Mask R-CNN

Presenter: Shuhan Tan

2021.9.1
Object Detection

Identify, locate and classify objects.

Image source: Matlab
Object Detection supports many applications

Image source: Mask RCNN - COCO - instance segmentation
Object Detection supports many applications

Image source: [Mask R-CNN for Surgery Robot](#)
Object Detection supports many applications

Image source: viso.ai
Object Detection is popular in research

Image source: Object Detection in 20 Years: A Survey.
Taxonomy of Object Detection

Image source: cvhub
Parallel development of One-stage/Two-stage methods

Image source: cvhub
Upsurge of Anchor Free methods

Image source: cvhub
Questions:

❖ What is Anchor? What is Stage?
❖ Why are them the key features to distinguish different methods?
❖ What are the Pros and cons of different method family?
… by understanding Mask-RCNN

We introduce a representative **Anchor-based Two-Stage** detection method:

- **Key design choices**
- **Advantages**
- **Drawbacks**

---

**Mask R-CNN**

Kaiming He  Georgia Gkioxari  Piotr Dollár  Ross Girshick  
Facebook AI Research (FAIR)

**Abstract**

We present a conceptually simple, flexible, and general framework for object instance segmentation. Our approach efficiently detects objects in an image while simultaneously generating a high-quality segmentation mask for each instance. The method, called Mask R-CNN, extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. Mask R-CNN is simple to train and adds only a small overhead to Faster R-CNN, running at 5 fps. Moreover, Mask R-CNN is easy to generalize to other tasks (e.g.,

---

Figure 1. The **Mask R-CNN** framework for instance segmentation.
Background

2D Object Detection
Bounding Box (bbox)

Image source: Intro to Object Detection
Intersection over Union (IoU)

Image source: Intro to Object Detection
True positive prediction

True positive:

IoU (Prediction, Ground Truth) > threshold (=0.5, 0.7, 0.9...)
Average Precision (AP)

<table>
<thead>
<tr>
<th></th>
<th>TP/FP</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>TP</td>
<td>1/1 = 1</td>
<td>1/3 = 0.33</td>
</tr>
<tr>
<td>3</td>
<td>FP</td>
<td>1/2 = 0.5</td>
<td>1/3 = 0.33</td>
</tr>
<tr>
<td>4</td>
<td>TP</td>
<td>2/3 = 0.67</td>
<td>2/3 = 0.67</td>
</tr>
<tr>
<td>5</td>
<td>FP</td>
<td>2/4 = 0.5</td>
<td>2/3 = 0.67</td>
</tr>
<tr>
<td>6</td>
<td>FP</td>
<td>2/5 = 0.4</td>
<td>2/3 = 0.67</td>
</tr>
<tr>
<td>7</td>
<td>TP</td>
<td>3/6 = 0.5</td>
<td>3/3 = 1</td>
</tr>
<tr>
<td>8</td>
<td>FP</td>
<td>3/7 = 0.43</td>
<td>3/3 = 1</td>
</tr>
</tbody>
</table>

Average Precision* = \( \frac{1}{7} (1 + 0.5 + 0.67 + 0.5 + 0.4 + 0.5 + 0.43) \)
Average Precision (AP)

<table>
<thead>
<tr>
<th></th>
<th>TP/FP</th>
<th>Precision</th>
<th>Recall</th>
<th>Precision_inter</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>TP</td>
<td>1/1 = 1</td>
<td>1/3 = 0.33</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>FP</td>
<td>1/2 = 0.5</td>
<td>1/3 = 0.33</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>TP</td>
<td>2/3 = 0.67</td>
<td>2/3 = 0.67</td>
<td>0.67</td>
</tr>
<tr>
<td>5</td>
<td>FP</td>
<td>2/4 = 0.5</td>
<td>2/3 = 0.67</td>
<td>0.67</td>
</tr>
<tr>
<td>6</td>
<td>FP</td>
<td>2/5 = 0.4</td>
<td>2/3 = 0.67</td>
<td>0.67</td>
</tr>
<tr>
<td>7</td>
<td>TP</td>
<td>3/6 = 0.5</td>
<td>3/3 = 1</td>
<td>0.5</td>
</tr>
<tr>
<td>8</td>
<td>FP</td>
<td>3/7 = 0.43</td>
<td>3/3 = 1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Average Precision = \( \frac{1}{7} \times (1 + 0.67 + 0.67 + 0.67 + 0.5 + 0.5) \)
Faster R-CNN
The base architecture of Mask R-CNN
Use CNN to process all possible bounding boxes

1. Apply a CNN to many different crops of image
2. Classify whether this crop contains an object or is a background.

Image source: Stanford CS231n
Use CNN to process all possible bounding boxes

**Problem:** there are **too many** possible bounding boxes! (\(O(W^2H^2)\))

Image source: [Stanford CS231n](https://cs231n.stanford.edu)
Region Proposal

Use a **Region Proposal** method to obtain a small number (~2000) of regions that may contain objects.

