.7 (%)
naïve MNC	59.1	36.0
InstFCN + R-FCN	62.7	41.5
FCIS (translation invariant)	52.5	38.5
FCIS (separate score maps)	63.9	49.7
FCIS	65.7	52.1

Table 1. Ablation study of (almost) fully convolutional methods on PASCAL VOC 2012 validation set.

2、Experiments on COCO

method	sampling strategy in training	train time/img	test time/img	mAP^r@[0.5:0.95] (%)	mAP^r@0.5 (%)	mAP^r@[0.5:0.95] (%) (small)	mAP^r@[0.5:0.95] (%) (mid)	mAP^r@[0.5:0.95] (%) (large)
MNC	random	2.05s	1.37s	24.6	44.3	4.7	25.9	43.6
FCIS	random	0.53s	0.24s	28.8	48.7	6.8	30.8	49.5
MNC	OHEM	3.22s	1.37s	N/A	N/A	N/A	N/A	N/A
FCIS	OHEM	0.54s	0.24s	**29.2**	**49.5**	**7.1**	**31.3**	**50.0**

Table 2. Comparison with MNC [8] on COCO test-dev set, using ResNet-101 model. Timing is evaluated on a Nvidia K40 GPU.

	mAP^r@[0.5:0.95] (%)	mAP^r@0.5 (%)
FAIRCNN (2015)	25.0	45.6
MNC+++ (2015)	28.4	51.6
G-RMI (2016)	33.8	56.9
FCIS baseline	29.2	49.5
+multi-scale testing	32.0	51.9
+horizontal flip	32.7	52.7
+multi-scale training	33.6	54.5
+ensemble	**37.6**	**59.9**

Table 4. Instance-aware semantic segmentation results of different entries for the COCO segmentation challenge (2015 and 2016) on COCO test-dev set.

Instance-sensitive Fully Convolutional Networks[3]

- 扩展方向\未读完\想法\理解

  1、 其实非常有意思，对 FCN 来说，预测图片中哪些像素属于人是非常容易的；但是要区别这些人就非常难了。但同样，要预测图片中哪些像素是属于一个物体的上半部分也是非常容易的，对 FCN 来说，这样一个任务并不难。（作者能将 object segmentation 拆分成这样一个可以由 FCN 进行处理的部分，确实有意思）

  2、 非常有意思，如果将 CNN+CTC 看成是 RPN propose 的结果的话，那么就相当完成了整个 pipline.

  3、 之所以不将整张图 crop 成一个个小块，然后分别做显著物体检测，就在于这其实是 RPN+显著物体检测的原始版本。

  4、 DeepMask 和 FCN 的不同点：each output pixel is a low-dimensional classifier?

- 相关工作

  其实 RPN 同样也是基于 FCN instance detection

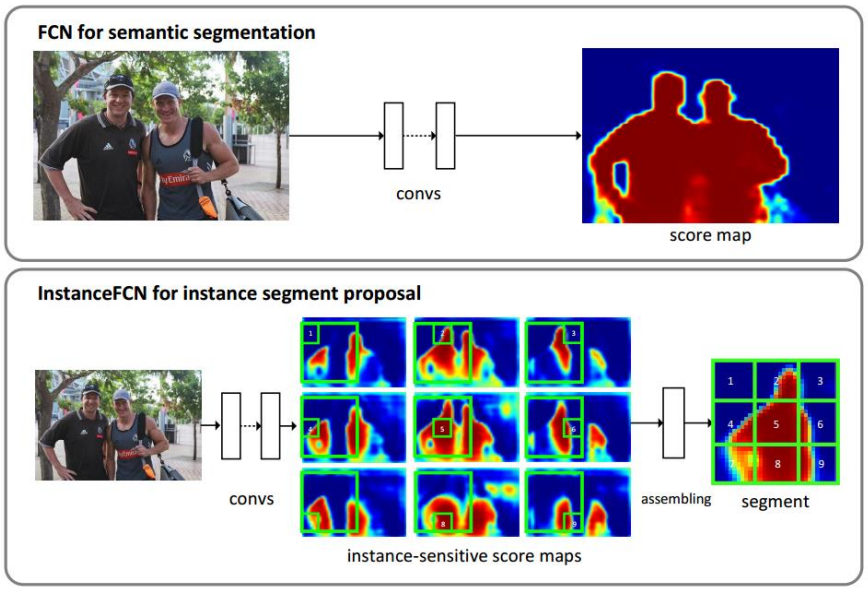


Figure 1. Methodological comparisons between: (**top**) FCN [1] for semantic segmentation; (**bottom**) our InstanceFCN for instance segment proposal.

- 主体框架

  1、Motivation: Although now the FCN output does not distinguish the two instances, we notice that the output is indeed reusable for most pixels, except for those where one object is conjunct the other——e.g., then the “right side” of the left instance is conjunct the “left side” of the right instance. If we can discriminate “right side” from “left side”, we can still rely on FCN-like score maps to generate instances.

  2、新概念：

  Instance-sensitive score maps

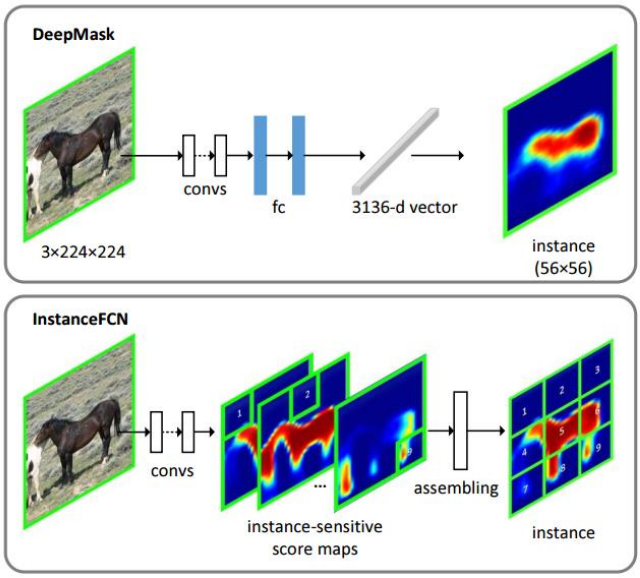
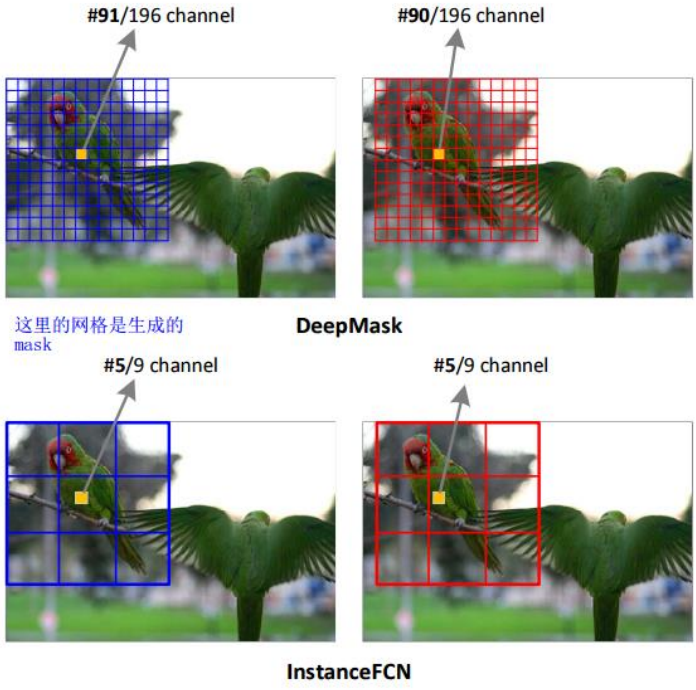


Figure 2. Methodological comparisons between DeepMask [8] and InstanceFCN for instance segment proposal. DeepMask uses a high-dimensional m^2-d *fc* layer to generate an instance, *e.g.*, $m = 56$ and $m^2 = 3136$. Our network has no any m^2-d layer.

