Airbus Kaggle Competition

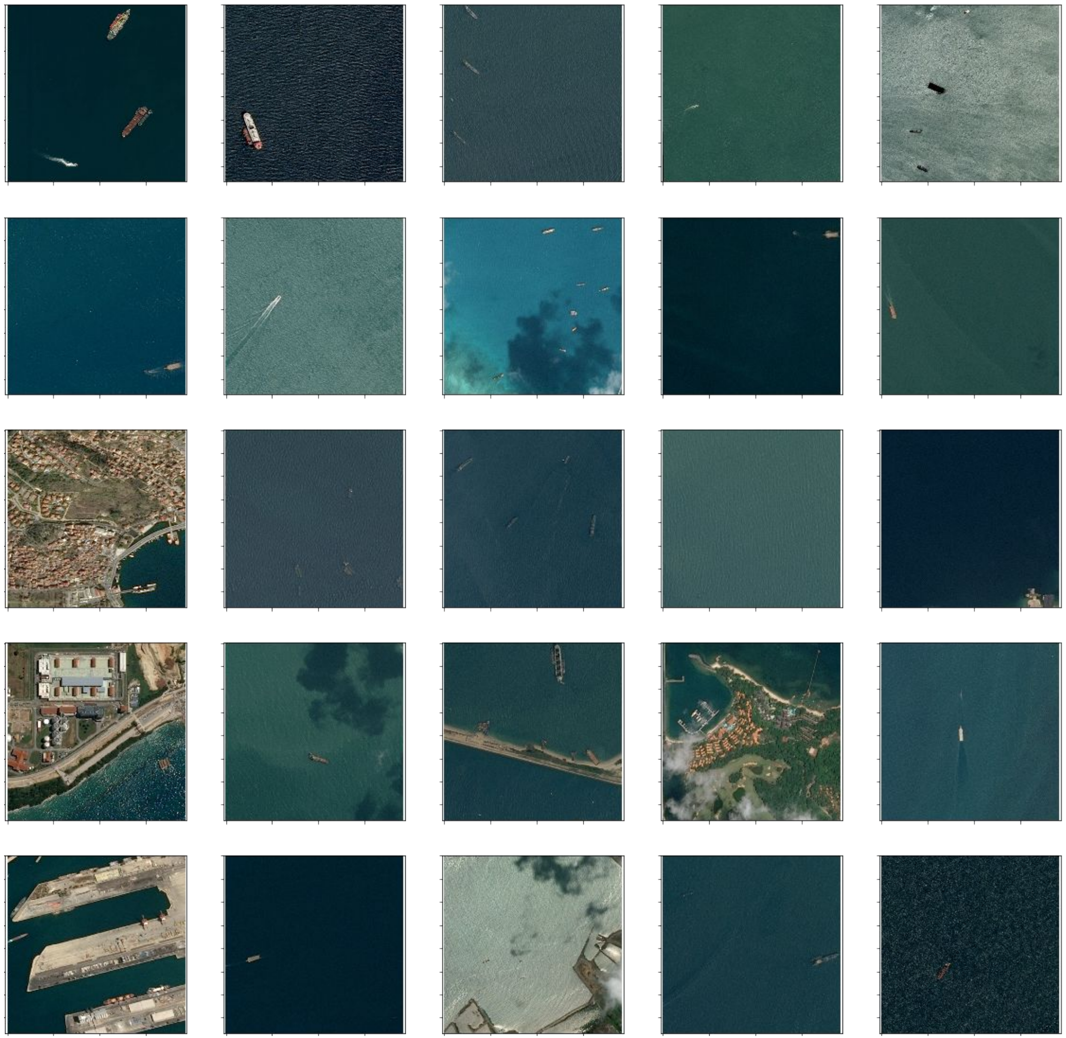
Team: Bruce, Evangelia, Andy, Adam, Adrian, Alan

TASK:

- FOR EACH TEST IMAGE DETECT ALL SHIPS

- FOR EACH SHIP ON THE IMAGE CREATE A SINGLE MASK

- SUBMIT CSV WITH THE MASKS



$ 60,000 PRIZE MONEY

- 1ST PLACE - $25,000
- 2ND PLACE - $15,000
- 3RD PLACE - $5,000
- ALGORITHM SPEED PRIZE (POST COMPETITION PRIZE) - $15,000

884 TEAMS

#	Δpub	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	[ods.ai] Rectangle is all you n...			0.85448	75	13d
2	▲15	[ods.ai] topcoders			0.85433	54	14d
3	▲7	bestfitting			0.85428	146	13d
4	▲22	[attention heads]			0.85411	55	13d
5	▲13	dhammack			0.85350	14	14d
657	▲10	FirstTimeVowels		+3	0.78893	8	14d

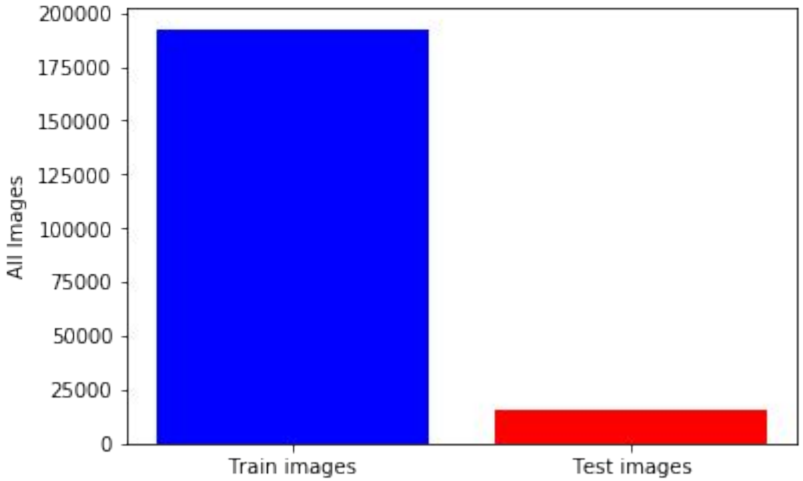
DATA:

TRAIN IMAGES:

192,556 x (758px x 758px)

TEST IMAGES:

15,606 x (758px x 758px)



TRAIN SHIP SEGMENTATION (CSV):

