Computer Vision capstone project

Face Detection

Table of content

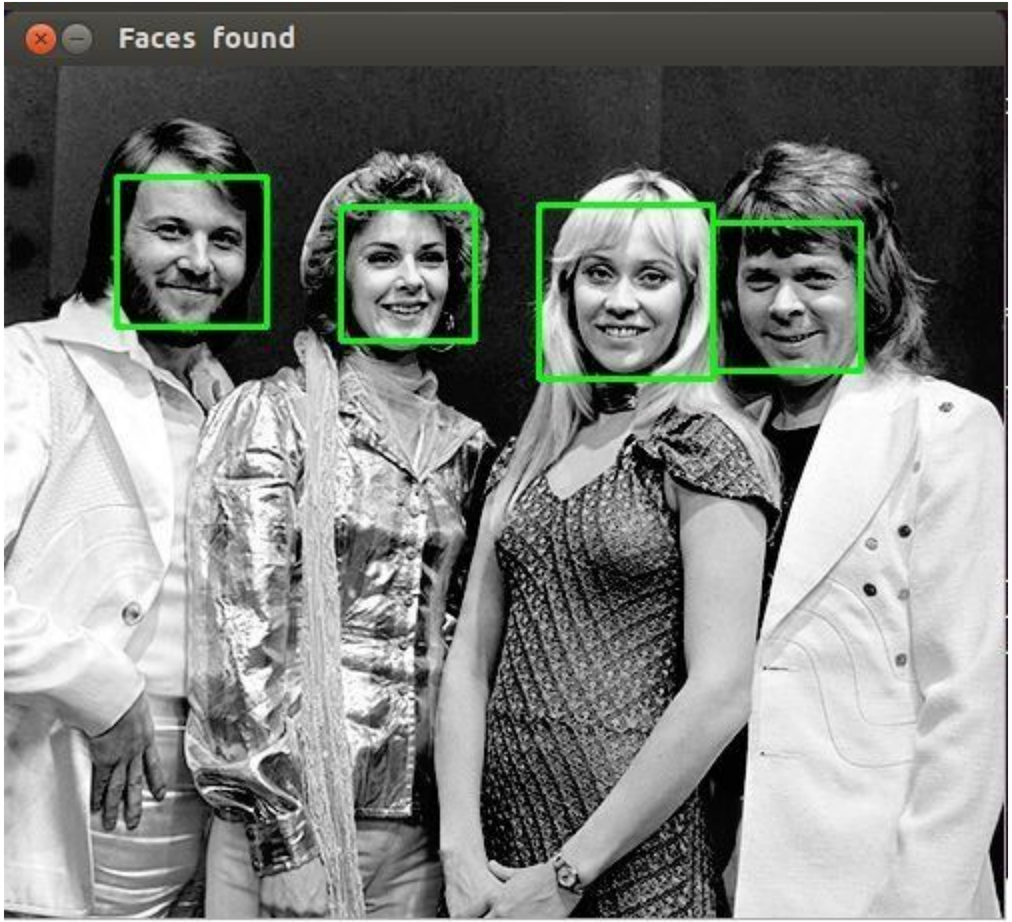
- Introduction
- Dataset
- Data augmentation
- Methods
    - HOG-SVM
    - Faster R-CNN
    - SSD
- Evaluation and Results
- Conclusion

Introduction

Introduction

Face Detection is one of the active fields of Computer Vision with many applications: security, biometrics, entertainment,...

In this project, we apply 3 face detection methods, from classical to more modern ones, and examine their performance.



Dataset

Dataset

We use WIDER FACE dataset:

- Full dataset: roughly 32000 images, more than 390000 labeled faces.
- Test and dev set categorized into 3 subsets: easy, medium, and hard.
- Official split: 40% train / 10% dev / 50% test.

Dataset (Cont.)

We use WIDER FACE dataset:

- However, the ground truth label for test set is not publicly available.

