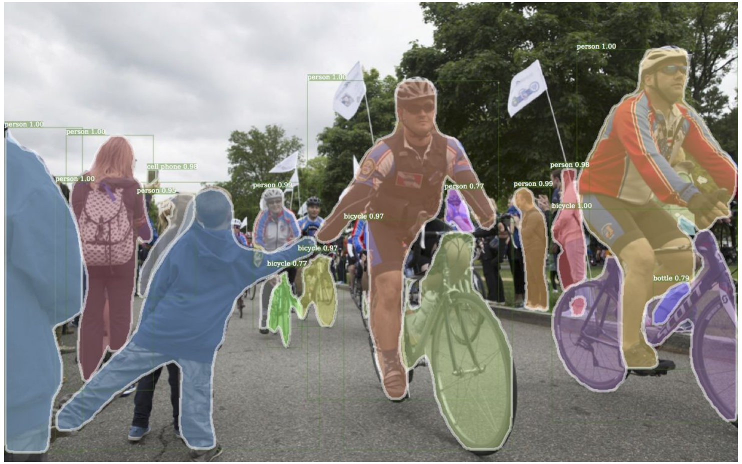
Focal Loss for Dense Object Detection

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 ***α*** 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 **γ**
- Focal Loss defined as:

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

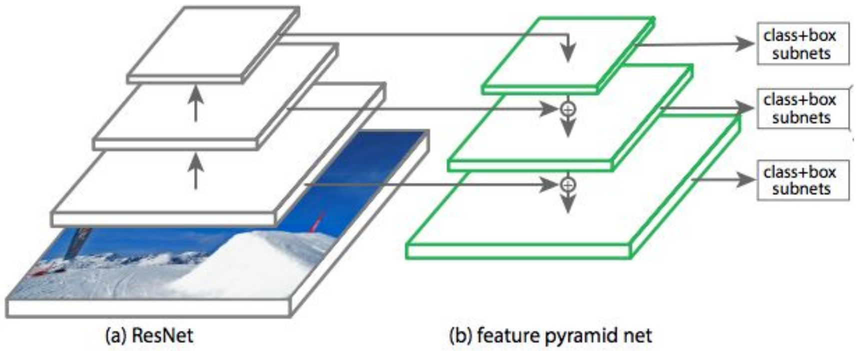
- When an example is misclassified and pt is small, the modulating factor is near 1 and the loss is unaffected
- As pt approaches 1, the factor goes to 0 and the loss for well-classified examples is down-weighted.

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

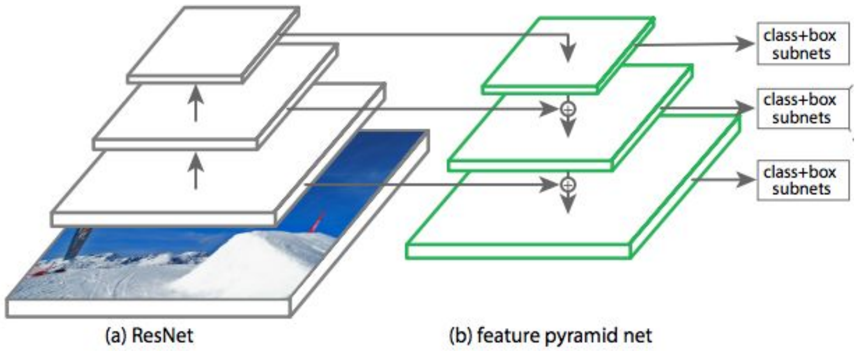
RetinaNet Detector

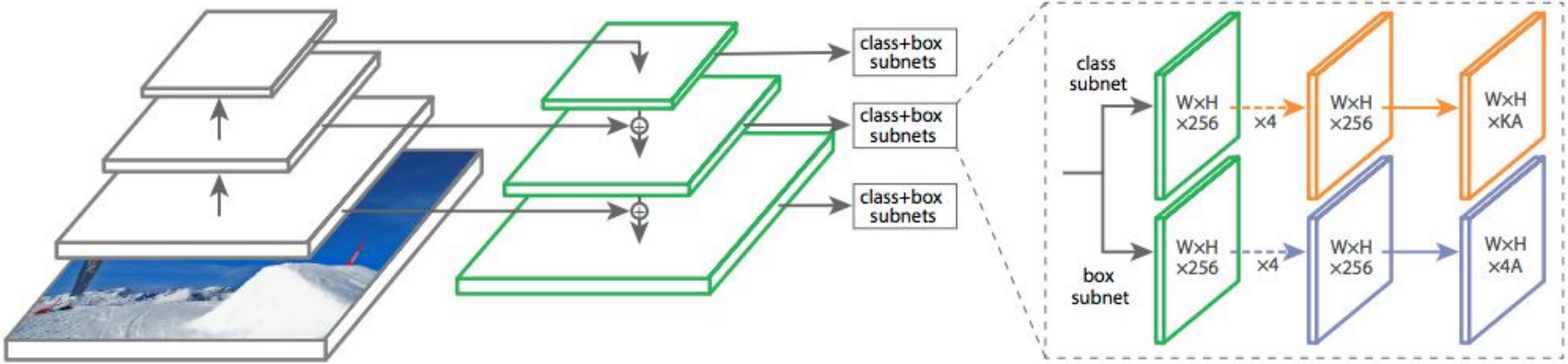
- Adopt Feature Pyramid Network (FPN)
    - FPN augments a standard CNN with top-down pathway
    - Network efficiently constructs a multi-scale feature pyramid from a single resolution input image
- Each level of the pyramid can be used for detecting objects at a different scale