IMAGE - NO SHIPS

IMAGE - ONE SHIP

IMAGE - SEVERAL SHIPS

ImageId	EncodedPixels
00003e153.jpg	
0001124c7.jpg	
000155de5.jpg	264661 17 265429 33 266197 33 266965 33 267733 33 268501 33 269269 33 270037 33 270805 33 271573 33 272341 33 273109 33 273877 33 274645 33 275413 33 276181 33 276949 33 2777
000194a2d.jpg	360486 1 361252 4 362019 5 362785 8 363552 10 364321 10 365090 9 365858 10 366627 10 367396 9 368165 9 368933 10 369702 10 370471 9 371240 9 372009 9 372777 10 373546 9 374315
000194a2d.jpg	51834 9 52602 9 53370 9 54138 9 54906 9 55674 7 56442 7 57210 7 57978 7 58746 7 59514 7 60282 7 61050 9 61818 9 62586 9 63354 9 64122 9 64890 9
000194a2d.jpg	198320 10 199088 10 199856 10 200624 10 201392 10 202160 10 202928 10 203696 10 204464 10 205232 10 206000 10 206768 10 207536 10 208304 10 209072 10 209840 10 210608 10 2113
000194a2d.jpg	55683 1 56451 1 57219 1 57987 1 58755 1 59523 1 60291 1
000194a2d.jpg	254389 9 255157 17 255925 17 256693 17 257461 17 258229 17 258997 17 259765 17 260533 17 261301 17 262068 18 262836 17 263604 17 264372 17 265140 17 265908 17 266676 17 26744
0001b1832.jpg	
00021ddc3.jpg	108287 1 109054 3 109821 4 110588 5 111356 5 112123 6 112890 7 113657 8 114424 9 115191 10 115958 11 116725 12 117493 12 118260 13 119027 14 119794 14 120561 14 121328 15 1220
00021ddc3.jpg	101361 1 102128 3 102896 4 103663 6 104430 9 105198 10 105209 1 105965 14 106732 14 107500 14 108267 14 109034 14 109802 14 110569 14 111336 14 112104 14 112871 14 113638 14 11
00021ddc3.jpg	74441 3 75207 5 75975 5 76743 5 77511 5 78280 4 79048 5 79816 5 80584 5 81352 5 82120 5 82888 6 83657 5 84425 5 85193 5 85961 2
00021ddc3.jpg	74444 4 75212 4 75980 4 76748 4 77517 3 78285 3 79053 4 79821 4 80589 4 81357 4 82125 4
00021ddc3.jpg	150423 2 151190 3 151958 3 152726 4 153495 3 154263 3
00021ddc3.jpg	139644 2 140408 6 141174 9 141942 9 142711 6 143479 2
00021ddc3.jpg	75972 3 76738 5 77506 5 78274 5 79042 6 79811 5 80579 5 81347 5 82115 5 82883 5 83651 6 84420 5 85188 5 85956 2
00021ddc3.jpg	86727 2 87493 4 88261 4 89030 3 89798 4 90566 4 91334 4 92103 3 92871 1
00021ddc3.jpg	95225 2 95992 5 96760 7 97527 9 98294 9 99062 9 99829 9 100596 9 101364 9 102132 8 102901 6 103671 3 104440 2
0002756f7.jpg	255784 2 256552 4 257319 7 258087 9 258854 12 259622 14 260389 16 261159 16 261929 16 262699 16 263469 16 264239 16 265009 14 265779 11 266549 9 267319 6 268089 4 268859 1
0002756f7.jpg	248878 1 249645 4 250413 6 251180 9 251948 10 252715 13 253482 16 254250 18 255019 18 255789 18 256559 18 257329 18 258099 17 258869 15 259639 12 260409 10 261179 7 261949 5 26
0002d0f32.jpg	
000303d4d.jpg	

RLE = RUN-LENGTH ENCODING:

0002756f7.jpg 248878 1 249645 4 250413 6 251180 9 251948 10 252715 13

MASK: PIXEL POSITION NUMBER OF 'SHIP' PIXELS

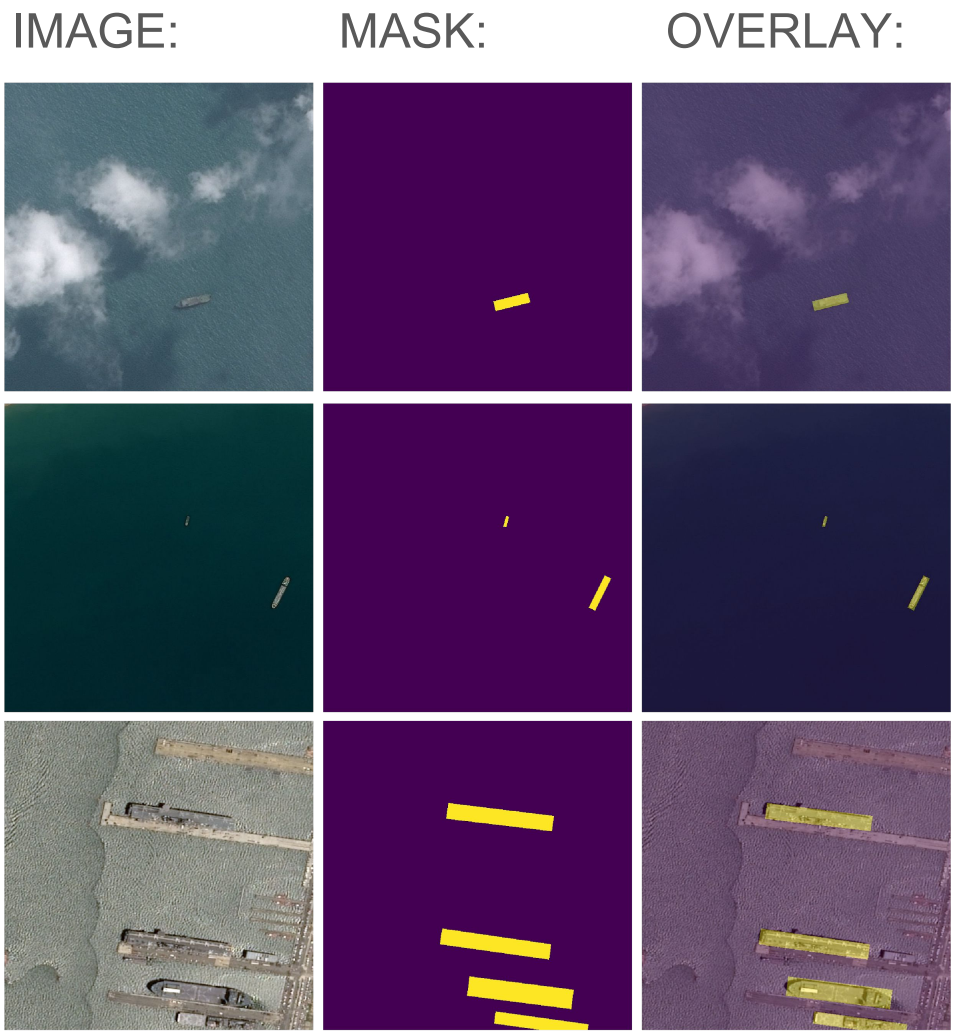


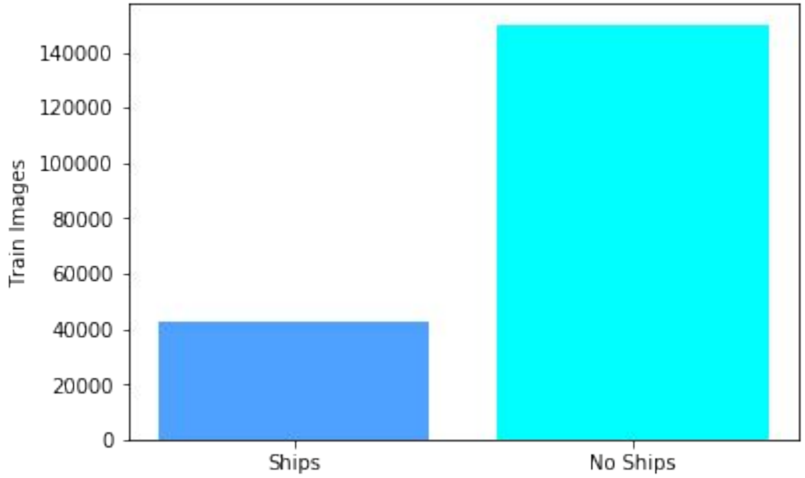
IMAGE:
MASK:
OVERLAY:

IMAGES WITH SHIPS:

42,556 x (758px x 758px)

IMAGES - NO SHIPS:

150,000 x (758px x 758px)



Evaluation Metric

- For each image we calculate the **average F_2 score** over different ***IoU*** (intersection over union) thresholds *t*.

- *t* takes values in {*0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95*}

- IoU for a prediction:

$$IoU(A, B) = \frac{A \cap B}{A \cup B}$$

where

A : a proposed set of object pixels

B : the set of the true object pixels

F_2 score for each image and each IoU threshold *t*:

$$F_2(t) = \frac{5\ TP(t)}{5\ TP(t) + 4\ FN(t) + FP(t)}$$

where

$TP(t)$: # of true positives, i.e. $IoU > t$
$FN(t)$: # of false negatives, i.e. ground truth objects without prediction
$FP(t)$: # of false positives, i.e. $IoU < t$

For images without ships the score is 0 if we make a prediction for an object and 1 if we predict that there is no object.

Average F_2 for each image:

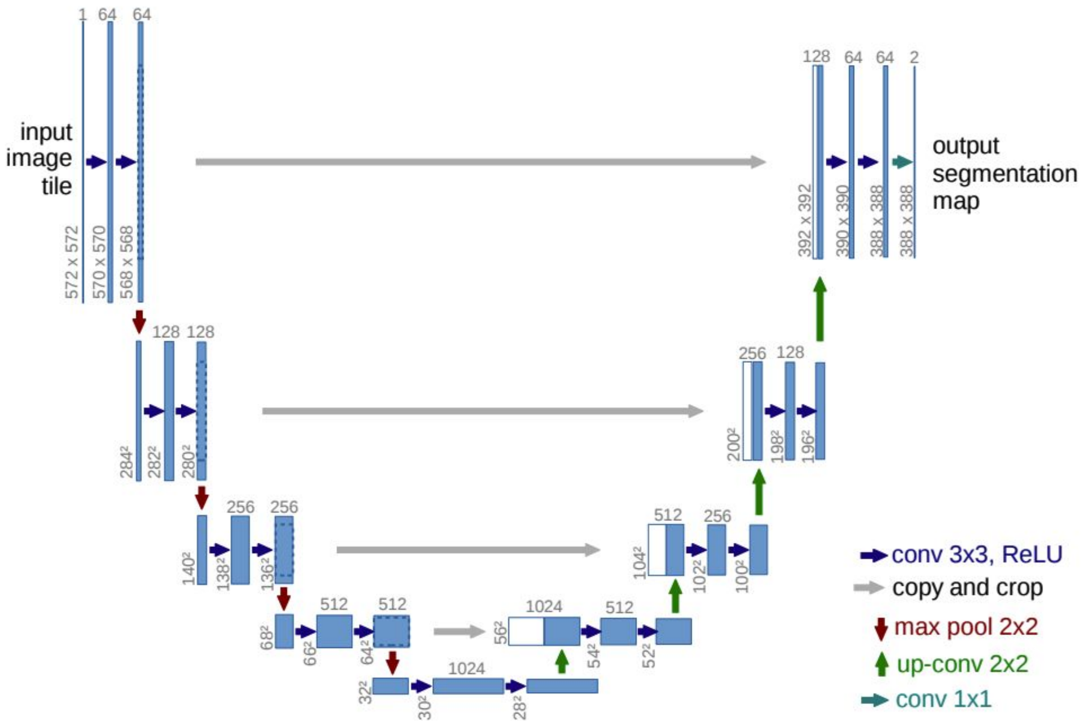
$$\frac{1}{\#\ of\ thresholds} \sum_t F_2(t)$$

Final score is the average of all the averages F_2

How to learn the mask?

- **U-NET**
- **MASK R-CNN**

Unet



1
64
64
input
image
tile
572 x 572
570 x 570
568 x 568
128
128
284²
282²
280²
256
256
140²
138²
136²
512
512
68²
66²
64²
1024
32²
30²
28²
1024
56²
54²
512
52²
512
104²
102²
256
100²
256
200²
128
198²
196²
128
64
64
2
392 x 392
390 x 390
388 x 388
388 x 388
output
segmentation
map
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1

Key ideas

- Design network as a bunch of convolutional layers, with downsampling and upsampling inside the network!
  - High res output
- Conv + pooling
- Learnable Up-convolution
- Crop