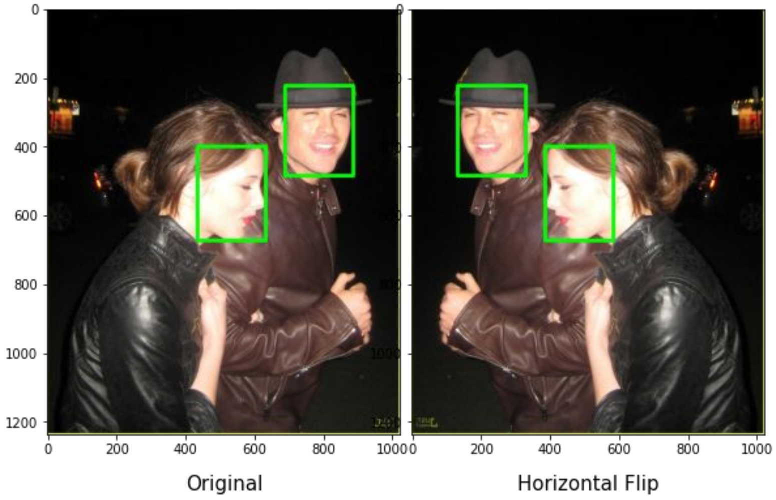
  => We use the 3000 labeled validation images as test set, and further split the original training set into a new training and validation set.

Subset	No. Images
Train	10304
Validation	2576
Test	3226

Data Augmentation

Data Augmentation

Some augmentation methods requires the adjustment of the ground truth labels.



… some other augmentation methods don’t.

Data Augmentation

Augmentations we use: horizontal flip, scaling, translation, shearing, blurring, distortion (random contrast, brightness ...).

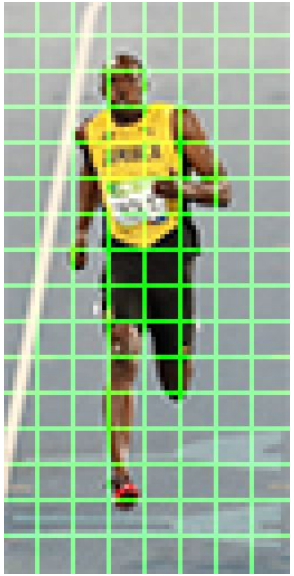
Data augmentation pipeline: images passing through this pipeline will be applied:

- Each method which affect bounding boxes at 50% chance.
- Each distortion and blurring method at 50% chance.