Instance assembling module

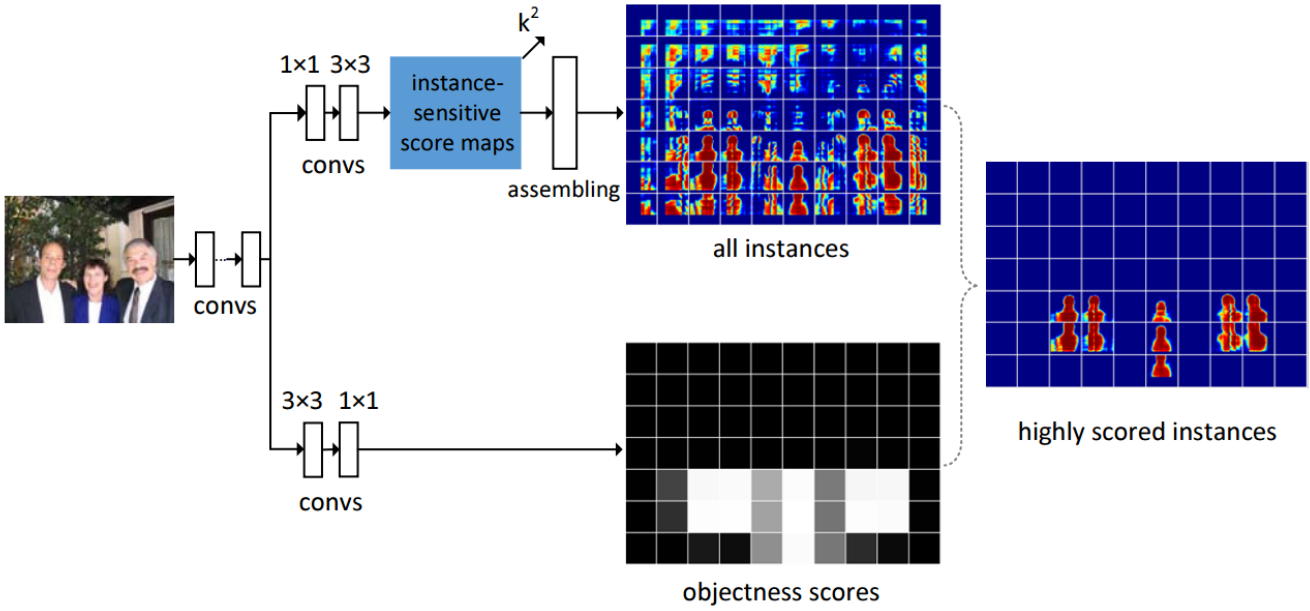
Local Coherence: the same pixel in the image coordinate system is predicted by two different channels of the fc layer. So the prediction of this pixel is in general not the same when evaluated in two neighboring windows.



3、Network

→ 1*1*512 → 3*3*512 → instance-sensitive score maps

→ 3*3*512 → 1*1*512 → scoring instance map



4、training

A set of 256 sliding windows are randoml sampled and the instances are only assembled from these 256 windows for computing the loss function

$$\sum_i (\mathcal{L}(p_i, p_i^*) + \sum_j \mathcal{L}(S_{i,j}, S_{i,j}^*)). \quad (1)$$

Here i is the index of a sampled window, p_i is the predicted objectness score of the instance in this window, and p_i^* is 1 if this window is a positive sample and 0 if a negative sample. S_i is the assembled segment instance in this window, S_i^* is the ground truth segment instance, and j is the pixel index in the window. $\mathcal{L}$ is the logistic regression loss. We use the definition of positive/negative samples

- 实验

Table 2. Ablation comparisons between ~DeepMask and our method on the PASCAL VOC 2012 validation set. "~DeepMask" is our implementation based on controlled settings (see more descriptions in the main text).

method	train	test	AR@10 (%)	AR@100 (%)	AR@1000 (%)
~DeepMask	crop 224×224	sliding *fc*	31.2	42.9	47.0
ours	crop 224×224	fully conv.	37.4	48.4	51.4
	fully conv.	fully conv.	**38.9**	**49.7**	**52.6**

Table 3. Comparisons with state-of-the-art segment proposal methods on the PASCAL VOC 2012 validation set. The results of SS [6] and MCG [12] are from the publicly available code, and the results of MNC [20] is provided by the authors of [20].

method	AR@10 (%)	AR@100 (%)	AR@1000 (%)
SS [6]	7.0	23.5	43.3
MCG [12]	18.9	36.8	49.5
~DeepMask	31.2	42.9	47.0
MNC [20]	<u>33.4</u>	<u>48.5</u>	**53.8**
ours	**38.9**	**49.7**	<u>52.6</u>

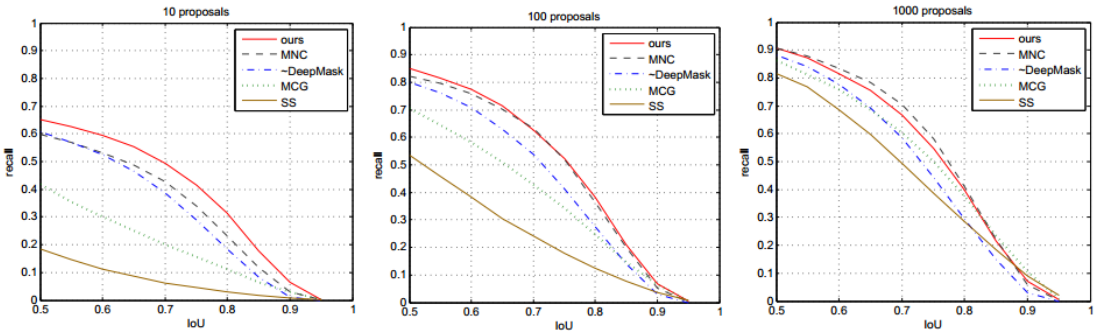


Figure 6. Recall *vs.* IoU curves of different segment proposals on the PASCAL VOC 2012 validation set. AR is the area under the curves.

