Tensor Mask: A Foundation for Dense Object Segmentation

Xinlei Chen, Ross Girshick, Kaiming He, Piotr Dollar
Facebook AI Research (FAIR)

Presenter: Mahavir Dwivedi
Faculty: Yogesh Rawat
Overview

- Problem Statement
- Related Work
- Tensor Representation for Masks
- Tensor Mask Architecture
- Training
- Losses
- Results
- Implementation
Problem Statement

Instance Segmentation Task

- Label each foreground pixel with object and instance
- Object detection + semantic segmentation
Image classification

- boat
- person

Object detection

- boat: 0.853
- person: 0.972
- person: 0.981
- person: 0.907
- person: 0.993

Source: PASCAL Dataset
Semantic Segmentation

Semantic segmentation
(pixel-level classification)
The Instance Segmentation Task

Our task

Semantic segmentation
(pixel-level classification)

Instance segmentation
(pixel-level detection)
Related Work

Detect then Segment (MaskRCNN)

- **BackBone + RPN**
- **Parallel heads for box regression and classification**
- **RoI Align**
Related Work

Label Pixels then Cluster

SGN: Sequential Grouping Networks for Instance Segmentation
Related Work

Dense Sliding Window methods

Learning to Segment Object Candidates Deep Mask
Related Work

Results from DeepMAsk
Why Tensor Mask?

- CNN’s we have (C, H, W) information (3D Tensors)
Why Tensor Mask?

- H, W axis has a geometric meaning as it represents object position
Why Tensor Mask?

• Channel Axis does not have a geometric meaning (has semantic meaning)
Why Tensor Mask?

- Channel Axis, Difficult to manipulate
Why Tensor Mask?

• Tensor Mask proposes 4D Tensors (V,U,H,W)
• We convert channel axes to a new dimension (V,U)
Image Pyramid
Feature Pyramid

Feature Pyramid Network
Tensor Representation for Masks

$(V,U,H,W)$

- Higher dimensional (4D) tensor with shape $(V,U,H,W)$.
- Sub-tensor $(V,U)$ represents a mask as a 2D spatial entity.
- Use these high dimensional Tensors to represent image content.
- Dense set of sliding windows.
Tensor Representation for Masks

Natural Representation
Tensor Representation for Masks

Aligned Representation
Tensor Representation for Masks

\[ F(v, u, y, x) = \hat{F}(v, u, y + \alpha v, x + \alpha u). \]

Coordinate Transformation
Tensor Representation for Masks

Upscale Transformation \((\Lambda)\)

- Coarse\((\hat{V}, \hat{U})\) sub-tensors to create Finer\((V, U)\) sub-tensors
- Generates high-resolution masks without inflating channel counts in preceding feature map.
Tensor Representation for Masks

Tensor BiPyramid

- Number of mask pixels based on scale
- List containing masks
- Shape of Tensors in the list:

\[(2^k V, 2^k U, \frac{1}{2^k} H, \frac{1}{2^k} W)\]
Tensor Mask Architecture

1. **Mask Prediction Head**
   Generate Masks in Sliding window

1. **Classification Head**
   Predicts object categories

   Similar to bounding box detection
Tensor Mask Architecture

Mask Prediction Head
Generate Masks in Sliding window

$C, H, W \xrightarrow{\text{conv+reshape}} V, U, H, W$
Tensor Mask Architecture

Mask Prediction Head
Generate Masks in Sliding window

\[ C, H, W \xrightarrow{\text{conv+reshape}} \hat{V}, \hat{U}, \hat{H}, \hat{W} \xrightarrow{\text{align2nat}} V, U, H, W \]
Tensor Mask Architecture

Mask Prediction Head
Generate Masks in Sliding window

\[ \frac{1}{\lambda} V, \frac{1}{\lambda} U, H, W \]

Conv + Reshape

\[ V, U, H, W \]

Up_bilinear
Tensor Mask Architecture

Mask Prediction Head
Generate Masks in Sliding window

\[ C, H, W \xrightarrow{\text{conv+reshape}} \frac{1}{\lambda} \hat{V}, \frac{1}{\lambda} \hat{U}, \hat{H}, \hat{W} \xrightarrow{\text{up_align2nat}} V, U, H, W \]
Tensor Mask Architecture

Mask Prediction Head
Generate Masks in Sliding window

\[ C, \frac{1}{4}H, \frac{1}{4}W \rightarrow V, U, \frac{1}{4}H, \frac{1}{4}W \]
\[ \sigma_{VU} = 4s \]
\[ \sigma_{HW} = 4s \]

\[ C, \frac{1}{2}H, \frac{1}{2}W \rightarrow V, U, \frac{1}{2}H, \frac{1}{2}W \]
\[ \sigma_{VU} = 2s \]
\[ \sigma_{HW} = 2s \]

\[ C, H, W \rightarrow V, U, H, W \]
\[ \sigma_{VU} = s \]
\[ \sigma_{HW} = s \]
Tensor Mask Architecture