Methods

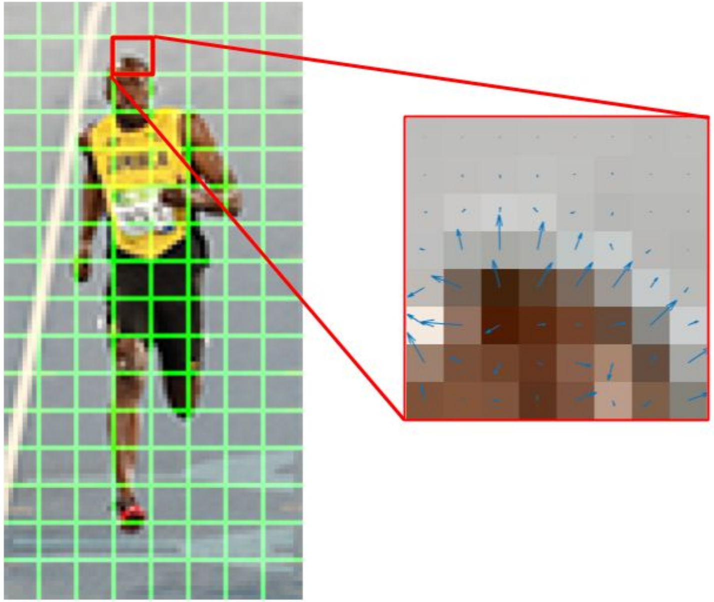
HOG-SVM

- Histogram of Oriented Gradients
  + Divide an image patch into C X C cells



HOG-SVM

- Histogram of Oriented Gradients
    + Calculate gradient for each cell



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

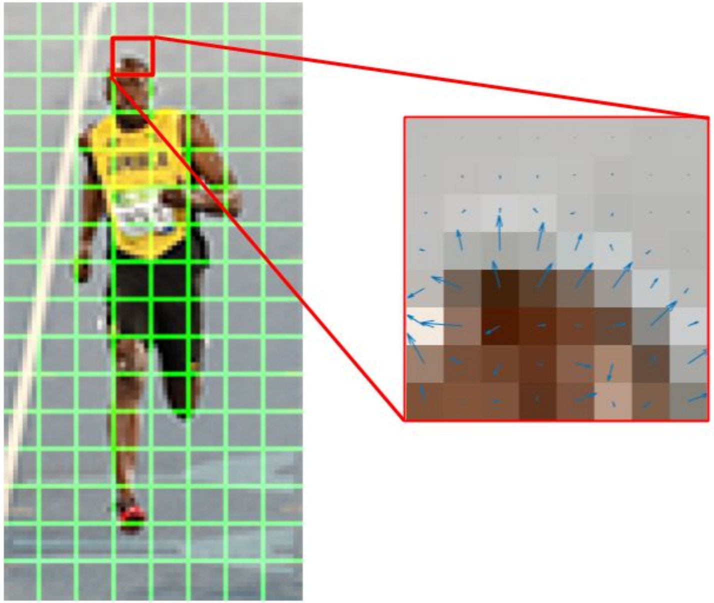
Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

HOG-SVM

- Histogram of Oriented Gradients
  - + Calculate gradient for each cell



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

Gradient Magnitude

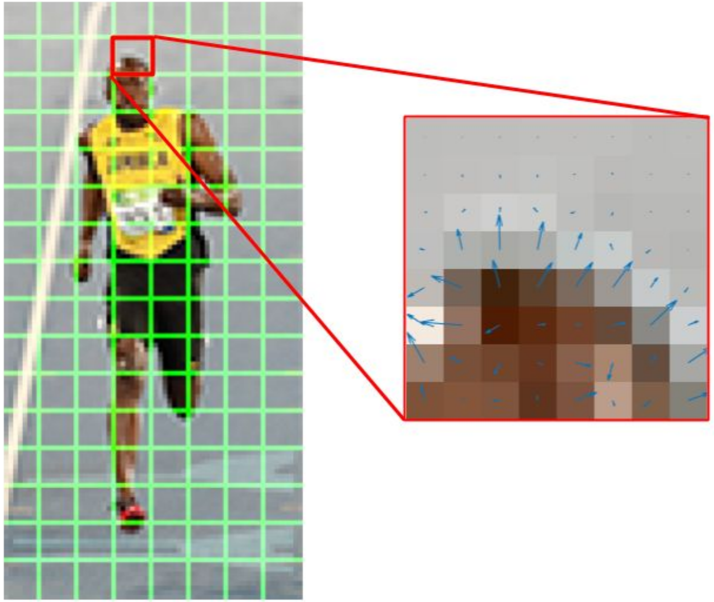
80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