Filter out most false positive regions.

Image source: [Stanford CS231n](https://cs231n.stanford.edu/)

---

CS391R: Robot Learning (Fall 2021)
Two-stage Object Detection

Two stages:

1. Generate a small number (~2000) of **Region Proposals**.

   1. Use a **CNN** to process each region, predict:
      
      a. **Object classification** label (object class or “background”)
      b. **Bounding box offset** value
Faster R-CNN

Two stages:

1. First stage: Run per image
   - backbone feature network
   - Region Proposal Network to generate ~2k region proposals.

1. Second stage: Run per region
   - crop region features
   - predict object label
   - predict bbox offset

Source: Faster R-CNN (2015)
Faster R-CNN

Two stages:

1. First stage: Run per image
   - backbone feature network
   - Region Proposal Network to generate ~2k region proposals.

1. Second stage: Run per region
   - crop region features
   - predict object label
   - predict bbox offset
Region Proposal Network: **Anchor filtering**

Main idea of RPN:

Use CNN to predict which region is more likely to have an object.

Same problem:

There are way **too many possible regions** than we can afford to predict!

**Image source:** [Stanford CS231n](http://cs231n.stanford.edu/)
Region Proposal Network: **Anchor filtering**

**Solution:**

At each location, only consider \( k \) predefined anchor boxes.

**Image source:** [Stanford CS231n](https://cs231n.stanford.edu)
Region Proposal Network: Anchor filtering

Image source: Stanford CS231n
Region Proposal Network: **Anchor filtering**

Image source: [Stanford CS231n](https://cs231n.stanford.edu/)

---

CS391R: Robot Learning (Fall 2021)
Region Proposal Network: **Anchor filtering**

Image source: [Stanford CS231n](https://cs231n.stanford.edu/)

---

For positive boxes, also predict a transformation from the anchor to the ground-truth box (regress 4 numbers per pixel).
Region Proposal Network: Anchor filtering

Image source: Stanford CS231n
Region Proposal Network: **Anchor filtering**

Image source: [Stanford CS231n](https://cs231n.stanford.edu/)

---->
Sort the $K \times 20 \times 15$ boxes by their “object” score; take top ~2000 as proposals.
Faster R-CNN

Two stages:

1. First stage: Run per image
   - backbone feature network
   - Region Proposal Network to generate ~2k region proposals.

1. Second stage: Run per region
   - crop region features
   - predict **object label**
   - predict **bbox offset**
Feature crop: RoIPool

**Problem: size conflict**

- Region proposals are with different sizes
- Prediction network takes features of the same size.

RoIPool:

obtain a **fixed size feature** for different region proposals
Feature crop: RoIPool

**Goal:** obtain a fixed-size (d x 2 x 2) feature for region proposals with different sizes
Feature crop: RoIPool

**Goal**: obtain a fixed-size (d x 2 x 2) feature for region proposals with different sizes
Feature crop: RoIPool

**Goal:** obtain a fixed-size (d x 2 x 2) feature for region proposals with different sizes
Feature crop: RoIPool

**Goal:** obtain a fixed-size (d x 2 x 2) feature for region proposals with different sizes
Region Prediction
Faster R-CNN training losses

Jointly train with 4 losses:

1. RPN object/background classification
2. RPN bbox offset regression
3. Final object classification
4. Final bbox offset regression
Results

Detection results (runs in 5fps on gpu)
Mask R-CNN
From box to pixel ...

Instance segmentation: predict **pixel-level** label for each instance in different classes
... we just need a mask!
Highlights

**Simplicity:** Simple to extend from Faster R-CNN.

**Performance:** Outperformed all existing, single-model entries on every task (COCO 2017).

**Efficiency:** Only add a small overhead to Faster R-CNN, running at 5fps.

**Flexibility:** Adapt to do human pose estimation with the same framework.
Mask R-CNN

Main modifications from Faster R-CNN:

- Use of a **Mask prediction header**
- Replace RoIPool with **RoIAlign**
Mask R-CNN: Mask prediction

Predicting a 14x14 binary mask for each of the 80 categories
Mask R-CNN: Mask example

Image source: Stanford CS231n
Problems of RoI Pool
Problems of RoIPool

These quantizations leads to **misalignment** of regions’ **locations** in **image and feature** spaces.