Convolution + pooling

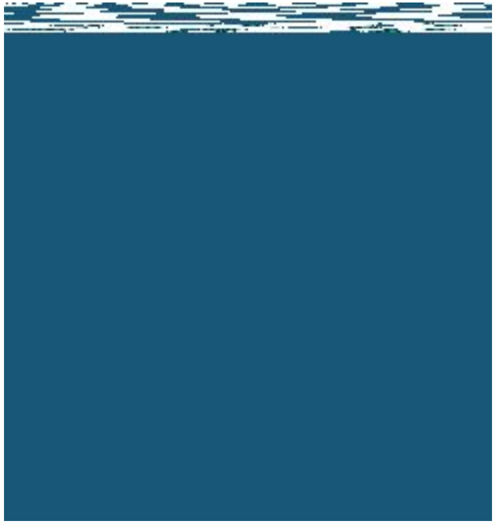
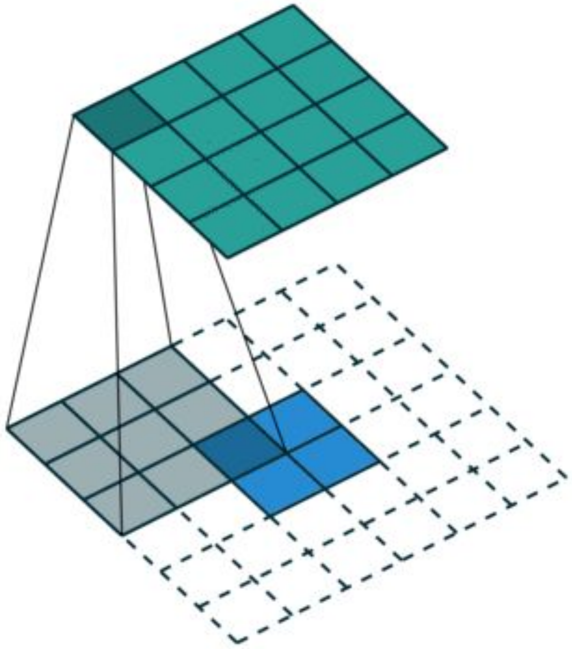
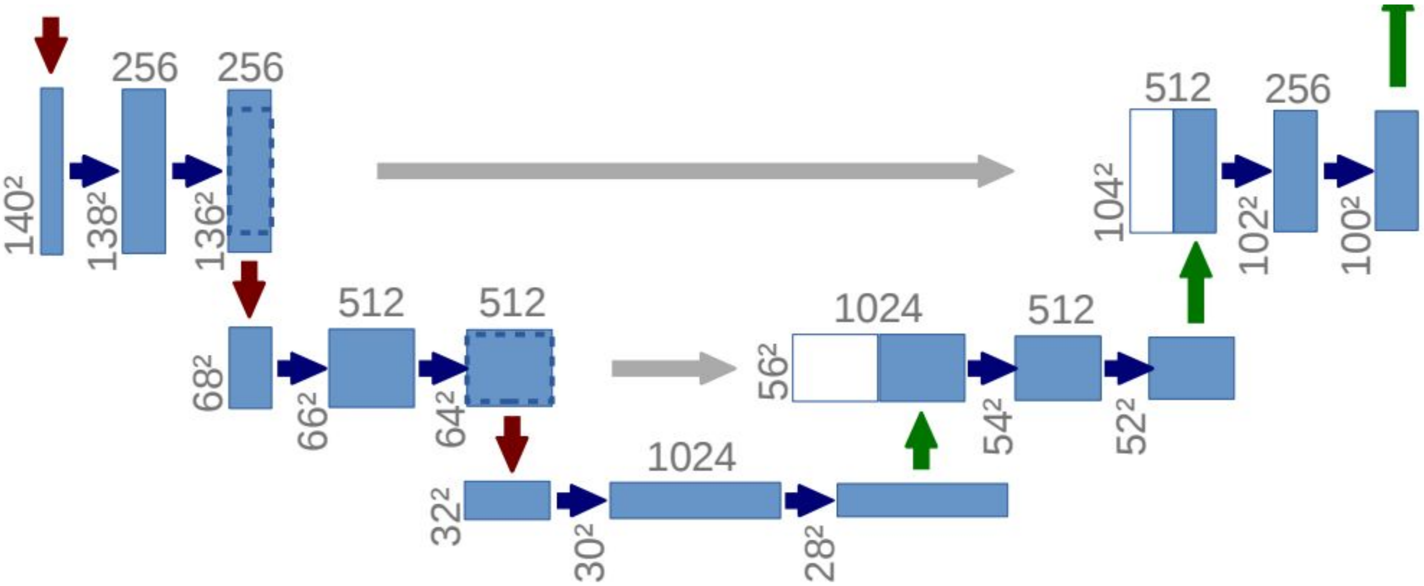


Image from this blog post

Learnable Up-convolution



Crop: stack features in upsampling into upsampling



But why?

“The cropping is necessary due to the loss of border pixels in every convolution.”

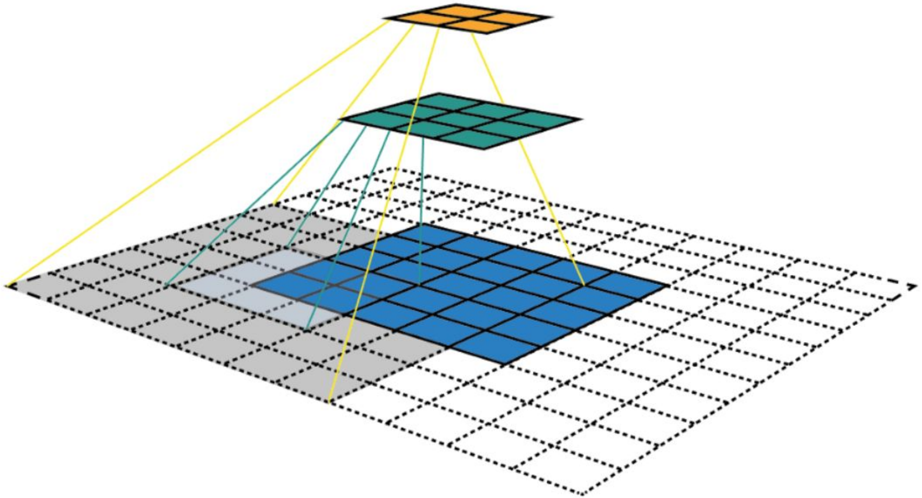


Image from this [blog post]

