Instance Segmentation

Models
About Me

- 1st year PhD Student
- Image and Video Computing Group
- Undergrad at Western Washington University
- Interned for a year at PNNL
- Features are cool!
- Skiing, running, etc.

Upper Left: credit Morgan Sjogren
Lower image: https://facultyweb.cs.wwu.edu/~wehrwes/semantic_pixels/
Upper Right: https://facultyweb.cs.wwu.edu/~jagodzf/publications.html
Review: What is Instance Segmentation?

Detect instances, give category, label pixels

“Simultaneous detection and segmentation”
Overview

• Faster R-CNN Recap
• Mask R-CNN
• SWIN Transformer
• Discussion
Overview

- Faster R-CNN Recap
- Mask R-CNN
- SWIN Transformer
- Discussion
Faster R-CNN: Architecture Recap

Composed of a region proposal network and a Fast R-CNN classifier.

Main novelty was using RPN instead of selective search.
**Faster R-CNN: ROI Pooling Motivation**

**Want:** Fixed sized feature maps for classification

**Challenge:** Proposals can be of different sizes

Faster R-CNN: ROI Pooling Mechanics

Divide the feature map into a **quantized grid** of $k \times k$ pixels then do max pool

**Quantized:** \[
\left\lfloor \frac{\text{height/width}}{k} \right\rfloor
\]

This will be the size of our grid cell

Source: https://deepsense.ai/region-of-interest-pooling-explained/
Faster R-CNN: ROI Pooling Mechanics

Example: We have a $5 \times 7$ region and want a $2 \times 2$ output

**Dimensions of each grid cell?**

\[
\begin{bmatrix}
\text{height/width} \\
\hline
k
\end{bmatrix}
\]

Source: https://deepsense.ai/region-of-interest-pooling-explained/
Example: We have a $5 \times 7$ region and want a $2 \times 2$ output.

Dimension of each grid cell?

$$\left\lfloor \frac{5}{2} \right\rfloor, \left\lfloor \frac{7}{2} \right\rfloor = (2,3)$$

Have to adjust if proposal dimensions are uneven.
Faster R-CNN: ROI Pooling Recap

Full example

- Quantize
- Divide into $2 \times 2$ grid
- Pool

Source: https://deepsense.ai/region-of-interest-pooling-explained/
Faster R-CNN: Limitation

What if we want pixel-wise annotations?
Overview

• Faster R-CNN Recap
• Mask R-CNN
• SWIN Transformer
• Discussion
Mask R-CNN

Faster R-CNN but make it pixel accurate

Source: https://medium.com/@umerfarooq_26378/from-r-cnn-to-mask-r-cnn-d6367b196cfd
Mask R-CNN: Main Contributions

1. Add an additional branch to Faster R-CNN
2. ROI Align
3. Learn mask in parallel
4. Simple and end-to-end

Source: https://towardsdatascience.com/instance-segmentation-with-mask-r-cnn-6e5c4132030b
Mask R-CNN: A Natural Extension

Mask R-CNN = Faster R-CNN + FCN

Inspiration: https://towardsdatascience.com-instance-segmentation-with-mask-r-cnn-6e5c4132030b
Mask R-CNN: ROI Alignment Motivation

We do not want quantization for pixel-wise accuracy

Why?

More sensitive to perturbations

RPN regress a bounding box

Proposal can have floating point coordinates

We don’t know the value at floating point coordinates

Losing data —> losing precision

Slides adopted from https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align
Mask R-CNN: ROI Alignment Motivation

Suppose we map a image to a 16x16xd feature map and the proposal has floating point coordinates, want a 3x3 output
Quantization causes us to lose data!

Green = info gained

Dark Blue = info lost

Light Blue = info lost if enforcing equal partitions

Slides adopted from https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align
Mask R-CNN: ROI Alignment Mechanics

- Quantize without floor
- Divide into 3x3 grid
- Sample $n$ points in each cell
- Bilinear interpolation
- Repeat

Slides adopted from https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align
Mask R-CNN: ROI Alignment Mechanics

Max pool into output map from newly computed values

3x3 RoIAlign

Slides adopted from https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align
Mask R-CNN: ROI Alignment Mechanics

Now we do not lose data but we do pick up extra information

Green = info gained
Dark Blue = info lost
Light Blue = info lost if enforcing equal partitions

Slides adopted from https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align
Mask R-CNN: ROI Alignment Results