Fine for classification, but **harmful for segmentation**, which requires accurate pixel alignment.
Mask R-CNN: RoIAlign

RoIAlign goals:

- avoid location quantization
- accurately map location of regions from the image space to the feature space.
Mask R-CNN: RoIAlign

Example: turn a $3.2 \times 2.8$ region feature into a standard $2 \times 2$ feature.

<table>
<thead>
<tr>
<th></th>
<th>0.4</th>
<th>0.08</th>
<th>0.73</th>
<th>0.57</th>
<th>0.13</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.88</td>
<td>0.13</td>
<td>0.32</td>
<td>0.64</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>0.98</td>
<td>0.64</td>
<td>0.16</td>
<td>0.16</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td>0.97</td>
<td>0.43</td>
<td>0.08</td>
<td>0.08</td>
<td>0.18</td>
<td></td>
</tr>
<tr>
<td>0.69</td>
<td>0.88</td>
<td>0.9</td>
<td>0.9</td>
<td>0.87</td>
<td></td>
</tr>
</tbody>
</table>

Image source: Mask R-CNN unmased
Mask R-CNN: RoIAlign

Directly split the region evenly into 2x2 grids.

Image source: [Mask R-CNN unmased](#)
Mask R-CNN: RoIAlign

Sample points in each grid

Image source: Mask R-CNN unmased
Mask R-CNN: RoIAlign

Use bi-linear interpolation to compute feature for each sample point

Image source: Mask R-CNN unmased
Mask R-CNN: RoIAlign

Compute feature value for each grid with max/avg pooling of sample points

Image source: Mask R-CNN unmased
Mask R-CNN

Main modifications from Faster R-CNN:

- Use of a **Mask prediction header**
- Replace RoIPool with **RoIAlign**
Mask R-CNN: Training

Jointly train with 5 losses:

1. RPN object/background classification
2. RPN bbox offset regression
3. Final object classification
4. Final bbox offset regression
5. Mask prediction
Experiment
Main result: instance segmentation

- Dataset: COCO (80k train images; 35k val images)

- Baselines:
  - MNC (2016): winner of COCO 2015 challenge
  - FCIS (2017): winner of COCO 2016 challenge

- Metrics:
  - AP (averaged over IoU thresholds .50:.05:.90)
  - AP@50 / AP@75
  - AP_S / AP_M / AP_L (AP of different object scales)
Main result: instance segmentation

<table>
<thead>
<tr>
<th></th>
<th>backbone</th>
<th>AP</th>
<th>AP&lt;sub&gt;50&lt;/sub&gt;</th>
<th>AP&lt;sub&gt;75&lt;/sub&gt;</th>
<th>AP&lt;sub&gt;S&lt;/sub&gt;</th>
<th>AP&lt;sub&gt;M&lt;/sub&gt;</th>
<th>AP&lt;sub&gt;L&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNC [10]</td>
<td>ResNet-101-C4</td>
<td>24.6</td>
<td>44.3</td>
<td>24.8</td>
<td>4.7</td>
<td>25.9</td>
<td>43.6</td>
</tr>
<tr>
<td>FCIS [26] +OHEM</td>
<td>ResNet-101-C5-dilated</td>
<td>29.2</td>
<td>49.5</td>
<td>-</td>
<td>7.1</td>
<td>31.3</td>
<td>50.0</td>
</tr>
<tr>
<td>FCIS+++ [26] +OHEM</td>
<td>ResNet-101-C5-dilated</td>
<td>33.6</td>
<td>54.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>ResNet-101-C4</td>
<td>33.1</td>
<td>54.9</td>
<td>34.8</td>
<td>12.1</td>
<td>35.6</td>
<td>51.1</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>ResNet-101-FPN</td>
<td>35.7</td>
<td>58.0</td>
<td>37.8</td>
<td>15.5</td>
<td>38.1</td>
<td>52.4</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>ResNeXt-101-FPN</td>
<td>37.1</td>
<td>60.0</td>
<td>39.4</td>
<td>16.9</td>
<td>39.9</td>
<td>53.5</td>
</tr>
</tbody>
</table>

Mask R-CNN **outperformed** the SOTA methods **in all entries** (AP improve ~3)
Main result: instance segmentation
Main result: instance segmentation

Mask R-CNN has much better performance on overlapping objects than FCIS
Ablation study: RoIAlign

❖ Question:

How important it is to align pixel location accurately?

