Focal Loss for Dense Object Detection

Tsung-Yi Lin    Priya Goyal    Ross Girshick    Kaiming He    Piotr Dollar

Presented by Erik Seetao
Detectron

- State of the art object detection presented by Facebook’s AI team
- Provides high quality, high performance codebase for object detection
- Includes:
  - Focal Loss for Dense Object Detection
  - Mask R-CNN
  - Fast R-CNN
  - Feature Pyramid Network for Object Detection
Problem Statement

- Two-stage approach
  - Best object detectors based on R-CNN
  - Classifier is applied to a sparse set of candidate object locations
- One-stage approach
  - Applied over a regular, dense sampling of possible object locations
  - Faster and simpler, but worse accuracy than two-stage approach
- Extreme foreground/background class imbalance encountered during training of dense detectors causes this
- We don’t want our training procedure to be dominated by easily classified background examples
Objective

- Address the class imbalance
  - Reshape standard cross entropy loss
  - Down-weight the loss assigned to well-classified examples
- Create **Focal Loss** that focuses training on sparse set of hard examples
  - Prevents vast number of easy negatives from overwhelming detector during training
- Benchmark effectiveness by designing and training simple dense detector **RetinaNet**
  - Should match speed of one-stage with better accuracy than two-stage
R-CNN

- Regions with Convolutional Neural Network Features
- Two-stage approach
  - First stage: generates a sparse set of candidate object locations
  - Second stage: classifies each candidate location as a foreground or background classes using CNN
- Rapidly narrows down number of candidate object locations to a small number
  - Filters out most background samples
  - Sampling heuristics like Online Hard Example Mining (OHEM) used to manage balance between foreground and background
**Focal Loss**

- Addresses one-stage object detection with imbalance between foreground and background
- Introduced from cross entropy loss for binary classification
  - Measures the performance of a classification model's output is a probability value between 0 and 1
  - Add a weighting factor $\alpha$ to address class imbalance
- Creates balanced cross entropy used as a baseline for one-stage Focal Loss

$$CE(p_t) = -\alpha_t \log(p_t)$$
Focal Loss

- Add a modulating factor to cross entropy loss and tunable focusing parameter $\gamma$
- Focal Loss defined as:

$$FL(p_t) = -(1 - p_t)^\gamma \log(p_t)$$

- When an example is misclassified and $p_t$ is small, the modulating factor is near 1 and the loss is unaffected
- As $p_t$ approaches 1, the factor goes to 0 and the loss for well-classified examples is down-weighted.
RetinaNet Detector

- Single, unified network composed of a backbone network and two task-specific subnetworks
- Backbone:
  - Responsible for computing a convolutional feature map over an entire input image
- Two task-specific subnetworks:
  - First subnet performs convolutional object classification on backbone's output
  - Second subnet performs convolutional bounding box regression
- Two subnetworks will feature design for one-stage dense object detection
RetinaNet Detector

● Adopt Feature Pyramid Network (FPN)
  ○ FPN augments a standard CNN with top-down pathway
  ○ Network efficiently constructs a multi-scale feature pyramid from a single resolution input image
● Each level of the pyramid can be used for detecting objects at a different scale
RetinaNet Detector

- Classification Subnet
  - Predicts probability of object at each spatial position (K object classes, A anchors)
  - Takes an input feature map with C channels from a given pyramid level, applies four $3\times3$ conv layers, each followed by ReLU activations, followed by a $3\times3$ conv layer with K A filters

- Box Regression Subnet
  - Is another small FCN to each pyramid level, regresses the offset from each anchor box to a object
  - Similar structure to classification subnet but different parameters
Training

- When training RetinaNet, Focal Loss is applied to all ~100k anchors in each sampled image
- Uses ResNet-50-FPN and ResNet-101-FPN backbone
- RetinaNet is trained with stochastic gradient descent
  - Synchronized over 8 GPUs with a total of 16 images per minibatch (2 images per GPU)
  - Unless otherwise specified, all models are trained for 90k iterations with an initial learning rate of 0.01
Results

Accuracy measured by Average Precision (AP)

<table>
<thead>
<tr>
<th>( \alpha )</th>
<th>AP</th>
<th>AP(_{50} )</th>
<th>AP(_{75} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>.10</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>.25</td>
<td>10.8</td>
<td>16.0</td>
<td>11.7</td>
</tr>
<tr>
<td>.50</td>
<td>30.2</td>
<td>46.7</td>
<td>32.8</td>
</tr>
<tr>
<td>.75</td>
<td>31.1</td>
<td>49.4</td>
<td>33.0</td>
</tr>
<tr>
<td>.90</td>
<td>30.8</td>
<td>49.7</td>
<td>32.3</td>
</tr>
<tr>
<td>.99</td>
<td>28.7</td>
<td>47.4</td>
<td>29.9</td>
</tr>
<tr>
<td>.999</td>
<td>25.1</td>
<td>41.7</td>
<td>26.1</td>
</tr>
</tbody>
</table>

(a) Varying \( \alpha \) for CE loss (\( \gamma = 0 \))

<table>
<thead>
<tr>
<th>( \gamma )</th>
<th>( \alpha )</th>
<th>AP</th>
<th>AP(_{50} )</th>
<th>AP(_{75} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>.75</td>
<td>31.1</td>
<td>49.4</td>
<td>33.0</td>
</tr>
<tr>
<td>0.1</td>
<td>.75</td>
<td>31.4</td>
<td>49.9</td>
<td>33.1</td>
</tr>
<tr>
<td>0.2</td>
<td>.75</td>
<td>31.9</td>
<td>50.7</td>
<td>33.4</td>
</tr>
<tr>
<td>0.5</td>
<td>.50</td>
<td>32.9</td>
<td>51.7</td>
<td>35.2</td>
</tr>
<tr>
<td>1.0</td>
<td>.25</td>
<td>33.7</td>
<td>52.0</td>
<td>36.2</td>
</tr>
<tr>
<td>2.0</td>
<td>.25</td>
<td>34.0</td>
<td>52.5</td>
<td>36.5</td>
</tr>
<tr>
<td>5.0</td>
<td>.25</td>
<td>32.2</td>
<td>49.6</td>
<td>34.8</td>
</tr>
</tbody>
</table>

