RetinaNet

Recap : R-CNN Family

- R-CNN
- Fast R-CNN
- Faster R-CNN

Analytics Vidhya

Single Stage Algorithms

- YOLO
- SSD
- RetinaNet

Analytics Vidhya

Performance Comparison

	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

Reason for Poor Performance of Single-stage Networks

- Class Imbalance

Background Class	Object Class

Focal Loss

- More importance to object class
- High penalty for misclassifying object class

Analytics Vidhya

Categorical Cross Entropy Loss

It is the negative average of the log of predicted class probabilities

$$LogLoss = \frac{-1}{N} \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} * \ log(p_{ij})$$

- N is the number of rows
- M is the number of classes

Analytics Vidhya

Focal Loss

- More importance to object class

$$LogLoss = \frac{-1}{N} \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} * \alpha_i * log(p_{ij})$$

Focal Loss

- More importance to object class
- High penalty for misclassifying object class

$$LogLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} x_{ij} * \alpha_i * (1 - p_{ij})^{\gamma} * log(p_{ij})$$

Focal Loss

- More importance to object class
- High penalty for misclassifying object class

$$LogLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} x_{ij} * \alpha_i * (1 - p_{ij})^{\gamma} * log(p_{ij})$$

Low probability ⬇️ Loss is unaffected

Focal Loss

- More importance to object class
- High penalty for misclassifying object class

$$LogLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} x_{ij} * \alpha_i * (1 - p_{ij})^{\gamma} * log(p_{ij})$$

High probability ⬆️ Loss is down-weighted

Focal Loss

Example 1:

Misclassified example → p_{ij} = 0.1

$\alpha_i * (1 - p_{ij})^{\gamma} * log(p_{ij})$ = 0.25 * (1-0.1)^2 * log(0.1)

= -0.2025

Analytics Vidhya

Focal Loss

Example 1:

Misclassified example → p_{ij} = 0.1

$\alpha_i * (1 - p_{ij})^{\gamma} * log(p_{ij})$ = 0.25 * (1-0.1)^2 * log(0.1)

= -0.2025

Example 2:

Misclassified example → p_{ij} = 0.9

$\alpha_i * (1 - p_{ij})^{\gamma} * log(p_{ij})$ = 0.25 * (1-0.9)^2 * log(0.9)

= -0.000114

Focal Loss

Example 1:

Misclassified example → p_{ij} = 0.1

$\alpha_i * (1 - p_{ij})^{\gamma} * log(p_{ij})$ = 0.25 * (1-0.1)^2 * log(0.1)

= -0.2025

Low probability

Loss is unaffected

Example 2:

Misclassified example → p_{ij} = 0.9

$\alpha_i * (1 - p_{ij})^{\gamma} * log(p_{ij})$ = 0.25 * (1-0.9)^2 * log(0.9)

= -0.000114

High probability

Loss is down-weighted

RetinaNet Architecture Details

- Backbone Architecture - ResNet

Analytics Vidhya

RetinaNet Architecture Details

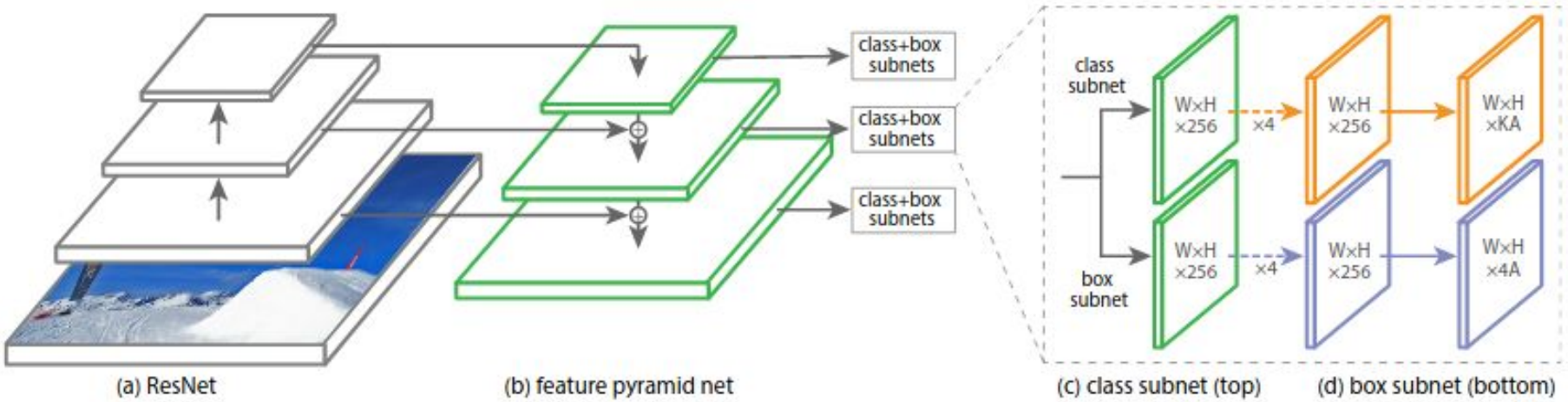
- Backbone Architecture - ResNet
- Uses Feature Pyramid Network for multiscale prediction

Analytics Vidhya

RetinaNet Architecture Details

- Backbone Architecture - ResNet
- Uses Feature Pyramid Network for multiscale prediction
- Anchor boxes of 3 different aspect ratios used

RetinaNet Architecture Details



Analytics Vidhya

RetinaNet Details

	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

Thank You

Analytics Vidhya

Improvements in Single-stage Network

- Build deeper models
- Make predictions at different scales
- Adding Batchnorm, residual block etc
- Capture smaller objects