Denotes the direction of change in intensity

HOG-SVM

- Histogram of Oriented Gradients
  + Calculate gradient for each cell



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

Gradient Magnitude

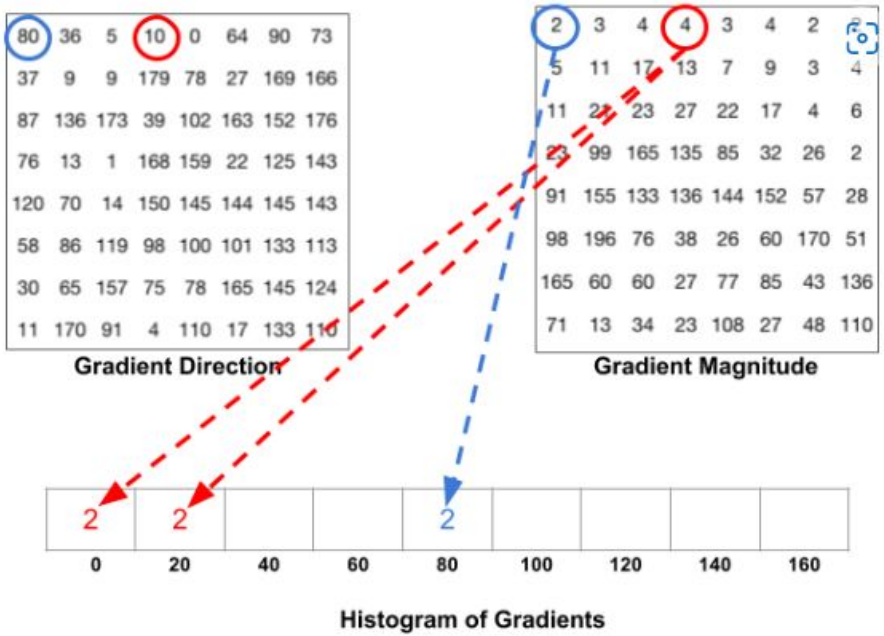
Denotes the magnitude of the change

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

HOG-SVM

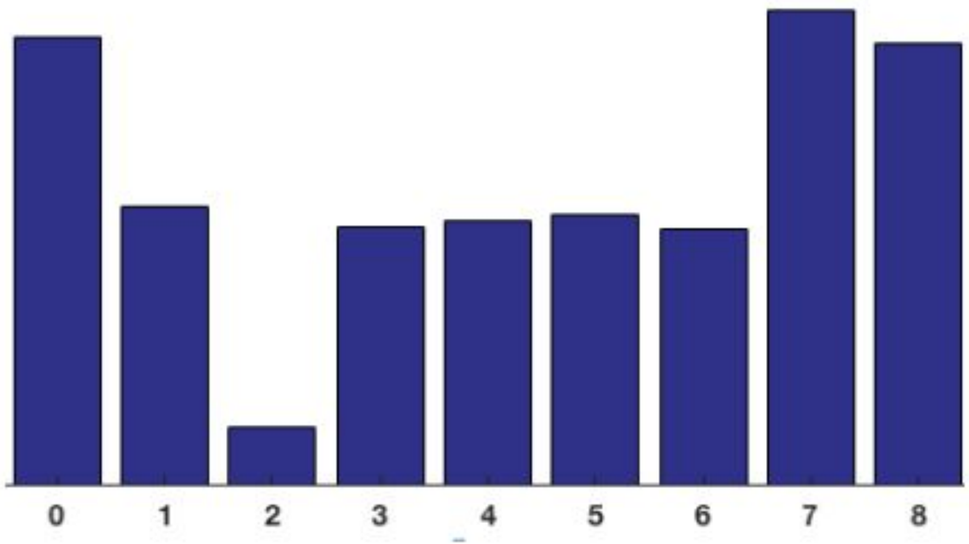
- Histogram of Oriented Gradients
  + Create a histogram of gradients with N bins



- A bin is selected based on the direction.
- The value that goes into the bin is based on the magnitude

HOG-SVM

- Histogram of Oriented Gradients
  + Create a histogram of gradients with N bins



HOG-SVM

- Train SVM classifier
    + Input: HOG descriptor
    + Trained to classify between Positive (face) and Negative (non-face) images

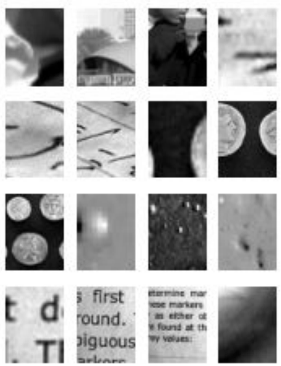
HOG-SVM

Positive Data: *LFW dataset*

- More than 13,000 images of cropped faces
- Converted to grayscale
- Resized to a same size (36 x 48)