Mask Prediction Head
Generate Masks in Sliding window

```
swap_align2nat head
```

- $C, H, W \rightarrow 4V, 4U, \frac{1}{4}H, \frac{1}{4}W$
  - $\sigma_{VU} = s$
  - $\sigma_{HW} = 4s$
- $C, H, W \rightarrow 2V, 2U, \frac{1}{2}H, \frac{1}{2}W$
  - $\sigma_{VU} = s$
  - $\sigma_{HW} = 2s$
- $C, H, W \rightarrow V, U, H, W$
  - $\sigma_{VU} = s$
  - $\sigma_{HW} = s$
Tensor Mask Architecture

Mask Prediction Head
Generate Masks in Sliding window
Label Assignment for ground truth mask ("m"): 

- Containment

  Window fully contains "m" and the longer side of "m"
Label Assignment:

- Centrality

Center of “m” bounding box is within one unit ($\sigma_{v_i}$) of the window center in L2 distance
Label Assignment:

- Uniqueness

There is no other mask $m' = m$ that satisfies the other two conditions
Tensor Mask : Losses

- For prediction of mask head:
  Per Pixel Binary classification Loss

- For box regression:
  L1 Loss

- Classification head:
  Focal Loss
Dataset: Coco

http://cocodataset.org/#detection-eval
Results

Natural up-scaling head
Results

Unaligned head
Results

Aligned head
## Results

<table>
<thead>
<tr>
<th>head</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>natural</td>
<td>28.5</td>
<td>52.2</td>
<td>28.6</td>
<td>14.4</td>
<td>30.2</td>
<td>40.1</td>
</tr>
<tr>
<td>aligned</td>
<td>28.9</td>
<td>52.5</td>
<td>29.3</td>
<td>14.6</td>
<td>30.8</td>
<td>40.7</td>
</tr>
</tbody>
</table>

Simple heads: natural vs. aligned with $V \times U = 15 \times 15$
### Results

<table>
<thead>
<tr>
<th>head</th>
<th>λ</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>Δ aligned - natural</th>
</tr>
</thead>
<tbody>
<tr>
<td>natural</td>
<td>1.5</td>
<td>28.0</td>
<td>51.7</td>
<td>27.8</td>
<td>+0.9 +0.7 +1.5</td>
</tr>
<tr>
<td>aligned</td>
<td>1.5</td>
<td>28.9</td>
<td>52.4</td>
<td>29.3</td>
<td></td>
</tr>
<tr>
<td>natural</td>
<td>3</td>
<td>24.7</td>
<td>48.4</td>
<td>22.7</td>
<td>+4.1 +3.9 +6.4</td>
</tr>
<tr>
<td>aligned</td>
<td>3</td>
<td>28.8</td>
<td>52.3</td>
<td>29.1</td>
<td></td>
</tr>
<tr>
<td>natural</td>
<td>5</td>
<td>19.2</td>
<td>42.1</td>
<td>15.6</td>
<td>+9.2 +9.7 +13.0</td>
</tr>
<tr>
<td>aligned</td>
<td>5</td>
<td>28.4</td>
<td>51.8</td>
<td>28.6</td>
<td></td>
</tr>
</tbody>
</table>

Upscaling heads: Natural vs. aligned heads (V×U=15×15)
## Results

<table>
<thead>
<tr>
<th>head</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>feature pyramid, best</td>
<td>28.9</td>
<td>52.5</td>
<td>29.3</td>
<td>14.6</td>
<td>30.8</td>
<td>40.7</td>
</tr>
<tr>
<td>tensor bipyramid</td>
<td>34.0</td>
<td>55.2</td>
<td>35.8</td>
<td>15.3</td>
<td>36.3</td>
<td>48.4</td>
</tr>
<tr>
<td>Δ</td>
<td>+5.1</td>
<td>+2.7</td>
<td>+6.5</td>
<td>+0.7</td>
<td>+5.5</td>
<td>+7.7</td>
</tr>
</tbody>
</table>

Beast Baseline head on Feature Pyramid vs Tensor bipyramid
## Results

<table>
<thead>
<tr>
<th>$V \times U$</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>15\times15</td>
<td>34.0</td>
<td>55.2</td>
<td>35.8</td>
<td>15.3</td>
<td>36.3</td>
<td>48.4</td>
</tr>
<tr>
<td>15\times15, 11\times11</td>
<td>35.2</td>
<td>56.4</td>
<td>37.0</td>
<td>17.4</td>
<td>37.4</td>
<td>49.7</td>
</tr>
<tr>
<td>$\Delta$</td>
<td>+1.2</td>
<td>+1.2</td>
<td>+1.2</td>
<td>+2.1</td>
<td>+1.1</td>
<td>+1.3</td>
</tr>
</tbody>
</table>