(b) Varying \( \gamma \) for FL (w. optimal \( \alpha \))
## Results

Accuracy measured by Average Precision (AP)

<table>
<thead>
<tr>
<th>method</th>
<th>batch size</th>
<th>rms thr</th>
<th>AP</th>
<th>AP_{50}</th>
<th>AP_{75}</th>
</tr>
</thead>
<tbody>
<tr>
<td>OHEM</td>
<td>128</td>
<td>.7</td>
<td>31.1</td>
<td>47.2</td>
<td>33.2</td>
</tr>
<tr>
<td>OHEM</td>
<td>256</td>
<td>.7</td>
<td>31.8</td>
<td>48.8</td>
<td>33.9</td>
</tr>
<tr>
<td>OHEM</td>
<td>512</td>
<td>.7</td>
<td>30.6</td>
<td>47.0</td>
<td>32.6</td>
</tr>
<tr>
<td>OHEM</td>
<td>128</td>
<td>.5</td>
<td>32.8</td>
<td>50.3</td>
<td>35.1</td>
</tr>
<tr>
<td>OHEM</td>
<td>256</td>
<td>.5</td>
<td>31.0</td>
<td>47.4</td>
<td>33.0</td>
</tr>
<tr>
<td>OHEM</td>
<td>512</td>
<td>.5</td>
<td>27.6</td>
<td>42.0</td>
<td>29.2</td>
</tr>
<tr>
<td>OHEM 1:3</td>
<td>128</td>
<td>.5</td>
<td>31.1</td>
<td>47.2</td>
<td>33.2</td>
</tr>
<tr>
<td>OHEM 1:3</td>
<td>256</td>
<td>.5</td>
<td>28.3</td>
<td>42.4</td>
<td>30.3</td>
</tr>
<tr>
<td>OHEM 1:3</td>
<td>512</td>
<td>.5</td>
<td>24.0</td>
<td>35.5</td>
<td>25.8</td>
</tr>
<tr>
<td>FL</td>
<td>n/a</td>
<td>n/a</td>
<td><strong>36.0</strong></td>
<td><strong>54.9</strong></td>
<td><strong>38.7</strong></td>
</tr>
</tbody>
</table>

(d) **FL** vs. **OHEM** baselines (with ResNet-101-FPN)
## Results

<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><strong>Two-stage methods</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Faster R-CNN+++ [16]</td>
<td>ResNet-101-C4</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 by G-RMI [17]</td>
<td>Inception-ResNet-v2 [34]</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 [32]</td>
<td>Inception-ResNet-v2-TDM</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><strong>One-stage methods</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YOLOv2 [27]</td>
<td>DarkNet-19 [27]</td>
<td>21.6</td>
<td>44.0</td>
<td>19.2</td>
<td>5.0</td>
<td>22.4</td>
<td>35.5</td>
</tr>
<tr>
<td>SSD513 [22, 9]</td>
<td>ResNet-101-SSD</td>
<td>31.2</td>
<td>50.4</td>
<td>33.3</td>
<td>10.2</td>
<td>34.5</td>
<td>49.8</td>
</tr>
<tr>
<td>DSSD513 [9]</td>
<td>ResNet-101-DSSD</td>
<td>33.2</td>
<td>53.3</td>
<td>35.2</td>
<td>13.0</td>
<td>35.4</td>
<td>51.1</td>
</tr>
<tr>
<td>RetinaNet (ours)</td>
<td>ResNet-101-FPN</td>
<td>39.1</td>
<td>59.1</td>
<td>42.3</td>
<td>21.8</td>
<td>42.7</td>
<td>50.2</td>
</tr>
<tr>
<td>RetinaNet (ours)</td>
<td>ResNeXt-101-FPN</td>
<td><strong>40.8</strong></td>
<td><strong>61.1</strong></td>
<td><strong>44.1</strong></td>
<td><strong>24.1</strong></td>
<td><strong>44.2</strong></td>
<td>51.2</td>
</tr>
</tbody>
</table>
Analysis

- For both two-stage and one-stage, the FPN performs better than the other variants
- Focal Loss:
  - CDF is very similar for both foreground and background
  - For positive samples ($0<\gamma<1$), the change on the distribution is minor
  - For negative samples ($\gamma>1$), $\gamma$ concentrates loss on hard samples, which skews away from easy negatives
As expected, RetinaNet outperforms both two-stage and one-stage models. Achieved similar speeds relative to one-stage model with better accuracy than two-stage model. RetinaNet envelopes all current detectors, even surpassing that of Faster R-CNN.
Strengths

- Focal Loss, when trained on RetinaNet, outperforms all current detectors with an impressive $\sim 60AP$
  - Match speeds of one-stage detector
  - Better precision than two-stage detector
- Proposes a new, more effective loss function that deals with class imbalances
Weakness

- Does not address the special case of high frame rate regime
  - Will likely require special network design that is different from RetinaNet
- At the time of publication, a new variant of Faster R-CNN has surpassed Focal Loss
Takeaway Point

- We identify class imbalance as the primary obstacle preventing one-stage object detectors from surpassing top-performing, two-stage methods
- Solve by introducing $\alpha$ and $\gamma$ to prevent easily classified background samples to dominate
Thank you!

Discussion / Q&A