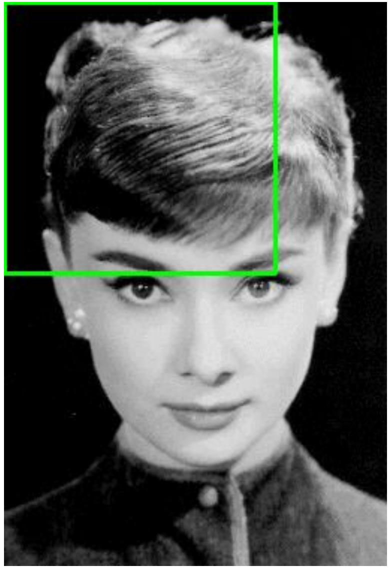
Negative Data

- 30000 image patches of random stuffs: newspaper, fields ...
- Obtained through skimage API
- Converted to grayscale
- Resized to a same size (36 x 48)



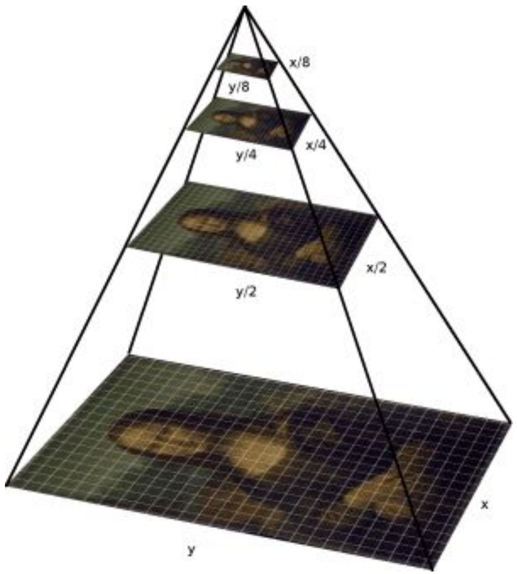
HOG-SVM

- Detection with HOG and SVM
  + Run a sliding window through an input image
  + Extract HOG feature for each windows
  + Classify window with SVM



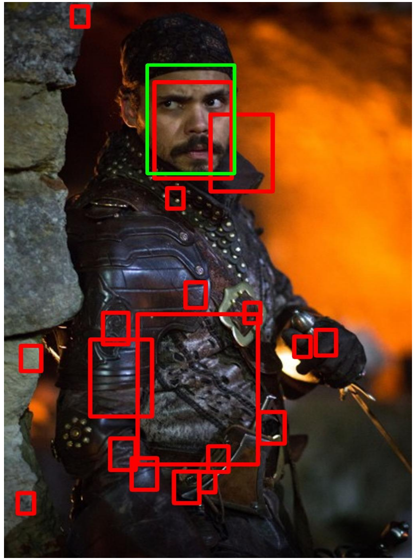
HOG-SVM

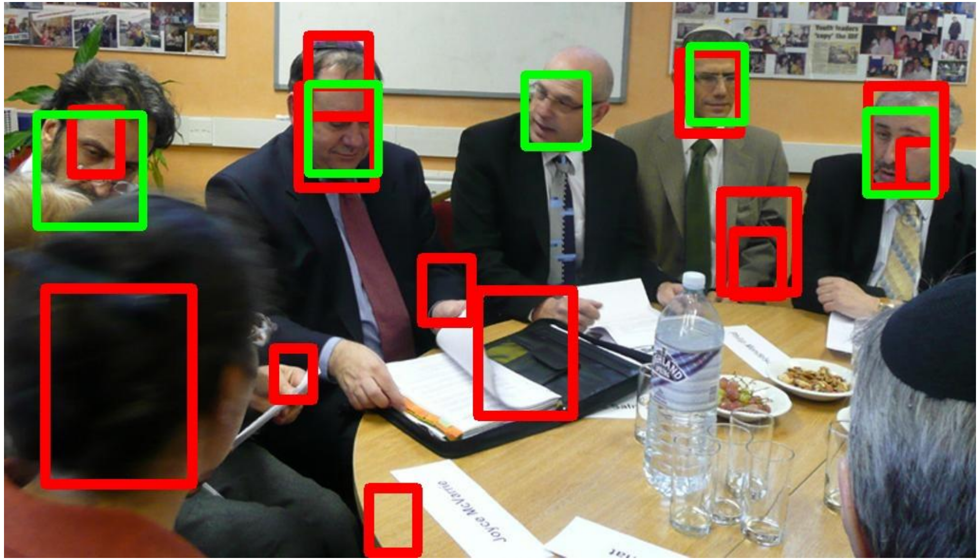
- Detection with HOG and SVM
    + Perform detection on multiple scales for faces with multiple sizes