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 3×3 conv layers, each followed by ReLU activations, followed by a 3×3 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





(a) ResNet (b) feature pyramid net (c) class subnet (top) (d) box subnet (bottom)

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

α	AP	AP_{50}	AP_{75}
.10	0.0	0.0	0.0
.25	10.8	16.0	11.7
.50	30.2	46.7	32.8
.75	31.1	49.4	33.0
.90	30.8	49.7	32.3
.99	28.7	47.4	29.9
.999	25.1	41.7	26.1

(a) **Varying α for CE loss** ($\gamma = 0$)

γ	α	AP	AP_{50}	AP_{75}
0	.75	31.1	49.4	33.0
0.1	.75	31.4	49.9	33.1
0.2	.75	31.9	50.7	33.4
0.5	.50	32.9	51.7	35.2
1.0	.25	33.7	52.0	36.2
2.0	.25	**34.0**	**52.5**	**36.5**
5.0	.25	32.2	49.6	34.8

(b) **Varying γ for FL** (w. optimal α)

Accuracy measured by Average Precision (AP)

Results

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

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

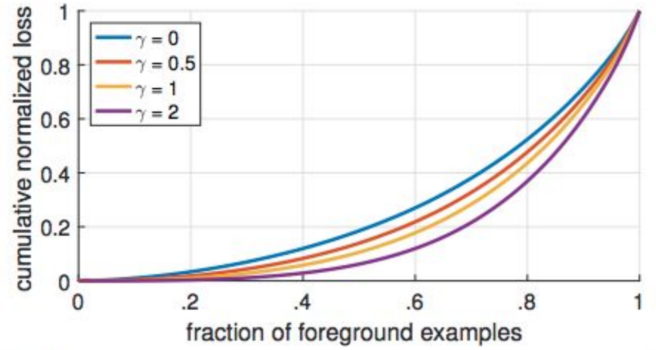
Accuracy measured by Average Precision (AP)

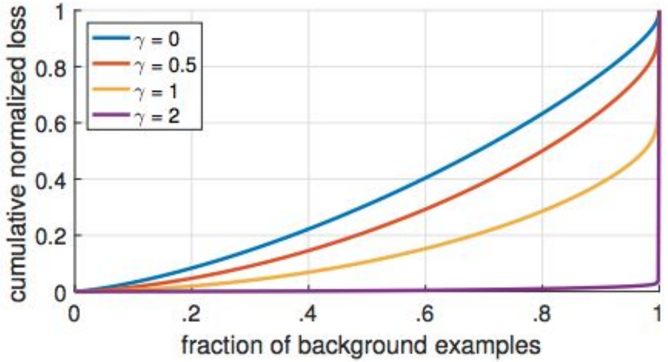
Results

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [16]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [20]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [17]	Inception-ResNet-v2 [34]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [32]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [27]	DarkNet-19 [27]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [22, 9]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [9]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet (ours)	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet (ours)	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2

Analysis

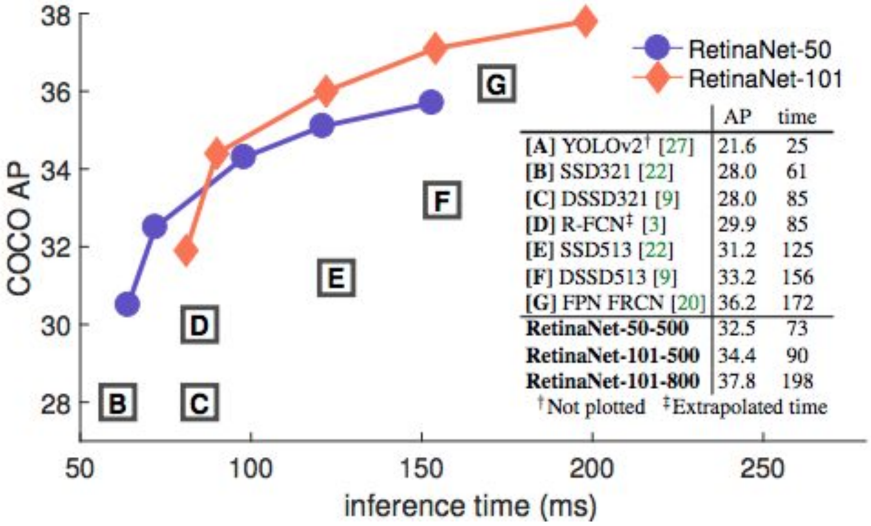
- For both two-stage and one-stage, the FPN performs better than the other variants
- Focal Loss:
    - CDF is very similar for both foreground and background
    - For positive samples (0<γ<1), the change on the distribution is minor
    - For negative samples (γ>1), γ concentrates loss on hard samples, which skews away from easy negatives





Analysis

- 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



	AP	time
[A] YOLOv2† [27]	21.6	25
[B] SSD321 [22]	28.0	61
[C] DSSD321 [9]	28.0	85
[D] R-FCN‡ [3]	29.9	85
[E] SSD513 [22]	31.2	125
[F] DSSD513 [9]	33.2	156
[G] FPN FRCN [20]	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	37.8	198

†Not plotted ‡Extrapolated time

Strengths

- Focal Loss, when trained on RetinaNet, outperforms all current detectors with an impressive ~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 $\boldsymbol{\alpha}$ and γ to prevent easily classified background samples to dominate

Thank you!

Discussion / Q&A