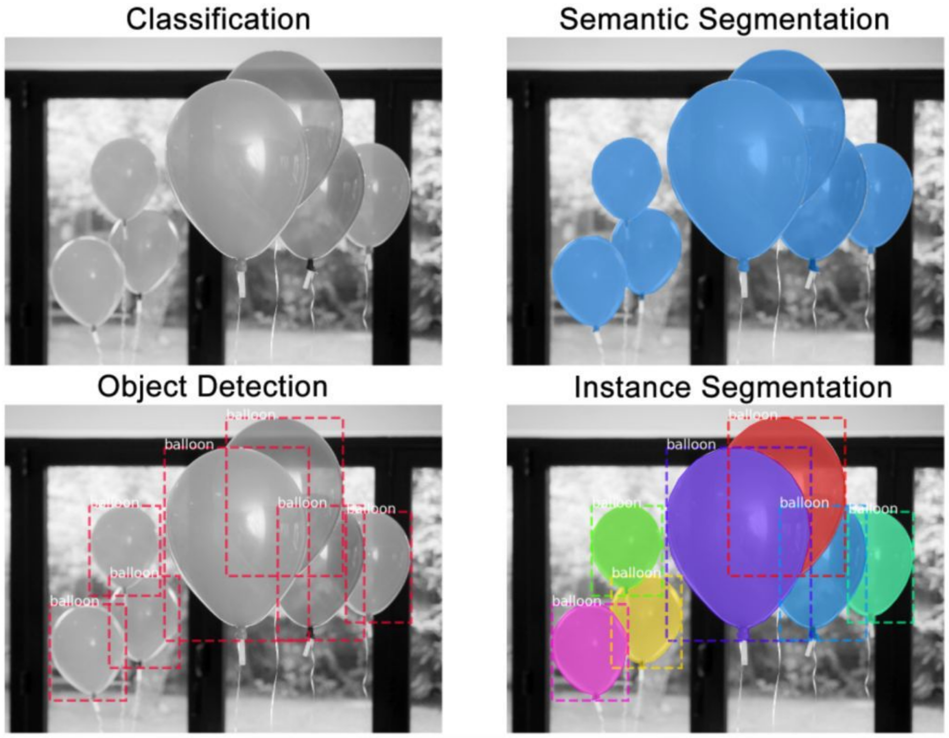
Mask R-CNN

[R-CNN]

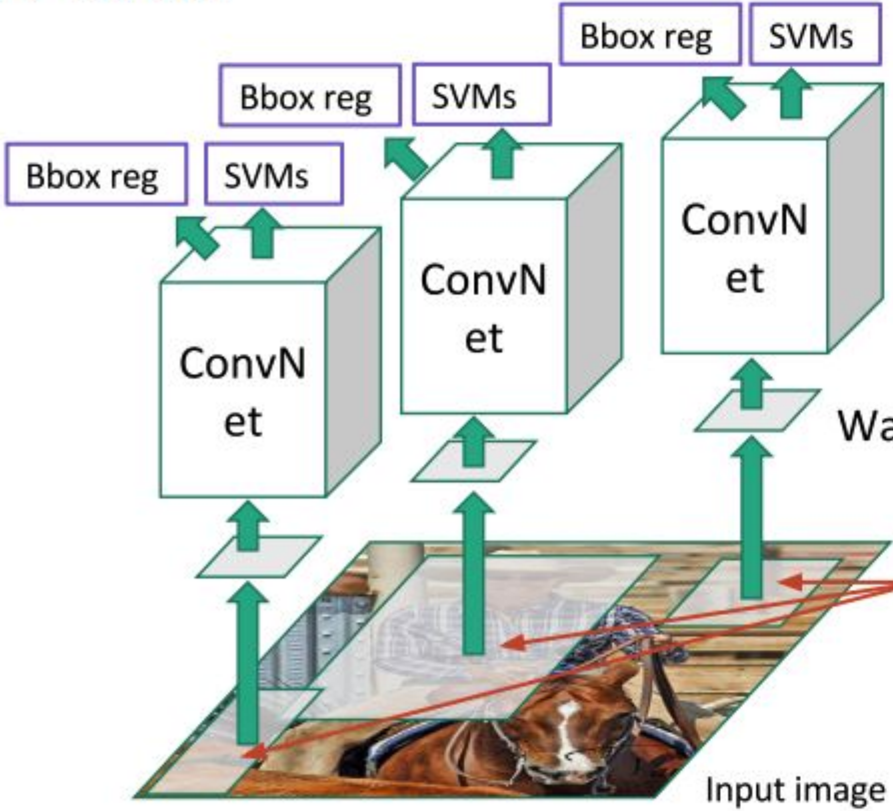
[Fast R-CNN]

[Faster R-CNN]

[Mask R-CNN]



R-CNN



Linear Regression for bounding box offsets

Fix bounding box

Classify regions with SVMs

Predict categories

Forward each region through ConvNet

Warped image regions

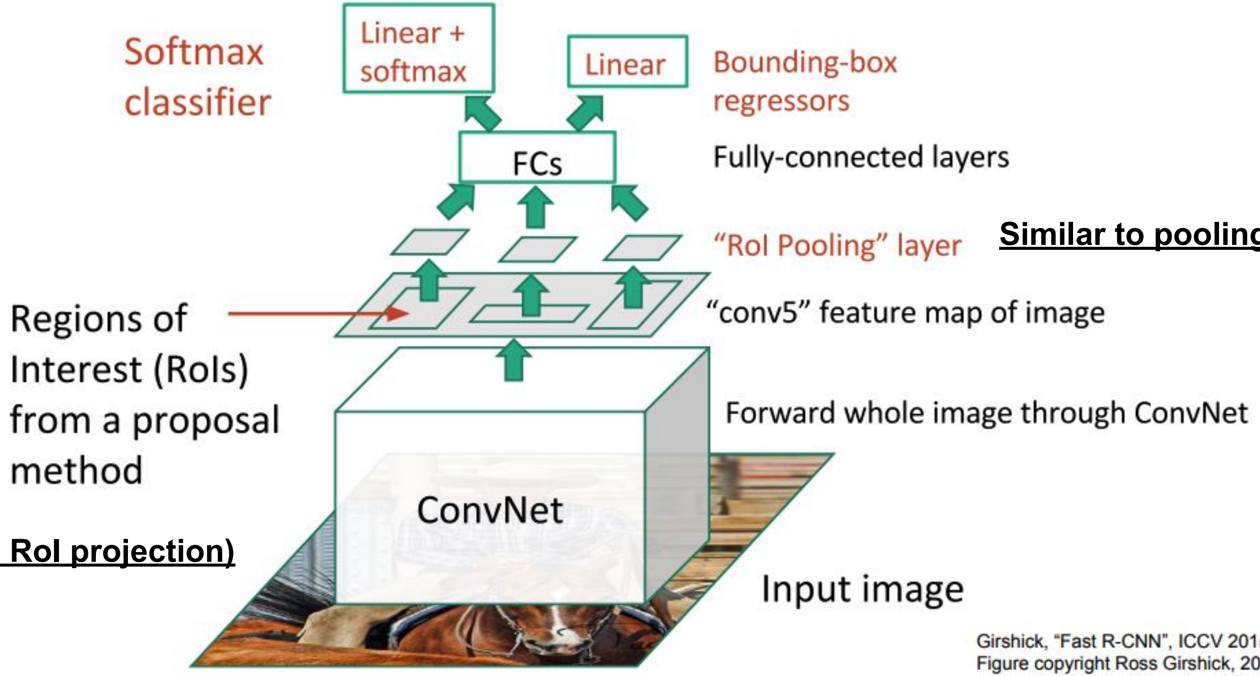
NN needs a fixed size input

Regions of Interest (RoI) from a proposal method (~2k)