HOG-SVM

- Hard Negative Mining
    + HOG-SVM are susceptible to False Positives





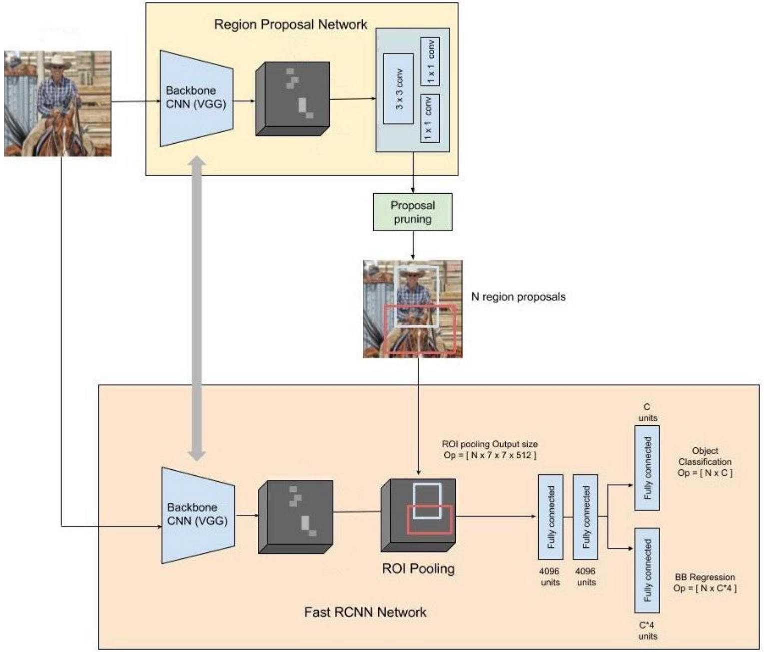
HOG-SVM

- Hard Negative Mining
    - Run SVM model on WIDER FACE training set
    - Save all False Positive image patches
    - Retrain model with these FPs as additional data

⇒ *Enhance the model with harder negative instances*

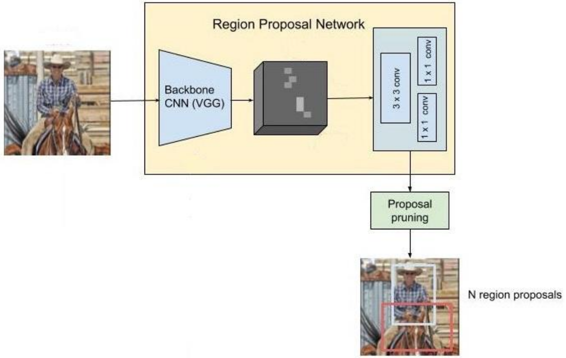
Faster-RCNN

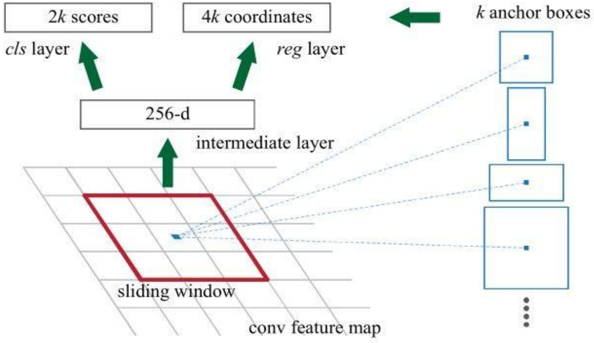
- Contains 2 modules:
    - Regional Proposal Network (RPN): For generating region proposals.
    - Fast R-CNN Network: For detecting objects in the proposed regions.
- Backbone network: ResNet-50
- Sharing the same backbone network's weight
- Pretrained with special training regime on Imagenet dataset