Table 4. Semantic instance segmentation on the PASCAL VOC 2012 validation set. All methods are based on VGG-16 except SDS based on AlexNet [15].

downstream classifier	proposals	mAP@0.5 (%)	mAP@0.7 (%)
SDS [3]	MCG [7]	49.7	25.3
Hypercolumn [4]	MCG [7]	60.0	40.4
CFM [5]	MCG [7]	60.7	39.6
MNC [20]	MNC [20]	**63.5**	41.5
MNC [20]	**ours**	61.5	**43.0**

Learning to Segment Object Candidates[4]

- 扩展方向\未读完\想法\理解
  1、 难道是我的误解，测试时是 applied on the whole image
  2、 This is an Object proposal algorithm，应该跟目标检测不太一样，还是偏向于 object seg
  3、 训练时：image patch → 未知类别的切分 mask/拥有一个完整物体处在正中央的概率
  4、 Object proposal algorithm = maximum number of possible objects + minimum number of regions + regions mach the object
  5、 这个时候会不会 faster 还没出来；
  已经出来了，文章已经提及，可以查看作者如何讨论异同，确实存在差别，这个是生成 segmentation mask instead of bounding boxes.
  6、 公式(1)的 loss 函数的理解
  7、 如何理解显著物体检测；按照文章的说法，应该是通过类似 crop 或者 slidding window 的方式工作的，但这有点反直觉。
  可能的答案是 convolution 的时候是没法问题的，但 deconvolution 的时候就不能了。

8、其实最后一步就是一种没有重叠的反卷积

- 相关工作

1、fast R-CNN: selection of a set of salient object proposals → ConvNet classifier

2、object proposal approaches:

Object scoring; seed segmentation; super-pixel merging

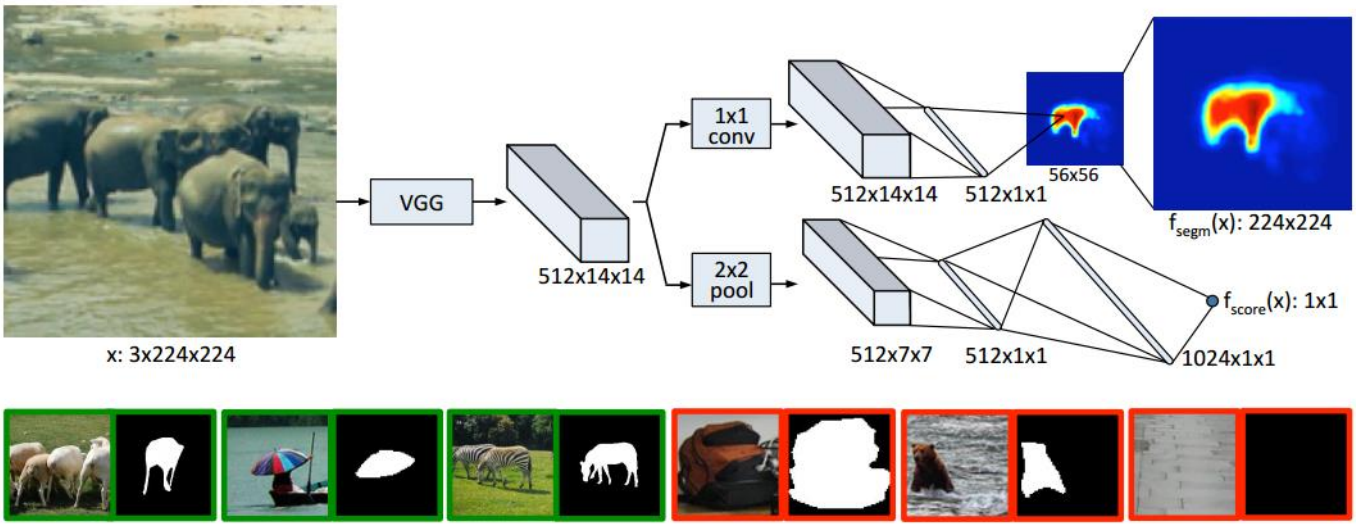


Figure 1: **(Top)** Model architecture: the network is split into two branches after the shared feature extraction layers. The top branch predicts a segmentation mask for the the object located at the center while the bottom branch predicts an object score for the input patch. **(Bottom)** Examples of training triplets: input patch x, mask m and label y. Green patches contain objects that satisfy the specified constraints and therefore are assigned the label $y = 1$. Note that masks for negative examples (shown in red) are not used and are shown for illustrative purposes only.

- 网络结构

1、 输入（x_k, m_k, y_k）x_k the RGB input patch; m_k binary mask; y_k specify whether the patch contains an object

Constrains: patch 正中央有一个物体；该物体完全被 path 包含；该物体不能太小；

2、 Given an input image of dimension 3*h*w, the output is a feature map of dimension $512 \times \frac{h}{16} \times \frac{w}{16}$.

3、 Note that each pixel in the output plane mush be able to utilize information contained in the entire feature map, and thus have a complete view of the object.

4、当输入一整张图片的时候，模型以 16 个像素滑窗扫描原图，同时原图还会做不同尺度的放缩（9 种尺度）

需要搞清楚，deconv 是如何处理的

3*244*244 → 512*14*14 →2*2max pooling →512fc → 1024 fc (dropout 0.5)

→ 1*1 conv 512 → 512*14*14 → 512*1*1 → 56*56 fc (no non-linearity)

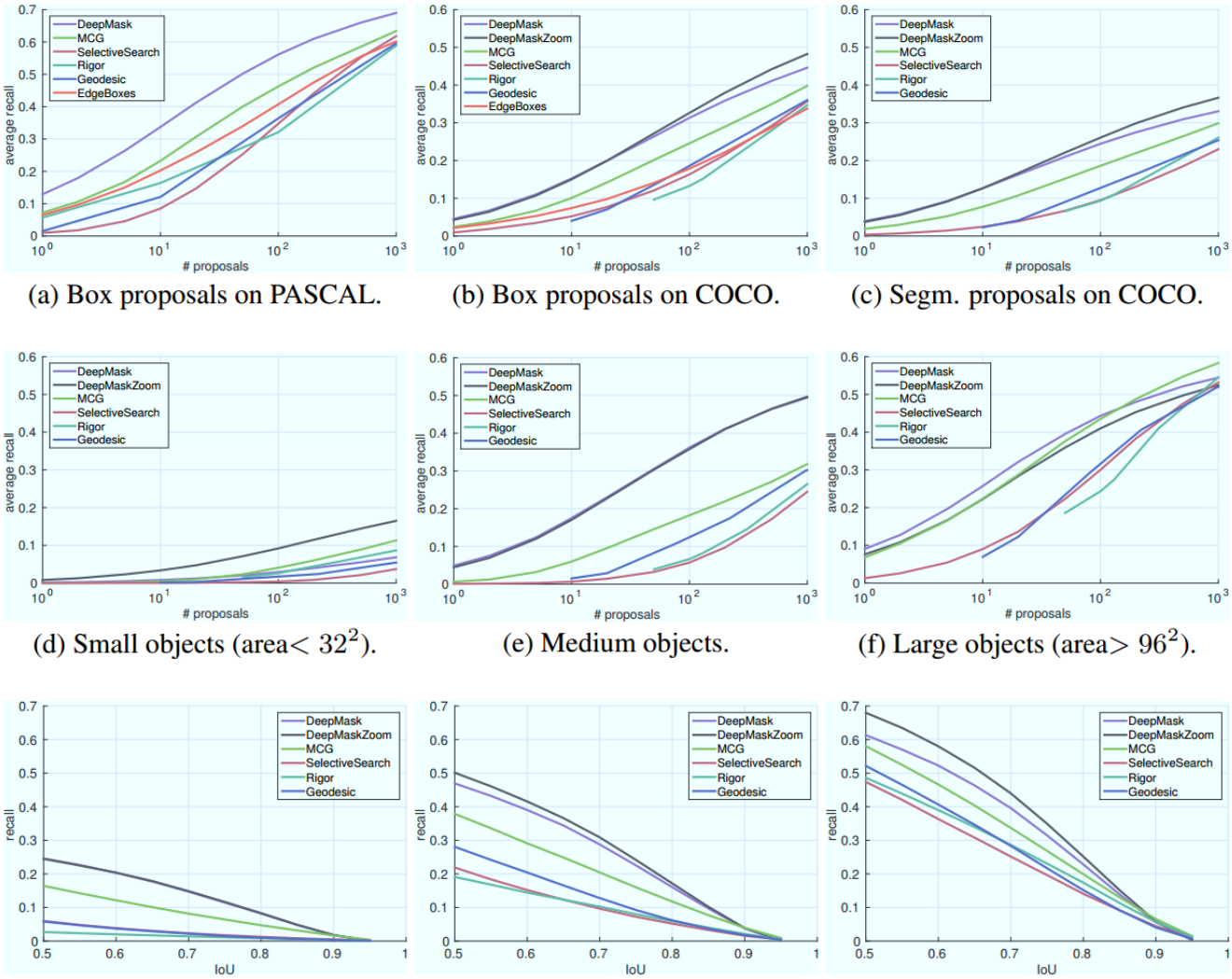
512*14*14 → 512*1*1 这一步其实还是不大明白，难道是为了适应各种不同的 size，故而使用的吗，是 max/average pooling 一类的？

- 实验

数据集 the PASCAL VOC 2007 test set & MS COCO validation data

on PASCAL (using boxes) and COCO (using both boxes and segmentations).