ROI Align gives gains to bounding box AP when used with Faster R-CNN

Mask R-CNN sees further benefits from MTL and backbone

<table>
<thead>
<tr>
<th>backbone</th>
<th>AP_{bb}</th>
<th>AP_{50}</th>
<th>AP_{75}</th>
<th>AP_{S}</th>
<th>AP_{M}</th>
<th>AP_{L}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster R-CNN+++ [19]</td>
<td>34.9</td>
<td>55.7</td>
<td>37.4</td>
<td>15.6</td>
<td>38.7</td>
<td>50.9</td>
</tr>
<tr>
<td>Faster R-CNN w FPN [27]</td>
<td>36.2</td>
<td>59.1</td>
<td>39.0</td>
<td>18.2</td>
<td>39.0</td>
<td>48.2</td>
</tr>
<tr>
<td>Faster R-CNN by G-RMI [21]</td>
<td>34.7</td>
<td>55.5</td>
<td>36.7</td>
<td>13.5</td>
<td>38.1</td>
<td>52.0</td>
</tr>
<tr>
<td>Faster R-CNN w TDM [39]</td>
<td>36.8</td>
<td>57.7</td>
<td>39.2</td>
<td>16.2</td>
<td>39.8</td>
<td><strong>52.1</strong></td>
</tr>
<tr>
<td>Faster R-CNN, RoIAlign</td>
<td>37.3</td>
<td>59.6</td>
<td>40.3</td>
<td>19.8</td>
<td>40.2</td>
<td>48.8</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>38.2</td>
<td>60.3</td>
<td>41.7</td>
<td>20.1</td>
<td>41.1</td>
<td>50.2</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td><strong>39.8</strong></td>
<td><strong>62.3</strong></td>
<td><strong>43.4</strong></td>
<td><strong>22.1</strong></td>
<td><strong>43.2</strong></td>
<td>51.2</td>
</tr>
</tbody>
</table>

Mask R-CNN: Mask Branch

Mask R-CNN = Faster R-CNN + FCN

Inspiration: https://towardsdatascience.com-instance-segmentation-with-mask-r-cnn-6e5c4132030b
Mask R-CNN: Mask Head

Head outputs $K \times m \times m$ binary masks

Use **sigmoid** output layer instead of softmax

Multi-class vs multi-label

Inspiration: https://towardsdatascience.com/instance-segmentation-with-mask-r-cnn-6e5c4132030b
Mask R-CNN: Sigmoid Activation

$$sigmoid(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$$

As $$x \to \infty$$, $$\sigma(x) \to 1$$

As $$x \to -\infty$$, $$\sigma(x) \to 0$$

$$\sigma(x) \in [0,1]$$

Sigmoid can be thought of as the **binary version** of softmax / softmax is a **generalization** of sigmoid for more than 2 classes
Mask R-CNN: Multi-label vs Multi-Class

Multi-class: Each sample has only one label

Multi-label: Each sample can have multiple labels

How does this fit in to the instance and semantic segmentation?
Mask R-CNN: Multi-label vs Multi-Class

In **semantic segmentation** we want to get the most probable class for a pixel, so we use **softmax** to create a probability distribution over all classes for that pixel.

In **instance segmentation** the mask is not responsible for the classification so every pixel in every mask can have an object in it so we use a **sigmoid** to create a distribution for each pixel.

$$\text{pixel}_{i,j} = [x_1, x_2, x_3, x_4, x_5]$$

- **Semantic**
  - Softmax on whole vector
  - $[0.2, 0.1, 0.3, 0.15, 0.25]$

- **Instance**
  - Sigmoid on each element of the vector
  - $[1, 0, 0, 1, 1]$
Mask R-CNN: Mask Head

We only care about the binary mask corresponding to the predicted class

Separates mask and class prediction

Inspiration: https://towardsdatascience.com/instance-segmentation-with-mask-r-cnn-6e5c4132030b
Mask R-CNN: Backbones Recap

- ResNet
- Feature Pyramid Network
Mask R-CNN: Backbones - ResNet

\[ \mathcal{F}(x) \]

\[ \mathcal{F}(x) + x \]

 relu

 weight layer

 x

 identity
Mask R-CNN: Backbones - ResNet

- Only uses ResNet up to stage 4
- Why would you want earlier features?
Mask R-CNN: Backbones - FPN

Allows different ROI scales to help with scale invariance

Image: https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c
Mask R-CNN: Training

Algorithm Training

Repeat until stopping criterion met:
1. **Forward pass**: propagate training data through model to make prediction
2. Quantify the dissatisfaction with a model’s results on the training data
3. **Backward pass**: using predicted output, calculate gradients backward to assign blame to each model parameter
4. Update each parameter using calculated gradients

Key question: how to quantify dissatisfaction?
Mask R-CNN: Loss

Multi-task loss

\[ L = L_{\text{class}} + L_{\text{box}} + L_{\text{mask}} \]

Mask R-CNN: Class Loss

- \( L = L_{\text{class}} + L_{\text{box}} + L_{\text{mask}} \)
- Negative log likelihood/cross entropy

What is the range of possible values?
- Minimum: 0 (negative log of 1)
- Maximum: Infinity (negative log of 0)

