# Dstl Safe Passage: Detecting and Classifying Vehicles in Aerial Imagery


## Vladimir Iglovikov

Physics, PhD

Kaggle Master

# Historical overview

December 2016 - March 2017

***Kaggle: Dstl Satellite Imagery Feature Detection***

Roman Solovyov, Artur Kuzin 2nd place ($30,000)

Vladimir Iglovikov, Sergey Mushinskiy 3rd place ($20,000)

- blog posts (rus, eng)
- meetup talks (rus, eng)
- paper (next week)

Organizers spent $465,000 and got state of the art solutions that they can not use.

# Historical overview

March 2017

- Press release: Dstl's Kaggle competition has been a great success
- DSTL pays BAE Systems to create their own Kaggle: https://www.datasciencechallenge.org and start two competitions (Computer Vision and Natural Language Processing)
- Problems are pretty good, but rules of the competitions are discriminatory (Everyone can participate, but only limited set of people can claim prize money)
- We got verbal and written promise from organizers that rules will be changed.

# Problem Statement

- RGB satelite images
- 2000x2000
- 5cm / pixel
- 600 train
- 600 test
- 9 classes

# Problem Statement: class distribution

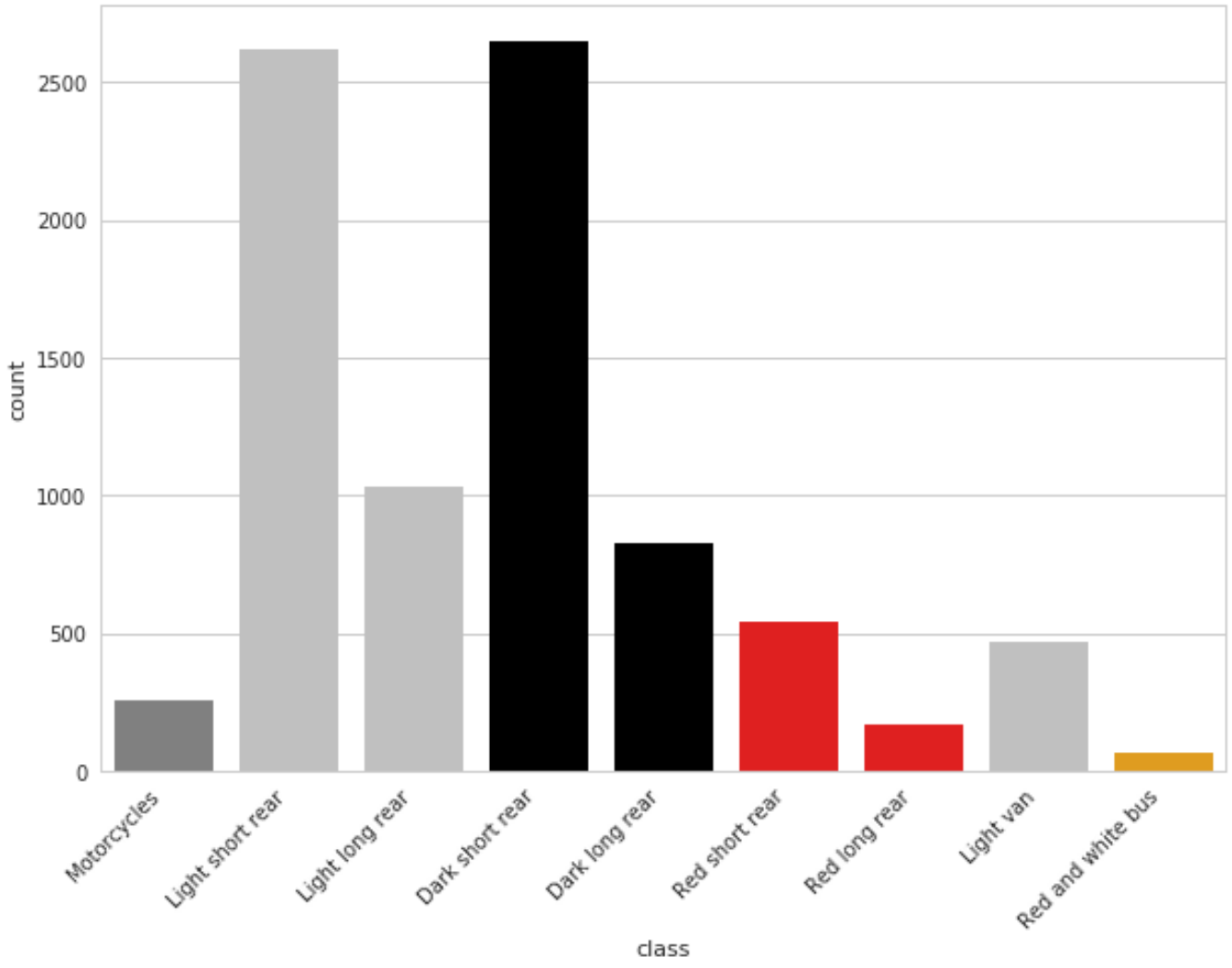



Figure by Vladislav Kassym

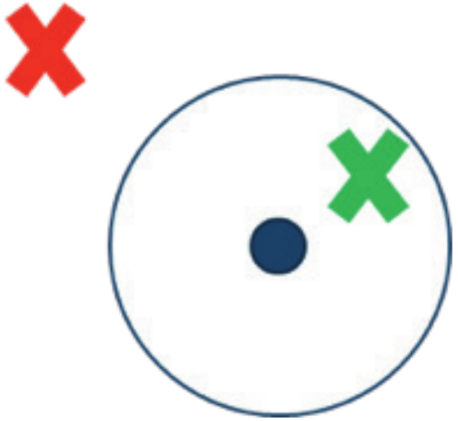
# Problem Statement

One quarter of one image

- train: 600 images
- test: 600 images
- 2000x2000
- 5 cm / pixel

# Evaluation Metric

$$Jaccard = \frac{TP}{TP+FN+FP}$$



| **Class** | **Radius** |
|---|---|
| motorcycle | 12 pixels (60 cm) |
| cars | 30 pixels (150 cm) |
| van | 40 pixels (200 cm) |
| bus | 45 pixels (225 cm) |

# Motivation

## Why participate?

- Very clean balanced dataset.
- Knowledge in Image Detection.
- Good amount of data. (Not too much, not too little.)
- No data leaks.
- Codebase will be reused in:
  - Kaggle: Cervix
  - Kaggle: Seals
  - ImageNet 2017

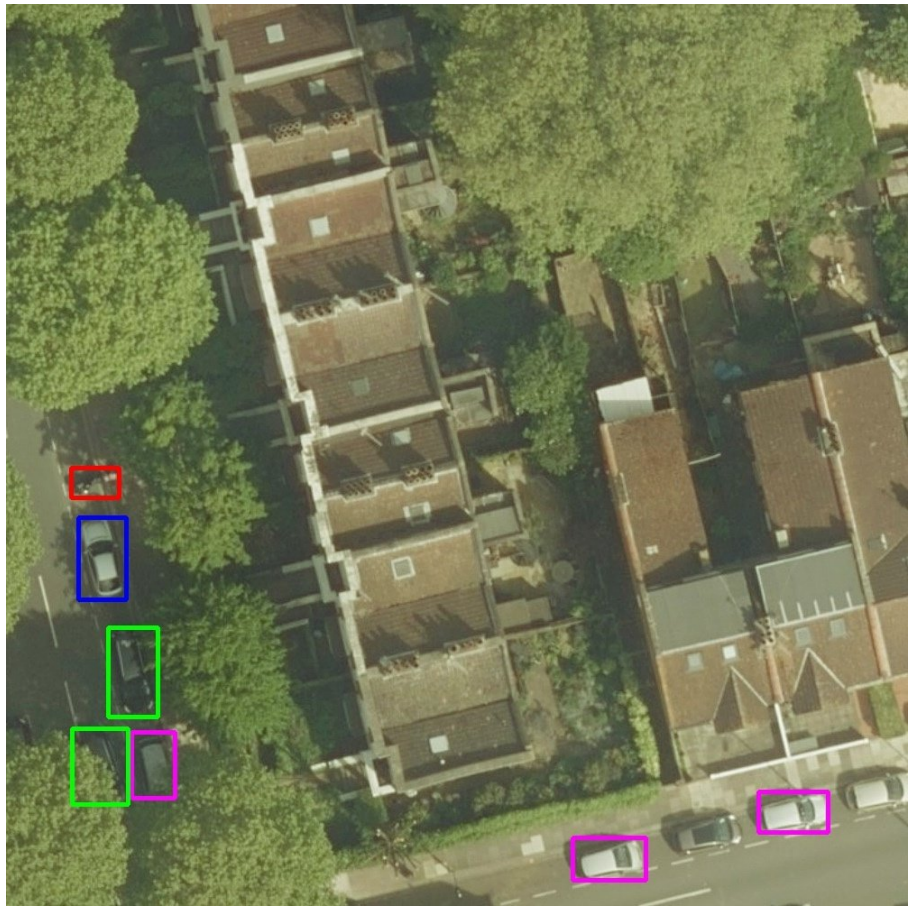
## Why not participate?