1.6s/image



(a) Box proposals on PASCAL. (b) Box proposals on COCO. (c) Segm. proposals on COCO.

(d) Small objects (area$< 32^2$). (e) Medium objects. (f) Large objects (area$> 96^2$).

(g) Recall with 10 proposals. (h) Recall with 100 proposals. (i) Recall with 1000 proposals.

Figure 4: (a-c) Average recall versus number of box and segmentation proposals on various datasets. (d-f) AR versus number of proposals for different object scales on segmentation proposals in COCO. (g-h) Recall versus IoU threshold for different number of segmentation proposals in COCO.

	Box Proposals				Segmentation Proposals						
	AR@10	AR@100	AR@1000	AUC	AR@10	AR@100	AR@1000	AUC^S	AUC^M	AUC^L	AUC
EdgeBoxes [34]	.074	.178	.338	.139	-	-	-	-	-	-	-
Geodesic [16]	.040	.180	.359	.126	.023	.123	.253	.013	.086	.205	.085
Rigor [14]	-	.133	.337	.101	-	.094	.253	.022	.060	.178	.074
SelectiveSearch [31]	.052	.163	.357	.126	.025	.095	.230	.006	.055	.214	.074
MCG [24]	.101	.246	.398	.180	.077	.186	.299	.031	.129	.324	.137
DeepMask20	.139	.286	.431	.217	.109	.215	.314	.020	.227	.317	.164
DeepMask20*	.152	.306	.432	.228	.123	.233	.314	.020	.257	.321	.175
DeepMaskZoom	.150	**.326**	**.482**	**.242**	**.127**	**.261**	**.366**	**.068**	**.263**	.308	**.194**
DeepMaskFull	.149	.310	.442	.231	.118	.235	.323	.020	.244	**.342**	.176
DeepMask	**.153**	.313	.446	.233	.126	.245	.331	.023	.266	.336	.183

Table 1: Results on the MS COCO dataset for both bounding box and segmentation proposals. We report AR at different number of proposals (10, 100 and 1000) and also AUC (AR averaged across all proposal counts). For segmentation proposals we report overall AUC and also AUC at different scales (small/medium/large objects indicated by superscripts S/M/L). See text for details.

PASCAL VOC07	AR@10	AR@100	AR@1000	AUC
EdgeBoxes [34]	.203	.407	.601	.309
Geodesic [16]	.121	.364	.596	.230
Rigor [14]	.164	.321	.589	.239
SelectiveSearch [31]	.085	.347	.618	.241
MCG [24]	.232	.462	.634	.344
DeepMask	**.337**	**.561**	**.690**	**.433**

Table 2: Results on PASCAL VOC 2007 test.

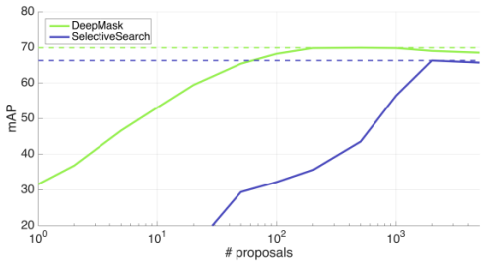


Figure 5: Fast R-CNN results on PASCAL.

R-FCN: Object Detection via Region-based Fully Convolutional Networks[5]

- 扩展方向\未读完\想法\理解

1、这篇文章是受 Instance-sensitive Fully Convolutional Networks 的启发，应该来说，这篇文章结合了 RPN 使得效率得到巨大的提升

2、170ms per image 2.5-20* faster than faster R-CNN counterpart

- Motivation

1、Region-based detector suchas Fast/Faster R-CNN that apply a costly per-region subnetwork hundreds of times.
相反，这篇文章给出的解决方案并没有这样的 subnetwork，自然而然也就不存在这样的问题

2、注意到 ROIpooing 得到的图像有很多是有重叠，意味着在 roi 之后的处理网络实际上存在着严重的重叠效应，导致计算资源的浪费。

但有一点，其实 per-region subnetwork 是一种不同的视角，如果说主干网络是类似一种大范围的扫视，那么 per-region subnetwork 就是在找到值得关注的物体之后的一种注意力集中的观察。因此主干网络和 per-region subnetwork 分开是有必要的，或者说多阶段的 pipline 是有必要的，更何况它是 end-to-end. 这或许就是本文效果始终差于 faster 的原因。

- 介绍

Object detection → a shared “fully convolutional” subnetwork idependent of RoIs

→ an RoI-wise subnetwork that does not share compuatition.

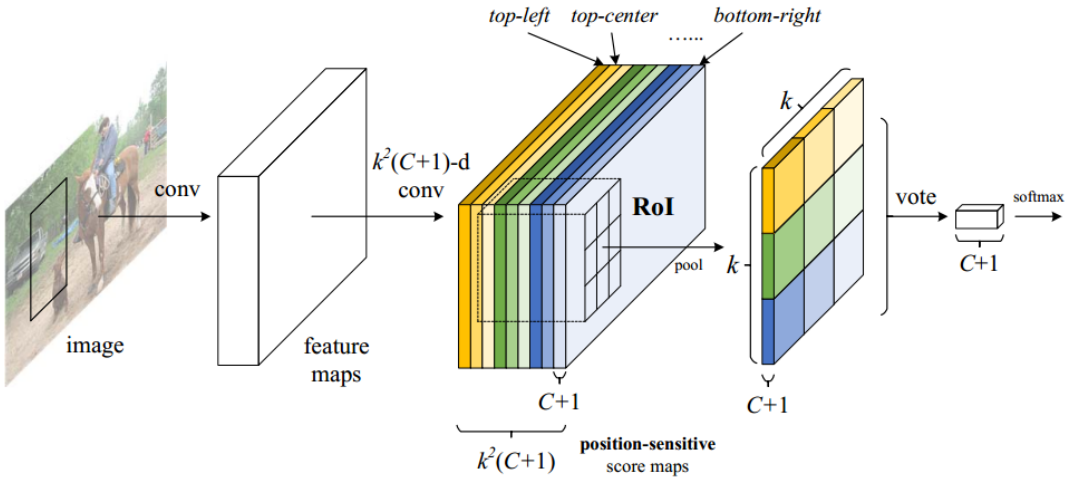


Figure 1: Key idea of **R-FCN** for object detection. In this illustration, there are $k \times k = 3 \times 3$ position-sensitive score maps generated by a fully convolutional network. For each of the $k \times k$ bins in an RoI, pooling is only performed on one of the k^2 maps (marked by different colors).