❖ Compared methods:

○ RoIPool (2 quantization step 2)
○ RoIWarp (1 quantization step)
○ RoIAlign (0 quantization step, accurate position)
RoIAlign is key to improve instance segmentation

<table>
<thead>
<tr>
<th></th>
<th>align?</th>
<th>bilinear?</th>
<th>agg.</th>
<th>AP</th>
<th>AP$_{50}$</th>
<th>AP$_{75}$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RoIPool [12]</strong></td>
<td></td>
<td></td>
<td>max</td>
<td>26.9</td>
<td>48.8</td>
<td>26.4</td>
</tr>
<tr>
<td><strong>RoI Warp [10]</strong></td>
<td>✓</td>
<td>✓</td>
<td>max</td>
<td>27.2</td>
<td>49.2</td>
<td>27.1</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td></td>
<td>ave</td>
<td>27.1</td>
<td>48.9</td>
<td>27.1</td>
</tr>
<tr>
<td><strong>RoIAlign</strong></td>
<td>✓</td>
<td>✓</td>
<td>max</td>
<td>30.2</td>
<td>51.0</td>
<td>31.8</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>✓</td>
<td>ave</td>
<td>30.3</td>
<td>51.2</td>
<td>31.5</td>
</tr>
</tbody>
</table>

RoIAlign improves AP by $\sim3$ points and AP$@75$ by $\sim5$ points
Ablation study: Detection with Mask / RoIAlign

❖ Question:
  ○ Is mask prediction helping object detection accuracy?
  ○ Is RoIAlign helping object detection accuracy?

❖ Compared methods:
  ○ Faster R-CNN and variants
Mask / RoIAlign helps object detection

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>backbone</td>
<td></td>
<td></td>
<td></td>
<td>backbone</td>
<td></td>
<td>backbone</td>
</tr>
<tr>
<td>AP^{bb}_{bb}</td>
<td>AP^{bb}_{50}</td>
<td>AP^{bb}_{75}</td>
<td></td>
<td>AP^{bb}_{bb}</td>
<td>AP^{bb}_{bb}</td>
<td>AP^{bb}_{bb}</td>
</tr>
<tr>
<td>34.9</td>
<td>55.7</td>
<td>37.4</td>
<td></td>
<td>15.6</td>
<td>38.7</td>
<td>50.9</td>
</tr>
<tr>
<td>36.2</td>
<td>59.1</td>
<td>39.0</td>
<td></td>
<td>18.2</td>
<td>39.0</td>
<td>48.2</td>
</tr>
<tr>
<td>34.7</td>
<td>55.5</td>
<td>36.7</td>
<td></td>
<td>13.5</td>
<td>38.1</td>
<td>52.0</td>
</tr>
<tr>
<td>36.8</td>
<td>57.7</td>
<td>39.2</td>
<td></td>
<td>16.2</td>
<td>39.8</td>
<td><strong>52.1</strong></td>
</tr>
<tr>
<td>37.3</td>
<td>59.6</td>
<td>40.3</td>
<td></td>
<td>19.8</td>
<td>40.2</td>
<td>48.8</td>
</tr>
<tr>
<td>38.2</td>
<td>60.3</td>
<td>41.7</td>
<td></td>
<td>20.1</td>
<td>41.1</td>
<td>50.2</td>
</tr>
<tr>
<td><strong>39.8</strong></td>
<td><strong>62.3</strong></td>
<td><strong>43.4</strong></td>
<td></td>
<td><strong>22.1</strong></td>
<td><strong>43.2</strong></td>
<td>51.2</td>
</tr>
</tbody>
</table>

The gain of Mask R-CNN over Faster R-CNN come from:

- Using RoIAlign (+1.1 AP_{bb})
- Mask prediction (+0.9 AP_{bb})
Discussions
Understanding the trend

Object Detection Milestones

Traditional Detection Methods

Image source: cvhub
Understanding the trend

Questions:

❖ What is Anchor? What is Stage?
❖ Why are them the key features to distinguish different methods?
❖ What are the Pros and cons of different method family?
Pros/Cons of Mask R-CNN: Two-stage

Mask R-CNN is a *Two-stage* method: 1) region proposal; 2) region classification/regression

❖ Pros
  ○ The second-stage model is easier to train.
  ○ The accuracy is usually higher.

❖ Cons: Slow
  ○ every region needs to be individually processed.
  ○ redundant computations for classification/regression in both stages.
Alternatives: One-stage method

YOLO runs at **45fps**, while Mask R-CNN runs at **5fps**

Alternatives: One-stage method

Trade Off between speed and accuracy

Source: YOLOv3
Pros/Cons of Mask R-CNN: Anchor-based

Mask R-CNN is an Anchor-based method: pre-defined anchors for region proposal network.