- No way to claim prize money.
- No community.
- Unknown platform. (Hard to sell results.)

# Step 1: bounding boxes

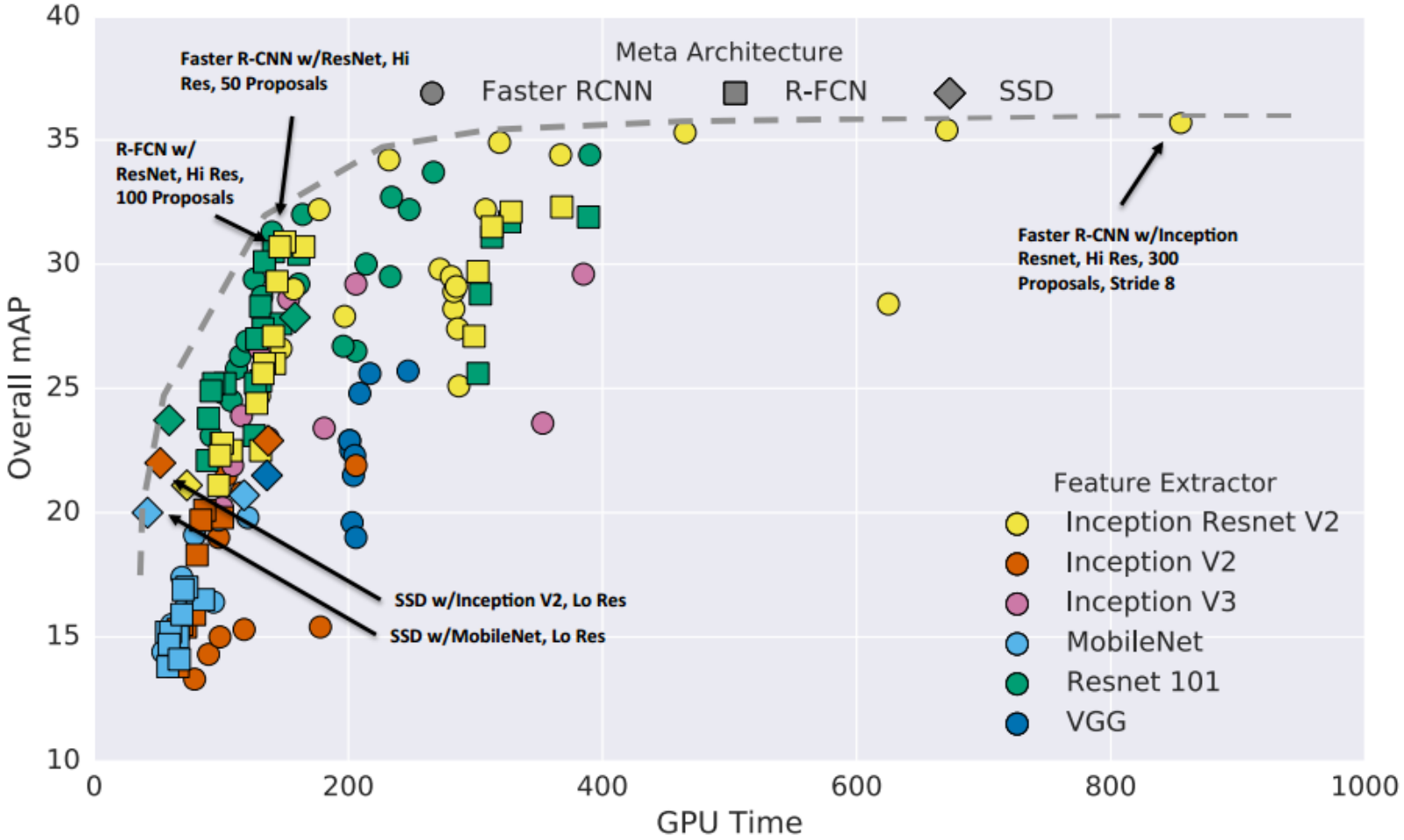
Before

After



~ 10 hours

# What network architecture to use?




Speed/accuracy trade-offs for modern convolutional object detectors

arXiv:1611.10012

# What network architecture to use?

## Faster RCNN

- Slow to train
- Slow to predict
- Accurate in general
- Accurate on small objects