Position-sensitive score maps & Position-sensitive RoI pooling. To explicitly encode position information into each RoI, we divide each RoI rectangle into $k \times k$ bins by a regular grid. For an RoI rectangle of a size $w \times h$, a bin is of a size $\approx \frac{w}{k} \times \frac{h}{k}$ [9, 7]. In our method, the last convolutional layer is constructed to produce k^2 score maps for each category. Inside the (i, j)-th bin ($0 \leq i, j \leq k-1$), we define a position-sensitive RoI pooling operation that pools only over the (i, j)-th score map:

$$r_c(i, j \mid \Theta) = \sum_{(x,y) \in \text{bin}(i,j)} z_{i,j,c}(x + x_0, y + y_0 \mid \Theta)/n. \quad (1)$$

Here $r_c(i, j)$ is the pooled response in the (i, j)-th bin for the c-th category, $z_{i,j,c}$ is one score map out of the $k^2(C+1)$ score maps, (x_0, y_0) denotes the top-left corner of an RoI, n is the number of pixels in the bin, and Θ denotes all learnable parameters of the network. The (i, j)-th bin spans $\lfloor i\frac{w}{k} \rfloor \leq x < \lceil (i+1)\frac{w}{k} \rceil$ and $\lfloor j\frac{h}{k} \rfloor \leq y < \lceil (j+1)\frac{h}{k} \rceil$. The operation of Eqn.(1) is illustrated in Figure 1, where a color represents a pair of (i, j). Eqn.(1) performs average pooling (as we use throughout this paper), but max pooling can be conducted as well.

公式 1 里并没有体现 k*k 个 channel 的 feature map 合并的过程？
有的，其实 pooled response 的每个 bin(i,j)是由对应的(I,j) score map 生成的
其实就是所谓的前两篇论文中提及的复制粘贴操作，这里居然表达得如此数学。。。

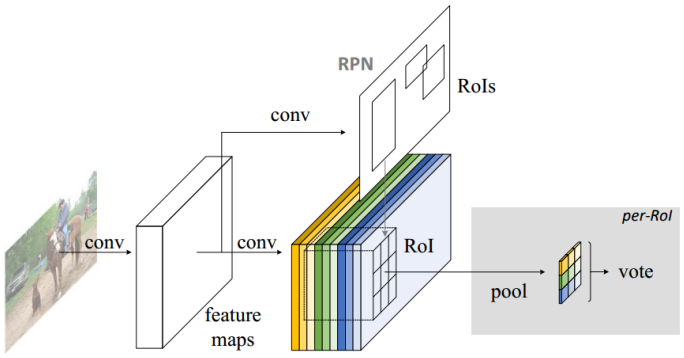


Figure 2: Overall architecture of R-FCN. A Region Proposal Network (RPN) [19] proposes candidate RoIs, which are then applied on the score maps. All learnable weight layers are convolutional and are computed on the entire image; the per-RoI computational cost is negligible.

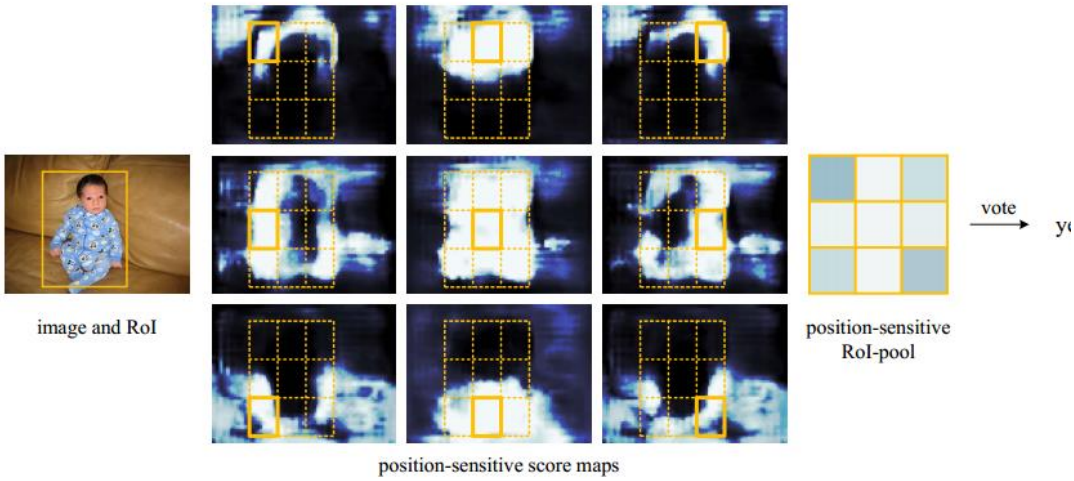


Figure 3: Visualization of R-FCN ($k \times k = 3 \times 3$) for the *person* category.

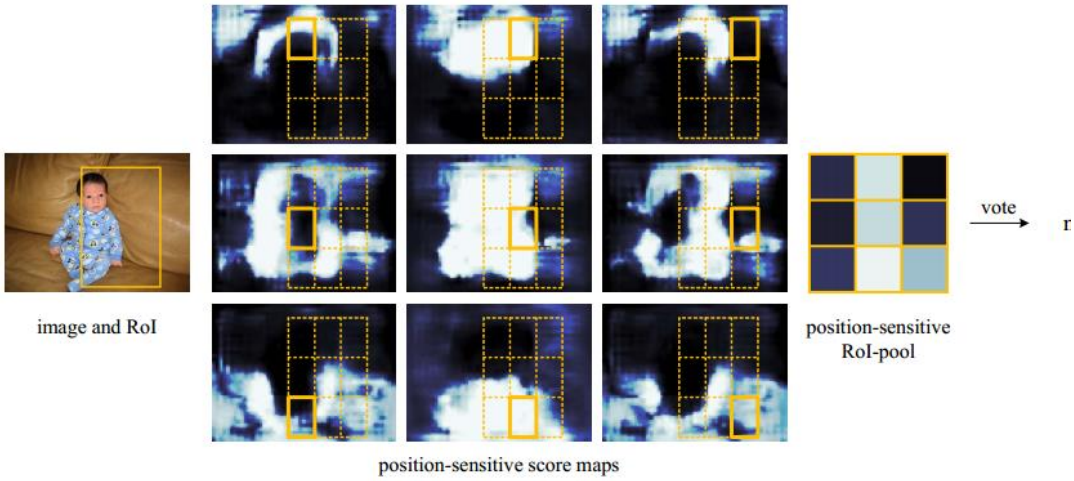


Figure 4: Visualization when an RoI does not correctly overlap the object.