Window Sizes
## Results

<table>
<thead>
<tr>
<th>method</th>
<th>backbone</th>
<th>aug</th>
<th>epochs</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>Mask R-CNN [13]</td>
<td>R-50-FPN</td>
<td></td>
<td>24</td>
<td>34.9</td>
<td>57.2</td>
<td>36.9</td>
<td>15.4</td>
<td>36.6</td>
<td>50.8</td>
</tr>
<tr>
<td>Mask R-CNN, ours</td>
<td>R-50-FPN</td>
<td></td>
<td>24</td>
<td>34.9</td>
<td>56.8</td>
<td>36.8</td>
<td>15.1</td>
<td>36.7</td>
<td>50.6</td>
</tr>
<tr>
<td>Mask R-CNN, ours</td>
<td>R-50-FPN</td>
<td>✓</td>
<td>72</td>
<td>36.8</td>
<td>59.2</td>
<td>39.3</td>
<td>17.1</td>
<td>38.7</td>
<td>52.1</td>
</tr>
<tr>
<td>TensorMask</td>
<td>R-50-FPN</td>
<td>✓</td>
<td>72</td>
<td>35.4</td>
<td>57.2</td>
<td>37.3</td>
<td>16.3</td>
<td>36.8</td>
<td>49.3</td>
</tr>
<tr>
<td>Mask R-CNN, ours</td>
<td>R-101-FPN</td>
<td>✓</td>
<td>72</td>
<td>38.3</td>
<td>61.2</td>
<td>40.8</td>
<td>18.2</td>
<td>40.6</td>
<td>54.1</td>
</tr>
<tr>
<td>TensorMask</td>
<td>R-101-FPN</td>
<td>✓</td>
<td>72</td>
<td>37.1</td>
<td>59.3</td>
<td>39.4</td>
<td>17.4</td>
<td>39.1</td>
<td>51.6</td>
</tr>
</tbody>
</table>

Comparison with Mask R-CNN
### Results

<table>
<thead>
<tr>
<th>method</th>
<th>aug</th>
<th>epochs</th>
<th>( \text{AP}^{\text{bb}} )</th>
<th>( \text{AP}_{50}^{\text{bb}} )</th>
<th>( \text{AP}_{75}^{\text{bb}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>RetinaNet, <em>ours</em></td>
<td></td>
<td>24</td>
<td>37.1</td>
<td>55.0</td>
<td>39.9</td>
</tr>
<tr>
<td>RetinaNet, <em>ours</em></td>
<td>√</td>
<td>72</td>
<td>39.3</td>
<td>57.2</td>
<td>42.4</td>
</tr>
<tr>
<td>Faster R-CNN, <em>ours</em></td>
<td>√</td>
<td>72</td>
<td>40.6</td>
<td>61.4</td>
<td>44.2</td>
</tr>
<tr>
<td>Mask R-CNN, <em>ours</em></td>
<td>√</td>
<td>72</td>
<td>41.7</td>
<td>62.5</td>
<td>45.7</td>
</tr>
<tr>
<td>TensorMask, <em>box-only</em></td>
<td>√</td>
<td>72</td>
<td>40.8</td>
<td>60.4</td>
<td>43.9</td>
</tr>
<tr>
<td>TensorMask</td>
<td>√</td>
<td>72</td>
<td>41.6</td>
<td>61.0</td>
<td>45.1</td>
</tr>
</tbody>
</table>

Object detection
## Results

<table>
<thead>
<tr>
<th>box head</th>
<th>NMS</th>
<th>AP</th>
<th>AP(_{50})</th>
<th>AP(_{75})</th>
<th>AP(^{bb})</th>
<th>AP(^{bb})</th>
<th>AP(^{bb})</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>bb</td>
<td>35.2</td>
<td>56.4</td>
<td>37.0</td>
<td>41.6</td>
<td>60.8</td>
<td>44.8</td>
</tr>
<tr>
<td>✓</td>
<td>mask-bb</td>
<td>34.9</td>
<td>56.0</td>
<td>36.7</td>
<td>39.7</td>
<td>59.1</td>
<td>41.8</td>
</tr>
<tr>
<td>✓</td>
<td>mask-bb</td>
<td>34.8</td>
<td>56.1</td>
<td>36.7</td>
<td>39.4</td>
<td>58.8</td>
<td>41.6</td>
</tr>
</tbody>
</table>

Multi-task benefits
Results

Mask R-CNN (top row) and TensorMask (bottom row)
Results

Mask R-CNN (top row) and TensorMask (bottom row)
Mask R-CNN (top row) and TensorMask (bottom row)
Implementation

TensorMask in Detectron2

A Foundation for Dense Object Segmentation

Xinlei Chen, Ross Girshick, Kaiming He, Puligandla

https://github.com/facebookresearch/detectron2/tree/master/projects/TensorMask
A research platform and a production library for object detection, mainly built by Facebook AI Research (FAIR)

https://github.com/facebookresearch/detectron2
What’s in Detectron2: model zoo

- Different settings for users to play with
- Standard baselines for researchers
- Efficient models for production (coming soon)

---

https://github.com/facebookresearch/detectron2/blob/master/MODEL_ZOO.md
What’s in Detectron2: Generalized R-CNN Models
(+ a few other types of models: RetinaNet, TensorMask, etc.)
Thank You