Greater penalty when predicted probability of true class is confidently wrong

Lesser penalty otherwise

Slide from Danna Gurari: [https://home.cs.colorado.edu/~DrG/Courses/RecentAdvancesInComputerVision/Lectures/07-ObjectDetection-Part2.pdf](https://home.cs.colorado.edu/~DrG/Courses/RecentAdvancesInComputerVision/Lectures/07-ObjectDetection-Part2.pdf)
Mask R-CNN: Box Loss

- $L = L_{\text{class}} + L_{\text{box}} + L_{\text{mask}}$
- Smooth $\ell_1$ loss

$$L_{\text{box}}(t^u, v) = \sum_{i\in\{x,y,w,h\}} L_{\text{smooth}}^1(t^u_i - v_i)$$

True location for true class “u”
Predicted location for class u
Less sensitive to outliers than SSE

$L_{\text{smooth}}^1(x) = \begin{cases} 
0.5x^2 & \text{if } |x| < 1 \\
|x| - 0.5 & \text{otherwise}
\end{cases}$

Slide from Danna Gurari: https://home.cs.colorado.edu/~DrG/Courses/RecentAdvancesInComputerVision/Lectures/07-ObjectDetection-Part2.pdf
Mask R-CNN: Mask Loss

- \( L = L_{\text{class}} + L_{\text{box}} + L_{\text{mask}} \)

- Binary cross entropy for the \( k^{th} \) mask corresponding to ground truth class \( k \)

- \( \hat{y}_{ij}^k = \sigma(y_{ij}^k) \in [0,1] \quad y_{ij} \in \{0,1\} \)

\[
L_{\text{mask}} = -\frac{1}{m^2} \sum_{1 \leq i,j \leq m} [y_{ij} \log(\hat{y}_{ij}^k) + (1 - y_{ij}) \log(1 - \hat{y}_{ij}^k)]
\]

Total number of pixels

Binary Cross entropy per class!

Loss simplifies depending on the ground truth value

\[
\text{Loss}(\hat{y}^k_{ij}) = \begin{cases} 
-\log(y^k_{ij}), & \text{if } y_{ij} = 1 \\
-\log(1 - \hat{y}^k_{ij}), & \text{if } y_{ij} = 0
\end{cases}
\]
Mask R-CNN: Results

We can achieve SOTA results with a simple extension

<table>
<thead>
<tr>
<th></th>
<th>backbone</th>
<th>AP</th>
<th>AP_{50}</th>
<th>AP_{75}</th>
<th>AP_S</th>
<th>AP_M</th>
<th>AP_L</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: Bonus!

We get pose estimation for free!
Overview

- Faster R-CNN
- Mask R-CNN
- SWIN Transformer
- Discussion
Swin Transformer: Motivation

- Previous work (ViT) great for image recognition not as a general backbone
- Motivation: General purpose backbone from a transformer

Image: https://medium.com/analytics-vidhya/how-to-select-the-perfect-cnn-back-bone-for-object-detection-a-simple-test-b3f9e9519174

Image: https://medium.com/analytics-vidhya/vision-transformers-bye-bye-convolutions-e929d022e4ab
• Transformers are inefficient for image data due to multiple scales
• Self-attention runs in $O(n^2)$
Swin Transformer: Contributions

• General purpose backbone for vision transformers
• More sensible attention algorithm
• Hierarchical patch embeddings

Swin Transformer: Architecture

Composed of stages of patch merging and successive transformer blocks
Swin Transformer: Patch Partition

Let’s just start with a finer resolution and get more coarse as we go further.

\[
\frac{H}{W} / 4 \quad \text{resolution instead of} \quad \frac{H}{W} / 16 \quad \text{resolution}
\]

(a) Swin Transformer (ours)  (b) ViT

Swin Transformer: Architecture

Composed of stages of patch merging and successive transformer blocks

Swin Transformer: Patch Merging

Merge 2x2 grid patches and then double the channels through a linear layer.

Allows representations at different resolutions like FPNs.
**Swin Transformer: Architecture**

Composed of stages of patch merging and successive transformer blocks.
**Swin Transformer: Architecture**

Composed of

1. Layer norm

2. \{Window | Sliding Window\} MSA

3. Skip Connections

4. MLP

Swin Transformer: W-MSA

**Typical MSA Approach:** For every pixel, attend to every other pixel in the image - expensive!

**Window-MSA Approach:** For every pixel in a local window, attend to every other pixel in that window.
Swin Transformer: SW-MSA

- Take prior windows, shift by \( \left\lfloor \frac{M}{2} \right\rfloor \), where \( M \) = size of window
- What to do with the empty space?
Swin Transformer: SW-MSA