- 实验

Table 2: Comparisons among fully convolutional (or "almost" fully convolutional) strategies using **ResNet-101**. *All competitors in this table use the à trous trick.* Hard example mining is not conducted.

method	RoI output size ($k \times k$)	mAP on VOC 07 (%)
naïve Faster R-CNN	1×1	61.7
	7×7	68.9
class-specific RPN	-	67.6
R-FCN (w/o position-sensitivity)	1×1	*fail*
R-FCN	3×3	75.5
	7×7	**76.6**

Table 3: Comparisons between Faster R-CNN and R-FCN using ResNet-101. Timing is evaluated on a single Nvidia K40 GPU. With OHEM, N RoIs per image are computed in the forward pass, and 128 samples are selected for backpropagation. 300 RoIs are used for testing following [19].

	depth of per-RoI subnetwork	training w/ OHEM?	train time (sec/img)	test time (sec/img)	mAP (%) on VOC07
Faster R-CNN	10		1.2	0.42	76.4
R-FCN	0		0.45	0.17	76.6
Faster R-CNN	10	✓(300 RoIs)	1.5	0.42	79.3
R-FCN	0	✓(300 RoIs)	0.45	0.17	**79.5**
Faster R-CNN	10	✓(2000 RoIs)	2.9	0.42	*N/A*
R-FCN	0	✓(2000 RoIs)	0.46	0.17	79.3

Table 4: Comparisons on PASCAL VOC 2007 *test* set using **ResNet-101**. "Faster R-CNN +++" [10] uses iterative box regression, context, and multi-scale testing.

	training data	mAP (%)	test time (sec/img)
Faster R-CNN [10]	07+12	76.4	0.42
Faster R-CNN +++ [10]	07+12+COCO	**85.6**	3.36
R-FCN	07+12	79.5	0.17
R-FCN multi-sc train	07+12	80.5	0.17
R-FCN multi-sc train	07+12+COCO	**83.6**	0.17

Table 5: Comparisons on PASCAL VOC 2012 *test* set using **ResNet-101**. "07++12" [7] denotes the union set of 07 *trainval+test* and 12 *trainval*. †: `http://host.robots.ox.ac.uk:8080/anonymous/44L5HI.html` ‡: `http://host.robots.ox.ac.uk:8080/anonymous/MVCM2L.html`

	training data	mAP (%)	test time (sec/img)
Faster R-CNN [10]	07++12	73.8	0.42
Faster R-CNN +++ [10]	07++12+COCO	**83.8**	3.36
R-FCN multi-sc train	07++12	77.6†	0.17
R-FCN multi-sc train	07++12+COCO	**82.0**‡	0.17

Scene Text Recognition with Sliding Convolutional Character Models[6]

效果较好，但还不太理解和 FCN+CTC 之间区别。

- 实验结果

Table 2: Recognition accuracies (%) on four English scene text datasets. In the second row, 50, 1k and Full denote the lexicon used, LM denotes the language model and None denotes recognition without language constraints. (*** [14] is not lexicon-free in the strict sense, as its outputs are constrained to a 90k dictionary.**)

	IIIT5k				SVT			IC03				IC13		
Method	**50**	**1k**	**LM**	**None**	**50**	**LM**	**None**	**50**	**Full**	**LM**	**None**	**LM**	**None**	**Model Size**
ABBYY [33]	24.3	-	-	-	35.0	-	-	56.0	55.0	-	-	-	-	-
Wang et al. [33]	-	-	-	-	57.0	-	-	76.0	62.0	-	-	-	-	-
Mishra et al. [23]	64.1	57.5	-	-	73.2	-	-	81.8	67.8	-	-	-	-	-
Novikova et al. [25]	-	-	-	-	72.9	-	-	82.8	-	-	-	-	-	-
Wang et al. [35]	-	-	-	-	70.0	-	-	90.0	84.0	-	-	-	-	-
Bissaco et al. [3]	-	-	-	-	90.4	78.0	-	-	-	-	-	87.6	-	-
Goel et al. [6]	-	-	-	-	77.3	-	-	89.7	-	-	-	-	-	-
Alsharif & Pineau [2]	-	-	-	-	74.3	-	-	93.1	88.6	-	-	-	-	-
Almazan et al. [1]	91.2	82.1	-	-	89.2	-	-	-	-	-	-	-	-	-
Yao et al. [38]	80.2	69.3	-	-	75.9	-	-	88.5	80.3	-	-	-	-	-
R.-Serrano et al. [27]	76.1	57.4	-	-	70.0	-	-	-	-	-	-	-	-	-
Jaderberg et al. [16]	-	-	-	-	86.1	-	-	96.2	91.5	-	-	-	-	-
Su & Lu et al. [32]	-	-	-	-	83.0	-	-	92.0	82.0	-	-	-	-	-
Gordo [8]	93.3	86.6	-	-	91.8	-	-	-	-	-	-	-	-	-
Jaderberg et al. [14]	97.1	92.7	-	-	95.4	-	80.7*	**98.7**	**98.6**	-	**93.1***	-	**90.8***	490M
Jaderberg et al. [15]	95.5	89.6	-	-	93.2	-	71.7	97.8	97.0	-	89.6	-	81.8	304M
Shi et al. [30]	97.8	95.0	-	81.2	**97.5**	-	82.7	**98.7**	98.0	-	91.9	-	89.6	8.3M
Shi et al. [29]	96.2	93.8	-	81.9	95.5	-	81.9	98.3	96.2	-	90.1	-	88.6	-
Lee et al. [19]	96.8	94.4	-	78.4	96.3	-	80.7	97.9	97.0	-	88.7	-	90	-
Ours(n=1)	98.6	96.3	83.0	80.9	94.4	82.1	76.7	97.2	96.0	88.8	84.1	88.2	84.9	8.1M
Ours(n=3)	**98.9**	**96.7**	**83.5**	81.6	95.1	**84.1**	76.5	97.7	96.4	90.5	84.5	89.0	85.2	-
Ours(n=1, Residual)	98.7	96.1	80.6	78.2	95.1	79.9	72.5	97.6	96.5	87.1	81.1	86.9	81.4	**0.41M**

Semantic Instance Segmentation via Deep Metric Learning[7]

- 扩展方向\未读完\想法\理解

IoU=0.5 的情况下排名第 4，但在非 proposed method 中排名第一

- Motivation:

1、基于 box 框出物体后再 segment 的做法在处理复杂问题时的效果并不好;

- 创新点

1、计算相邻两像素属于同一个物体的可能性，并根据这些可能性将像素聚焦在一起
极像非监督的传统切分方法——super pixels，但不同点在于，基于深度卷积网络，我们可以根据空间背景信息来有效作出切分判断。（this avoids ambiguities such as whether to treat parts of an object (e.g., the shirt and pants of a person) as separate segments, which plagues evaluation of unsupervised methods.）
2、借用 deep embedding model 来学习相似度量（the similarity metric）
类似于 FaceNet，但本文章是基于相依像素级别，而 FaceNet 是基于 bounding boxes.
3、计算 embedding space 下，各个点到 K 个种点“seed points”之间的距离
4、种点的选择是另外一个模型来生成的。（其实应该类似 semantic seg 的 score map，只不过预测的是种子，其实有点像之前说的 predict a centeredness score）
5、跟文章[19]很像 Associative Embedding:End-to-End learning for joint detection and grouping.