Traditional vision algorithm

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

Fast R-CNN

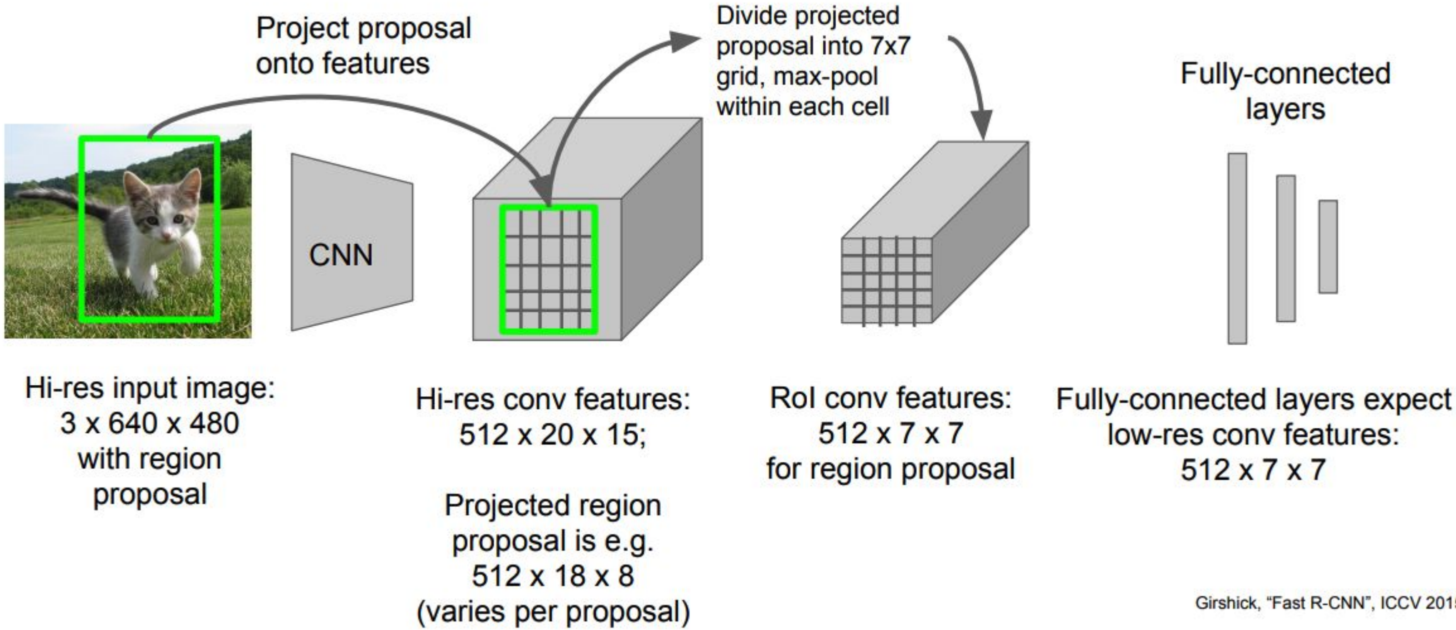


<u>Similar to pooling in CNN</u>

<u>(By RoI projection)</u>

Girshick, "Fast R-CNN", ICCV 2015.

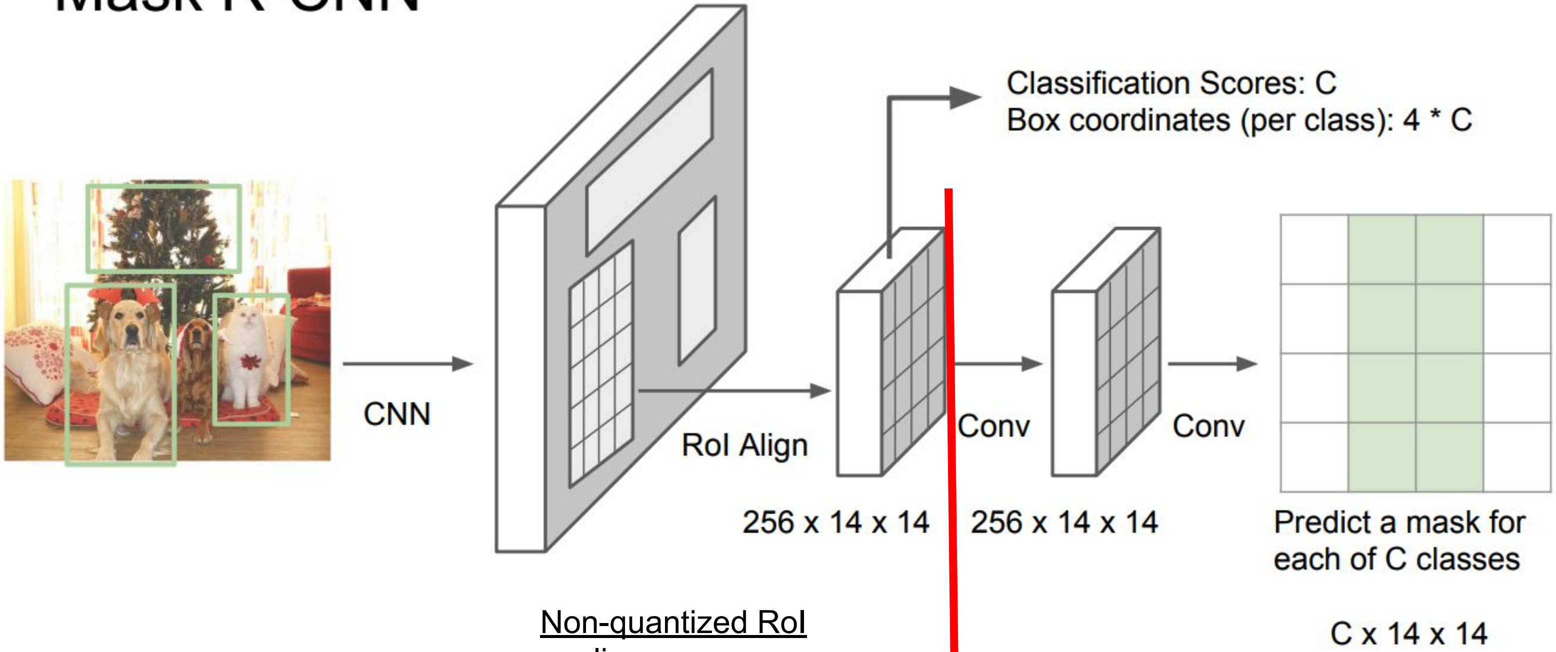
Faster R-CNN: RoI Pooling



Girshick, "Fast R-CNN", ICCV 2015.