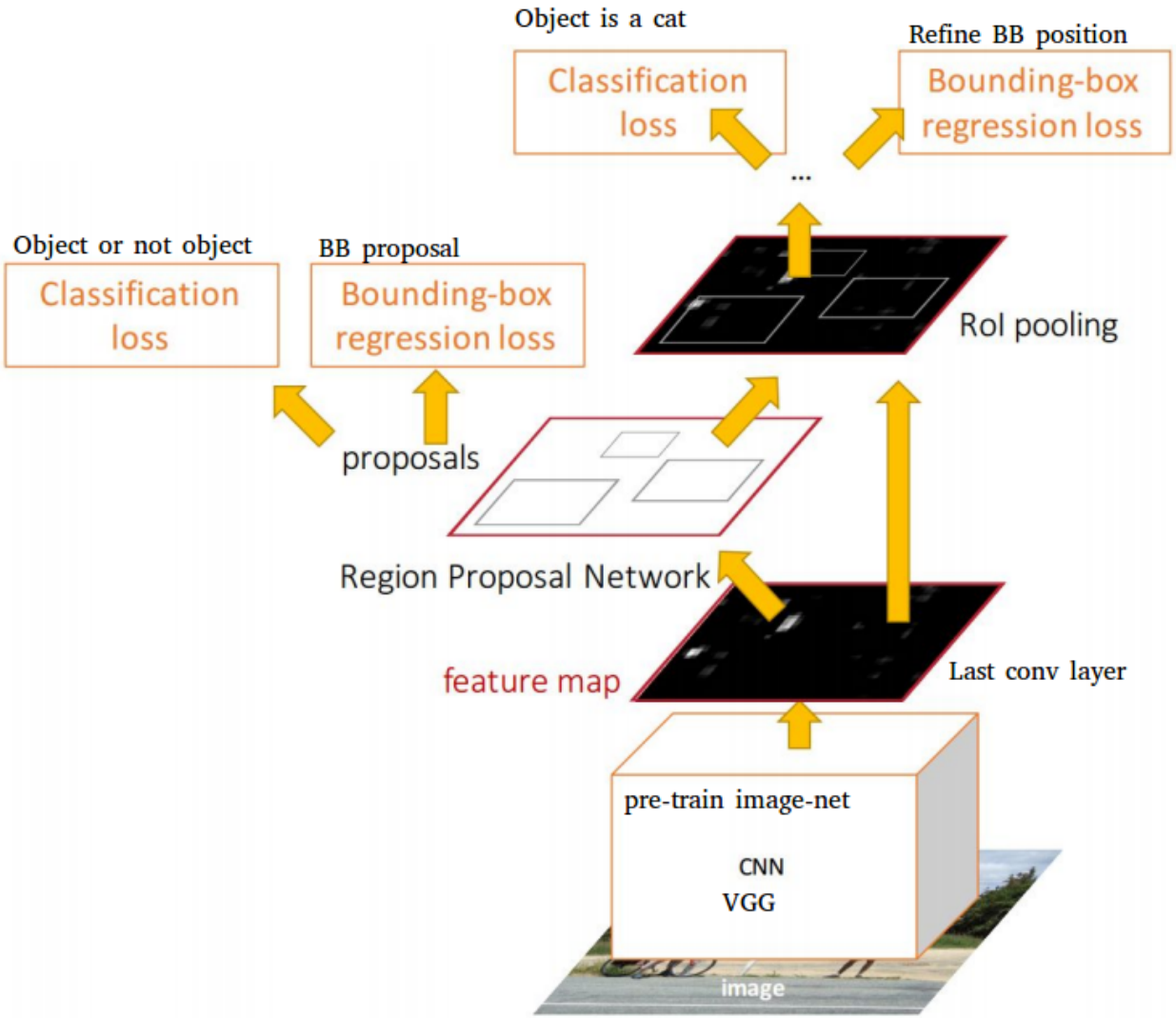
## SSD

- Fast to train
- Fast to predict
- Less accurate in general
- Pretty bad with small objects

=>

For this task winner: Faster RCNN

# Faster RCNN


Object is a cat
Classification loss
Refine BB position
Bounding-box regression loss
...
Object or not object
Classification loss
BB proposal
Bounding-box regression loss
RoI pooling
proposals
Region Proposal Network
feature map
Last conv layer
pre-train image-net
CNN
VGG
image

# What framework to use?

## Keras + TensorFlow

- Existing Faster RCNN implementation
- Familiar code base
- Good documentation
- Slow
- Pain to parallelize