❖ Pros
  ○ Well-designed anchors makes region proposal task easier

❖ Cons
  ○ Waste computation on processing all the anchors.
  ○ Anchors' parameters needs to be carefully designed, specifically for different tasks.
  ○ Imbalance ratio between positive & negative anchors.
Alternatives (Anchor-free methods)

Directly predict top-left and bottom-right corners

Frontiers: Real-time detection in an open-world

❖ Detection across multiple datasets.
   Zhou et al, "Simple multi-dataset detection", 2021

❖ Detection across multiple data domain.

❖ Open-world incremental detection.

❖ Detection with resource constraints.

❖ Detection with input of multiple modalities.
Extended Readings

Anchor-based Two-stage detectors:

- Faster R-CNN
- FPN
- Cascade R-CNN

Anchor-based One-stage detectors:

- RetinaNet
- YOLOv5

Anchor-free detectors:

- CornerNet
- Objects as Points
Summary
Summary

❖ Object detection application and backgrounds
❖ Recent trend of object detection research: Stage and Anchor
❖ Faster R-CNN
❖ Mask R-CNN
❖ Pros/cons of Mask R-CNN and its alternatives
❖ Future directions for 2D object detection
Training process of Faster R-CNN

1. Pre-train a CNN network on image classification tasks.
2. Fine-tune the RPN (region proposal network) end-to-end for the region proposal task, which is initialized by the pre-train image classifier. Positive samples have IoU (intersection-over-union) > 0.7, while negative samples have IoU < 0.3.
   ○ Slide a small n x n spatial window over the conv feature map of the entire image.
   ○ At the center of each sliding window, we predict multiple regions of various scales and ratios simultaneously. An anchor is a combination of (sliding window center, scale, ratio). For example, 3 scales + 3 ratios => k=9 anchors at each sliding position.
3. Train a Fast R-CNN object detection model using the proposals generated by the current RPN
4. Then use the Fast R-CNN network to initialize RPN training. While keeping the shared convolutional layers, only fine-tune the RPN-specific layers. At this stage, RPN and the detection network have shared convolutional layers!
5. Finally fine-tune the unique layers of Fast R-CNN
6. Step 4-5 can be repeated to train RPN and Fast R-CNN alternatively if needed.

Source: Lilian Weng
Loss function for Faster R-CNN (RPN)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_i$</td>
<td>Predicted probability of anchor $i$ being an object.</td>
</tr>
<tr>
<td>$p_i^*$</td>
<td>Ground truth label (binary) of whether anchor $i$ is an object.</td>
</tr>
<tr>
<td>$t_i$</td>
<td>Predicted four parameterized coordinates.</td>
</tr>
<tr>
<td>$t_i^*$</td>
<td>Ground truth coordinates.</td>
</tr>
<tr>
<td>$N_{\text{cls}}$</td>
<td>Normalization term, set to be mini-batch size (~256) in the paper.</td>
</tr>
<tr>
<td>$N_{\text{box}}$</td>
<td>Normalization term, set to the number of anchor locations (~2400) in the paper.</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>A balancing parameter, set to be ~10 in the paper (so that both $L_{\text{cls}}$ and $L_{\text{box}}$ terms are roughly equally weighted).</td>
</tr>
</tbody>
</table>

The multi-task loss function combines the losses of classification and bounding box regression:

$$L = L_{\text{cls}} + L_{\text{box}}$$

$$L({\{p_i}\}, {\{t_i\}}) = \frac{1}{N_{\text{cls}}} \sum_i L_{\text{cls}}(p_i, p_i^*) + \frac{\lambda}{N_{\text{box}}} \sum_i p_i^* \cdot L_{1}\text{smooth}(t_i - t_i^*)$$

Source: Lilian Weng
Loss function for Mask R-CNN

\[ \mathcal{L} = \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{box}} + \mathcal{L}_{\text{mask}} \]

\[ \mathcal{L}_{\text{mask}} \] is defined as the average binary cross-entropy loss, only including k-th mask if the region is associated with the ground truth class k.

\[ \mathcal{L}_{\text{mask}} = - \frac{1}{m^2} \sum_{1 \leq i, j \leq m} \left[ y_{ij} \log \hat{y}_{ij}^k + (1 - y_{ij}) \log(1 - \hat{y}_{ij}^k) \right] \]
Mask R-CNN training process
Mask R-CNN architecture
Mask R-CNN architecture: FPN
Mask R-CNN mask decoding:
Mask R-CNN architecture: FPN
Detection across multiple datasets

Zhou et al, "Simple multi-dataset detection", 2021