Region Proposal Networks(RPN): Use NN to propose region

Mask R-CNN

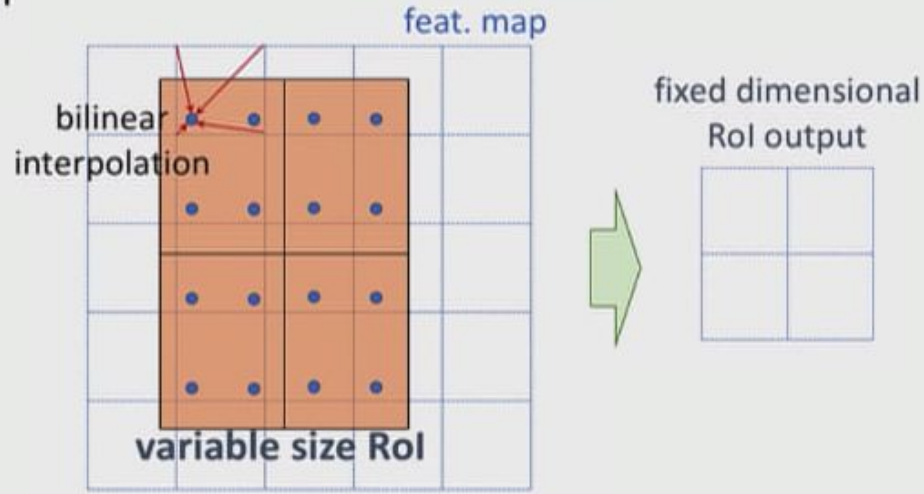


Multi-class loss

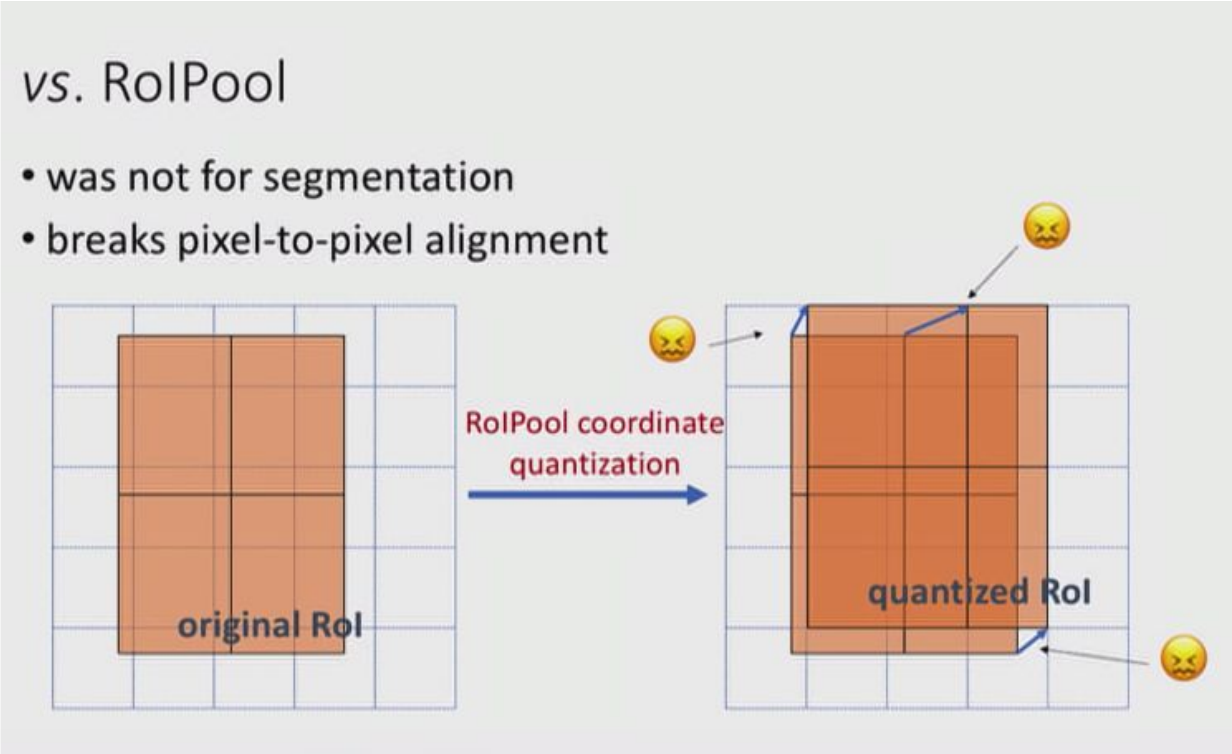
Non-quantized RoI pooling

RoIAlign

- No quantization

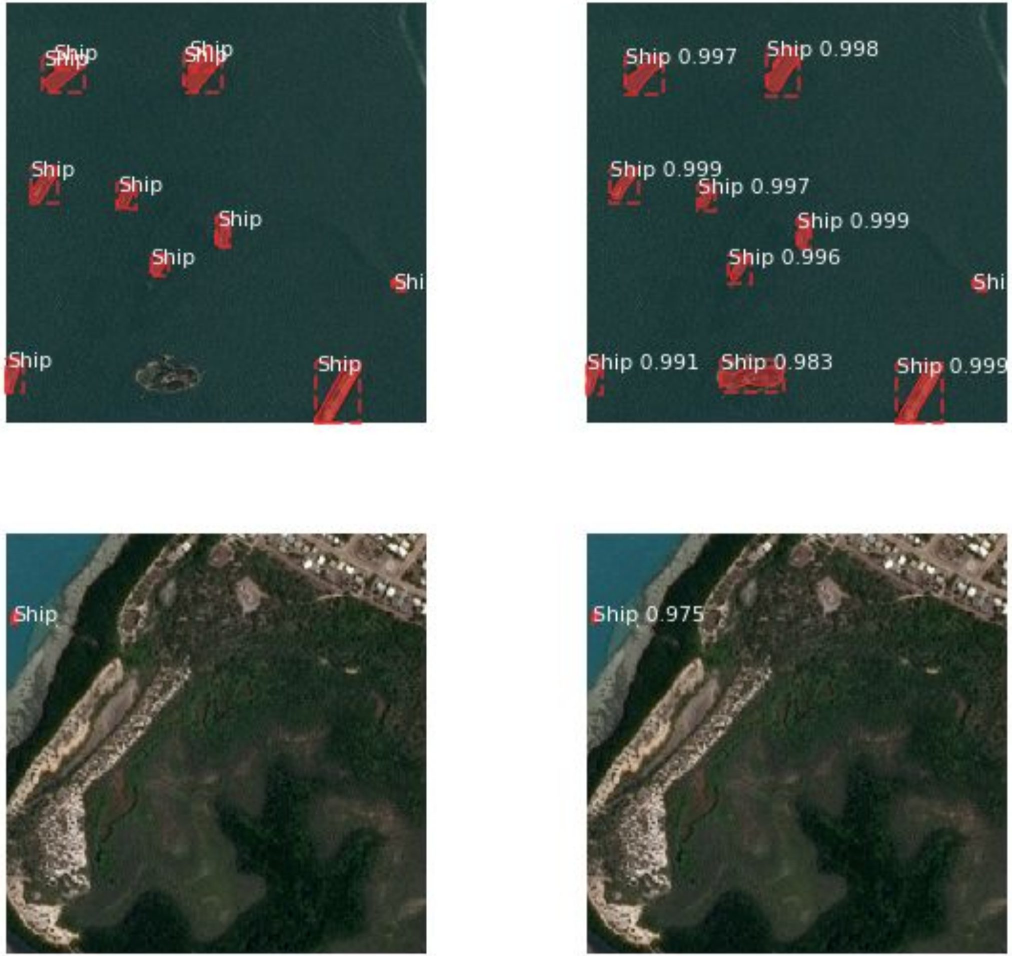


RoIAlign vs RoI Pooling



Our solution

- Mask R-CNN with coco pretrained weight
    - Based on an open kernel
        - Code to produce submission file
    - Based on this Really good Mask R-CNN implementation
        - Handle run-length encoding
        - Easy to define parameters like number of classes(2 in this competition) and image dimension
    - BACKBONE: RESNET50
    - Drop learning rate after some epochs


Ship
Ship
Ship
Ship
Ship
Ship
Ship
Ship
Ship
Shi
Ship
Ship
Ship 0.997
Ship 0.998
Ship 0.999
Ship 0.997
Ship 0.999
Ship 0.996
Shi
Ship 0.991
Ship 0.983
Ship 0.999
Ship
Ship 0.975

Loss Functions used by the teams

Together with some optimization function can give smaller errors.

(Binary) Cross entropy loss

It increases as the predicted probability diverges from the actual label.

$$-(y\log(p) + (1-y)log(1-p))$$

where p is the prediction probability and

y a binary indicator on whether the classification is correct or not.

It doesn't work very well in a strongly unbalanced data set.

Dice loss function
(similar to the intersection over union)

$$\text{Dice Similarity Coefficient (DSC)} = \frac{2\, S \cap R}{S \cup R}$$

We take a differentiable version of the above:

$$\mathcal{L}_{DSC} = \frac{2 \sum s_i r_i}{\sum s_i + \sum r_i}$$

Often leads to unstable training.

Combinations of BCE and Dice loss

For example

$$BCE - \log(\mathcal{L}_{DSC})$$

(it pushes predictions to the ends of the [0, 1] interval)

Or

$$1 + BCE - \mathcal{L}_{DSC}$$

Focal loss function

Modifies BCE by adding a factor:

$$-(y(1-p)^{\gamma}\log(p) + (1-y)p^{\gamma}\,log(1-p))$$

where p is the prediction probability,

y a binary indicator on whether the classification is correct or not, and

γ a focusing parameter.

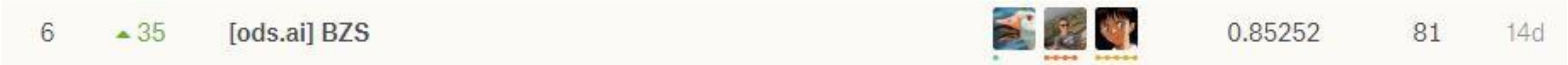
10th place solution

- Mask-RCNN
- No ensemble

Ours VS 10th solution

- RESNET 101
- Multi-Scale Training: Small object are hard for detection.
- Online Hard Example Mining selecting top 128 roi's for training
- Soft-NMS : Algorithm to remove repeated detection box
  - Sort all detection boxes based on detection scores
  - Penalize on overlap between masks

6th place solution



https://www.kaggle.com/c/airbus-ship-detection/discussion/71782