- Related work

1、box-free：modify Faster RCNN → predict a centeredness score
但长物体不好，有些物体并没有中点
2、MNC method，其实和 mask-RCNN 非常像
3、FCIS 和 DeepMask\SharpMask 非常像（后者没有使用 position-sensitive score maps）
4、[22]和[24]和此文章一样是针对按顺序每次预测一个 instance 的
5、the watershed algorithm

- 框架

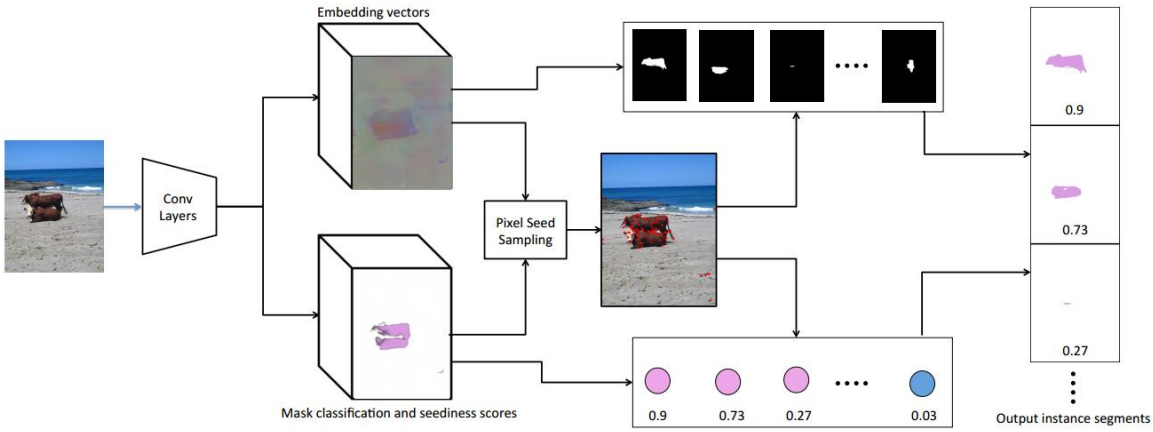


Figure 1. Given an image, our model predicts the embedding vector of each pixel (top head of the network) as well as the classification score of the mask each pixel will generate if picked as a seed (bottom head of the network). We derive the seediness scores from the classification scores and use them to choose which seed points in the image to sample. Each seed point generates a mask based on the embedding vectors; each mask is then associated with a class label and confidence score. In this figure, pink color corresponds to the “cow” class and blue to the “background” class.

1、每一个 seed point 都会基于 embedding space 生成一个 mask.
2、相似性：

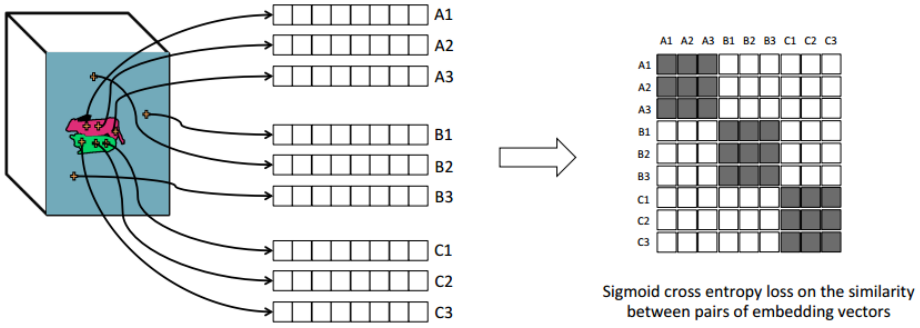
$$\sigma(p,q) = \frac{2}{1+\exp(||e_p - e_q||_2^2)} \quad (1)$$

We see that for pairs of pixels that are close in embedding space, we have $\sigma(p,q) = \frac{2}{1+e^0} = 1$, and for pairs of pixels that are far in embedding space, we have $\sigma(p,q) = \frac{2}{1+e^\infty} = 0$.

We train the network by minimizing the following loss:

$$\mathcal{L}_e = -\frac{1}{|S|}\sum_{p,q\in S} w_{pq}\left[1_{\{y_p=y_q\}}\log(\sigma(p,q)) + 1_{\{y_p\neq y_q\}}\log(1-\sigma(p,q))\right]$$

Loss 里面有一个点很有趣:w_{pq}跟物体的大小是呈反比的，以保证 loss 不会偏向于大的物体。



3、生成 mask

4、种子生成网络（含分类）

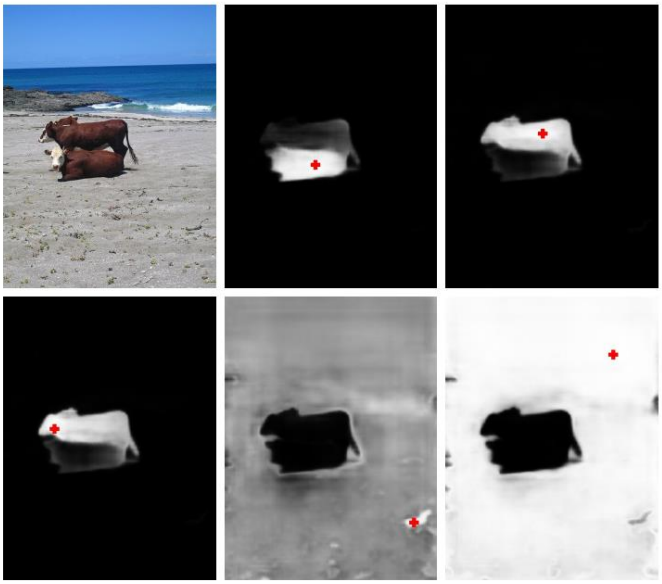


Figure 4. We visualize the similarity of each pixel and a randomly chosen seed pixel in each image. The randomly chosen seed pixel is shown by a red mark in the image. The brighter the pixels the higher the similarity.

- 实验

Table 1. Per-class instance-level segmentation comparison using APr metric over 20 classes at 0.5, 0.6 and 0.7 IoU on the PASCAL VOC 2012 validation set. All numbers are in %.

IoU score	Method	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	average
0.5	SDS [10]	58.8	0.5	60.1	34.4	29.5	60.6	40.0	73.6	6.5	52.4	31.7	62.0	49.1	45.6	47.9	22.6	43.5	26.9	66.2	66.1	43.8
	Chen et al. [4]	63.6	0.3	61.5	43.9	33.8	67.3	46.9	74.4	8.6	52.3	31.3	63.5	48.8	47.9	48.3	26.3	40.1	33.5	66.7	67.8	46.3
	PFN [17]	76.4	15.6	74.2	54.1	26.3	73.8	31.4	92.1	17.4	73.7	48.1	82.2	81.7	72.0	48.4	23.7	57.7	64.4	88.9	72.3	58.7
	MNC [6]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	63.5
	Li et al. [15]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	65.7
	R2-IOS [16]	87.0	6.1	90.3	67.9	48.4	86.2	68.3	90.3	24.5	84.2	29.6	91.0	71.2	79.9	60.4	42.4	67.4	61.7	94.3	82.1	66.7
	Assoc. Embedding [19]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	35.1
	Ours	69.7	1.2	78.2	53.8	42.2	80.1	57.4	88.8	16.0	73.2	57.9	88.4	78.9	80.0	68.0	28.0	61.5	61.3	87.5	70.4	62.1
0.6	SDS [10]	43.6	0	52.8	19.5	25.7	53.2	33.1	58.1	3.7	43.8	29.8	43.5	30.7	29.3	31.8	17.5	31.4	21.2	57.7	62.7	34.5
	Chen et al. [4]	57.1	0.1	52.7	24.9	27.8	62.0	36.0	66.8	6.4	45.5	23.3	55.3	33.8	35.8	35.6	20.1	35.2	28.3	59.0	57.6	38.2
	PFN [17]	73.2	11.0	70.9	41.3	22.2	66.7	26.0	83.4	10.7	65.0	42.4	78.0	69.2	72.0	38.0	19.0	46.0	51.8	77.9	61.4	51.3
	R2-IOS [16]	79.7	1.5	85.5	53.3	45.6	81.1	62.4	83.1	12.1	75.7	20.2	81.5	49.7	63.9	51.2	35.7	56.2	56.7	87.9	78.8	58.1
	Ours	64.2	0.1	64.8	37.2	34.5	73.5	50.6	84.7	8.9	59.3	48.2	84.3	65.1	69.6	56.6	14.9	51.8	50.7	81.7	64.4	53.3
0.7	SDS [10]	17.8	0	32.5	7.2	19.2	47.7	22.8	42.3	1.7	18.9	16.9	20.6	14.4	12.0	15.7	5.0	23.7	15.2	40.5	51.4	21.3
	Chen et al. [4]	40.8	0.07	40.1	16.2	19.6	56.2	26.5	46.1	2.6	25.2	16.4	36.0	22.1	20.0	22.6	7.7	27.5	19.5	47.7	46.7	27.0
	PFN [17]	68.5	5.6	60.4	34.8	14.9	61.4	19.2	78.6	4.2	51.1	28.2	69.6	60.7	60.5	26.5	9.8	35.1	43.9	71.2	45.6	42.5
	MNC [6]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	41.5
	Li et al. [15]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	52.1
	R2-IOS [16]	54.5	0.3	73.2	34.3	38.4	71.1	54.0	76.9	6.0	63.3	13.1	67.0	26.9	39.2	33.2	25.4	44.8	45.4	81.5	74.6	46.2
	Assoc. Embedding [19]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	26.0
	Ours	53.0	0.0	51.8	24.9	21.9	69.2	40.1	76.6	4.1	43.1	21.1	74.4	44.7	54.3	40.3	7.5	40.5	39.6	69.5	52.6	41.5