- Take prior windows, shift by \( \frac{M}{2} \), where \( M \) = size of window
- Non MxM patches stitched together cyclically
- **Sliding Window** transformer
- Convolution-esque

Swin Transformer: Architecture

Composed of stages of patch merging and successive transformer blocks

### Swin Transformer: Instance Segmentation

Experiments ran on COCO 2017

SOTA on detection and segmentation

---

#### (a) Various frameworks

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>AP\textsubscript{box}</th>
<th>AP\textsubscript{50}</th>
<th>AP\textsubscript{75}</th>
<th>#param.</th>
<th>FLOPs</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cascade</td>
<td>R-50</td>
<td>46.3</td>
<td>64.3</td>
<td>50.5</td>
<td>82M</td>
<td>739G</td>
<td>18.0</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>Swin-T</td>
<td>50.5</td>
<td>69.3</td>
<td>54.9</td>
<td>86M</td>
<td>745G</td>
<td>15.3</td>
</tr>
<tr>
<td>ATSS</td>
<td>R-50</td>
<td>43.5</td>
<td>61.9</td>
<td>47.0</td>
<td>32M</td>
<td>205G</td>
<td>28.3</td>
</tr>
<tr>
<td></td>
<td>Swin-T</td>
<td>47.2</td>
<td>66.5</td>
<td>51.3</td>
<td>36M</td>
<td>215G</td>
<td>22.3</td>
</tr>
<tr>
<td>RepPointsV2</td>
<td>R-50</td>
<td>46.5</td>
<td>64.6</td>
<td>50.3</td>
<td>42M</td>
<td>274G</td>
<td>13.6</td>
</tr>
<tr>
<td></td>
<td>Swin-T</td>
<td>50.0</td>
<td>68.5</td>
<td>54.2</td>
<td>45M</td>
<td>283G</td>
<td>12.0</td>
</tr>
<tr>
<td>Sparse</td>
<td>R-50</td>
<td>44.5</td>
<td>63.4</td>
<td>48.2</td>
<td>106M</td>
<td>166G</td>
<td>21.0</td>
</tr>
<tr>
<td>R-CNN</td>
<td>Swin-T</td>
<td>47.9</td>
<td>67.3</td>
<td>52.3</td>
<td>110M</td>
<td>172G</td>
<td>18.4</td>
</tr>
</tbody>
</table>

---

#### (b) Various backbones w. Cascade Mask R-CNN

<table>
<thead>
<tr>
<th>Backbone</th>
<th>AP\textsubscript{box}</th>
<th>AP\textsubscript{50}</th>
<th>AP\textsubscript{75}</th>
<th>AP\textsubscript{mask}</th>
<th>AP\textsubscript{50}</th>
<th>AP\textsubscript{75}</th>
<th>#param.</th>
<th>FLOPs</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeiT-S</td>
<td>48.0</td>
<td>67.2</td>
<td>51.7</td>
<td>41.4</td>
<td>64.2</td>
<td>44.3</td>
<td>80M</td>
<td>889G</td>
<td>10.4</td>
</tr>
<tr>
<td>R50</td>
<td>46.3</td>
<td>64.3</td>
<td>50.5</td>
<td>40.1</td>
<td>61.7</td>
<td>43.4</td>
<td>82M</td>
<td>739G</td>
<td>18.0</td>
</tr>
<tr>
<td>Swin-T</td>
<td>50.5</td>
<td>69.3</td>
<td>54.9</td>
<td>43.7</td>
<td>66.6</td>
<td>47.1</td>
<td>86M</td>
<td>745G</td>
<td>15.3</td>
</tr>
<tr>
<td>X101-32</td>
<td>48.1</td>
<td>66.5</td>
<td>52.4</td>
<td>41.6</td>
<td>63.9</td>
<td>45.2</td>
<td>101M</td>
<td>819G</td>
<td>12.8</td>
</tr>
<tr>
<td>Swin-S</td>
<td>51.8</td>
<td>70.4</td>
<td>56.3</td>
<td>44.7</td>
<td>67.9</td>
<td>48.5</td>
<td>107M</td>
<td>838G</td>
<td>12.0</td>
</tr>
<tr>
<td>X101-64</td>
<td>48.3</td>
<td>66.4</td>
<td>52.3</td>
<td>41.7</td>
<td>64.0</td>
<td>45.1</td>
<td>140M</td>
<td>972G</td>
<td>10.4</td>
</tr>
<tr>
<td>Swin-B</td>
<td>51.9</td>
<td>70.9</td>
<td>56.5</td>
<td>45.0</td>
<td>68.4</td>
<td>48.7</td>
<td>145M</td>
<td>982G</td>
<td>11.6</td>
</tr>
</tbody>
</table>

---

Overview

• Faster R-CNN
• Mask R-CNN
• Transformer Background
• SWIN Transformer
• Discussion (and Questions?)