Models they used:

- Ship/no-ship classifier: InceptionResNetV2
  trained on 299x299

- Semantic segmentation: ResNet34 + U-Net
  trained on 256x256 random crops, prediction on a full-size
  50/50 ship/no-ship images in the batch
  Performance: 0.846 → 0.848 (with Classification)

- Semantic segmentation: ResNet152 + U-Net
  trained on 224x224 random crops
  90/10 ship/no-ship images in the batch
  Performance: 0.775 → 0.848 (with Classification)

- Instance segmentation: ResNet18/ResNet50 + Mask R-CNN
  trained on 1000x1000
  Performance: 0.843 → 0.850 with Classification

Final solution

Geometric mean of 7 U-Net models and an ensemble of 3 Mask R-CNN models

4th place solution

4	▲ 22	[attention heads]	0.85411	55	14d

https://www.kaggle.com/c/airbus-ship-detection/discussion/71667

- They focused on having a better ship/no-ship classifier rather than an excellent segmentation model.
- Validation: 9000 images, mostly images with ships - no leaks
- Used U-Net
- Encoder: ResNet34, trained 300+ epochs on cropped images (256x256)
- With binary cross entropy loss they had higher chance of splitting ships, but worse mask shapes

Comments from bestfitting (won 3rd place):

- “To improve efficiency, learning from others and learn from previous competitions are important.”
- “I read papers everyday…>100 per competition often”
- “I do physical exercises every day at least for an hour”

Lesson Learnt as a first-time kaggler

- Learning curve is steep…
- Check out public kernels such as baseline models. They were great starting point to the competition + evaluation metric + basic on generating a submission file.
- Pre and post-processing the data + correct train/validation split as important as training the model.
- Bruce was a great mentor and really helped guide us through the process + a walking FAQ.
- In this dataset, there was a heavy bias to images with no ships. The best solutions seem to have a ship/no-ship identifier first, then added some no-ship images to balance out the training/validation set before re-training the model and post-processing.

Lesson Learnt as a first-time kaggler

- Having a pre-existing infrastructure to run and train models is important. Luckily Bruce provided us with access to his server. Learning to access the server and working together on one server was also challenging.
- Learnt far more than I would have expected and I feel comfortable with doing other Kaggle competitions now.

Thank you!