[1] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," *arXiv preprint arXiv:1703.06870,* 2017.

[2] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, "Fully convolutional instance-aware semantic segmentation," *arXiv preprint arXiv:1611.07709,* 2016.

[3] J. Dai, K. He, Y. Li, S. Ren, and J. Sun, "Instance-sensitive fully convolutional networks," in *European Conference on Computer Vision*, 2016, pp. 534-549.

[4] P. O. Pinheiro, R. Collobert, and P. Dollár, "Learning to segment object candidates," in *Advances in Neural Information Processing Systems*, 2015, pp. 1990-1998.

[5] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Advances in neural information processing systems*, 2016, pp. 379-387.

[6] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu, "Deep Direct Regression for Multi-Oriented Scene Text Detection," *arXiv preprint arXiv:1703.08289,* 2017.

[7] A. Fathi, Z. Wojna, V. Rathod, P. Wang, H. O. Song, S. Guadarrama*, et al.*, "Semantic Instance Segmentation via Deep Metric Learning," *arXiv preprint arXiv:1703.10277,* 2017.

Action Segmentation and Detection

1. Temporal Convolutional Networks for **Action** Segmentation and Detection
2. One-Shot Video **Object** Segmentation
3. Budget-Aware Deep Semantic **Video** Segmentation
4. Weakly Supervised Actor-**Action** Segmentation via Robust Multi-Task Ranking
5. Learning **Video** Object Segmentation From Static Images
6. **Video** Segmentation via Multiple Granularity Analysis
7. Primary **Object** Segmentation in Videos Based on Region Augmentation and Reduction
8. FusionSeg: Learning to Combine Motion and Appearance for Fully Automatic Segmentation of Generic **Objects** in Videos
9. Fast Multi-Frame Stereo Scene Flow With **Motion** Segmentation
10. SPFTN: A Self-Paced Fine-Tuning Network for Segmenting Objects in Weakly Labelled **Videos**
11. Online **Video Object** Segmentation via Convolutional Trident Network
12. Object Co-**Skeletonization** With Co-Segmentation
13. **Temporal Action** Co-Segmentation in 3D Motion Capture Data and **Videos**
14. Weakly Supervised Semantic Segmentation Using Web-Crawled **Videos**

Object

1. Superpixel-Based **Tracking**-By-Segmentation Using Markov Chains

2. Detecting Oriented Text in Natural Images by Linking Segments

3D Classification and Segmenation

1. PointNet: Deep Learning on Point Sets for **3D** Classification and Segmentation
2. SyncSpecCNN: Synchronized Spectral CNN for **3D Shape** Segmentation
3. **Multi-Scale** FCN With Cascaded **Instance** Aware Segmentation for Arbitrary Oriented Word Spotting in the Wild
4. 3D Shape **Segmentation** With Projective Convolutional Networks

Unsettled

1. Efficient Optimization for Hierarchically-structured Interacting Segments (HINTS)
2. Coarse-To-Fine Segmentation With Shape-Tailored Continuum Scale Spaces
3. **4D** Light Field Superpixel and Segmentation
4. Joint Sequence Learning and Cross-Modality Convolution for 3D **Biomedical** Segmentation
5. Improving **Facial** Attribute Prediction Using Semantic Segmentation

Dataset

2. IRINA: Iris Recognition (Even) in Inaccurately Segmented **Data**

Ordinary

1. Semantically Coherent Co-Segmentation and Reconstruction of Dynamic **Scenes**
2. Pixelwise **Instance** Segmentation With a Dynamically Instantiated Network
3. Simple Does It: Weakly Supervised Instance and **Semantic** Segmentation
4. WILDCAT: Weakly Supervised Learning of Deep ConvNets for Image Classification, **Pointwise** Localization and Segmentation
5. Convolutional Random Walk Networks for Semantic **Image** Segmentation
6. **Semantic** Amodal Segmentation
7. Object Region Mining With Adversarial Erasing: A Simple Classification to **Semantic** Segmentation Approach
8. RefineNet: Multi-Path Refinement Networks for High-Resolution **Semantic** Segmentation
9. **Semantic** Segmentation via Structured Patch Prediction, Context CRF and Guidance CRF
10. MCMLSD: A Dynamic Programming Approach to **Line** Segment Detection
11. Loss Max-Pooling for **Semantic** Image Segmentation
12. Fully Convolutional Instance-Aware **Semantic** Segmentation
13. Instance-Level Salient **Object** Segmentation
14. Not All Pixels Are Equal: Difficulty-Aware **Semantic** Segmentation via Deep Layer Cascade
15. Locality-Sensitive Deconvolution Networks With Gated Fusion for **RGB-D Indoor** Semantic Segmentation
16. Learning to Align **Semantic** Segmentation and 2.5D Maps for Geolocalization
17. Combining Bottom-Up, Top-Down, and Smoothness Cues for Weakly Supervised **Image** Segmentation
18. Webly Supervised **Semantic** Segmentation
19. Full-Resolution Residual Networks for **Semantic** Segmentation in Street Scenes
20. Large Kernel Matters -- Improve **Semantic** Segmentation by Global Convolutional Network
21. Exploiting Saliency for **Object** Segmentation From Image Level Labels
22. STD2P: RGBD **Semantic** Segmentation Using Spatio-Temporal Data-Driven Pooling
23. Deep Watershed Transform for **Instance** Segmentation
24. Indoor Scene Parsing With **Instance** Segmentation, Semantic Labeling and Support Relationship Inference

25. Boundary-Aware **Instance** Segmentation
26. Learning Object Interactions and Descriptions for **Semantic Image** Segmentation
27. Improving **RANSAC**-**Based** Segmentation Through CNN Encapsulation
28. End-To-End **Instance** Segmentation With Recurrent Attention
29. Joint Multi-Person **Pose** Estimation and Semantic Part Segmentation
30. Learning Random-Walk Label Propagation for Weakly-Supervised **Semantic** Segmentation
31. FastMask: Segment Multi-Scale Object Candidates in One Shot