## MXNet

- Existing Faster RCNN implementation
- Unfamiliar code base
- OK documentation
- Fast
- Zero pain with  parallelization

=>

For this task winner: MXNET

# Solution

Code - example from MXNet repository

## Train

- Faster RCNN + VGG16 base
- random crops 1000x1000
- D4 group augmentation

8 samples/sec

## Test

- overlapping tiles
- D4 group augmentation
- Non-Maximum Suppression

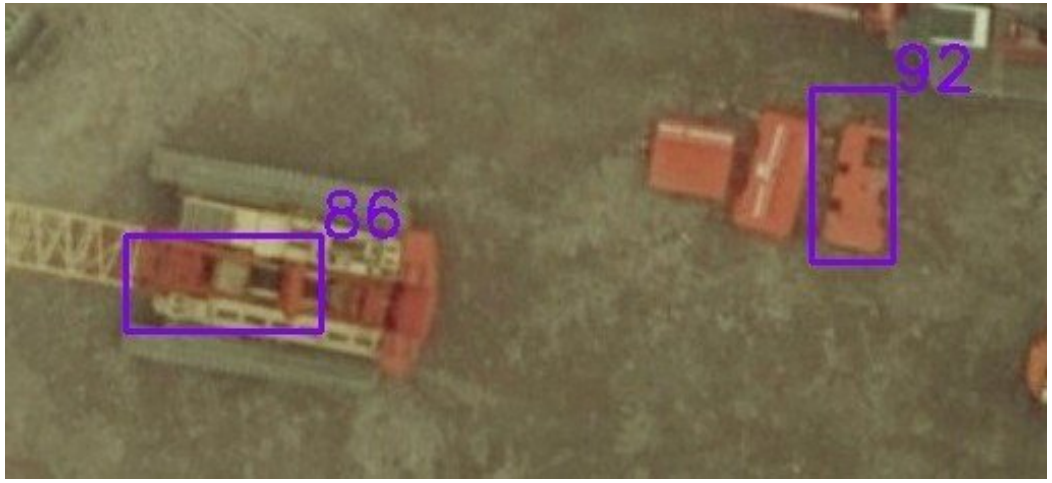
20 samples/sec

# Sources of mistakes: close- packed objects

# Sources of mistakes: trains like buses


# Sources of mistakes: debris as cars



# Main source of mistakes: misclassification

gray car in the shade <=> black car

gray car in the sun <=> white car

blue car in the shade <=> black car

white hatchback <=> white van

hatchback <=> sedan

=>

inconsistent labeling

low predictive power

## Top 10 entries

| RANK | USER | PUBLIC | PRIVATE | DATE | TREND | ENTRIES |
|---|---|---|---|---|---|---|
| 1 | gbarbadillo | 0.8662 | 0.8713 | 17 May 2017, 6:55PM BST | | 84 |
| 2 | ternaus | 0.8581 | 0.8569 | 17 May 2017, 4:47PM BST | | 31 |
| 3 | kit1 | 0.8633 | 0.8527 | 16 May 2017, 8:12PM BST | | 36 |
| 4 | jane.ostin | 0.8490 | 0.8477 | 14 May 2017, 8:40PM BST | | 22 |
| 5 | Xi_Lian | 0.8493 | 0.8455 | 16 May 2017, 11:33PM BST | | 8 |
| 6 | Kyle | 0.8320 | 0.8369 | 16 May 2017, 4:05PM BST | | 48 |
| 7 | vkassym | 0.8333 | 0.8325 | 14 May 2017, 8:22PM BST | | 17 |
| 8 | dlrocks | 0.8142 | 0.8189 | 17 May 2017, 10:46PM BST | | 20 |
| 9 | cogitae | 0.7964 | 0.8054 | 17 May 2017, 9:03PM BST | | 48 |
| 10 | codewarrior | 0.8043 | 0.8000 | 04 May 2017, 11:09PM BST | | 16 |


100
100
99
100
99
99
99
100
100
74
100
99
99
100
99
99
99
99
99
99
100
99
100
100
100
99
99
99
100
100
99
70
99
100
99
99
100
100

# Summary

- Centers of cars => bounding boxes (manually)
- Faster RCNN + VGG16, MXnet
- D4 group train and test time augmentation

# Hardware

- Intel i7
- 32Gb RAM
- 2 x Titan X (Pascal)

# Many thanks to:

- Sergey Mushinskiy
- Vladislav Kassym
- Sergey Belousov