Regional Proposal Network (RPN)

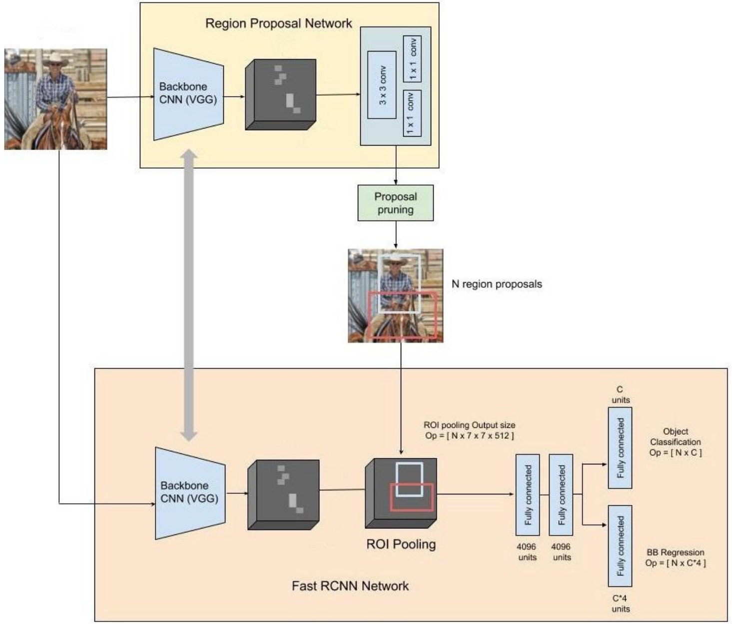
- 3x3x512 Conv. layer is applied on the backbone’s feature map
- For each Conv. window, we have k (k=9) region proposals, each are parameterized to an anchor
    - Filtered into “Positive”/“Negative”/”None” objectness based on IoU with ground truth boxes (Affect training)
    - ”None” objectness region proposal is pruned
- Afterward, apply separately 1x1x2*k and 1x1x4*k Conv. layer as output for the module
    - 2*k output (cls): give probabilities of whether each point in backbone’s feature map contains an object
    - 4*k output (reg): give the 4 regression coefficients of each of the k anchors for every point in backbone’s feature map.
- Proposal pruning:
    - All the boxes are sorted by their cls scores
    - Take top N region proposal (Test: N=1000, Train: N=2000)
    - Apply NMS with threshold = 0.7





Fast-RCNN Network

- The backbone’s feature map is fed to an ROI Pooling layer to extract a fixed-length feature vector from each region proposal.
- Afterward, passed to some FC layers. The output of the last FC layer is split into 2 branches:
    - Softmax layer to predict the class scores
    - FC layer to predict the bounding boxes of the detected objects



Faster R-CNN implementation

- Implemented using Pytorch library
  - Default Weights
  - COCO’s Weights
- Augmented Data
- Resized and Normalized input images (using mean and standard deviation of the ImageNet dataset)
- Optimizer: SGD with Momentum
  - lr = 0.05
  - momentum = 0.9
  - weight decay = 0.05

Loss function

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)$$

- i is the index of the anchor
- p_i^*: 1 for "Positive" objectiveness, 0 for "Negative" objectiveness
- The left term is the log loss over two classes (object vs not object)
    - p_i: output score from the classification branch
- The right term is the regression loss
    - t_i: output prediction of the regression layer
- N_{cls}, N_{reg} are normalizing term, λ are set so that the left and right term are roughly equal

Single Shot MultiBox Detector (SSD)

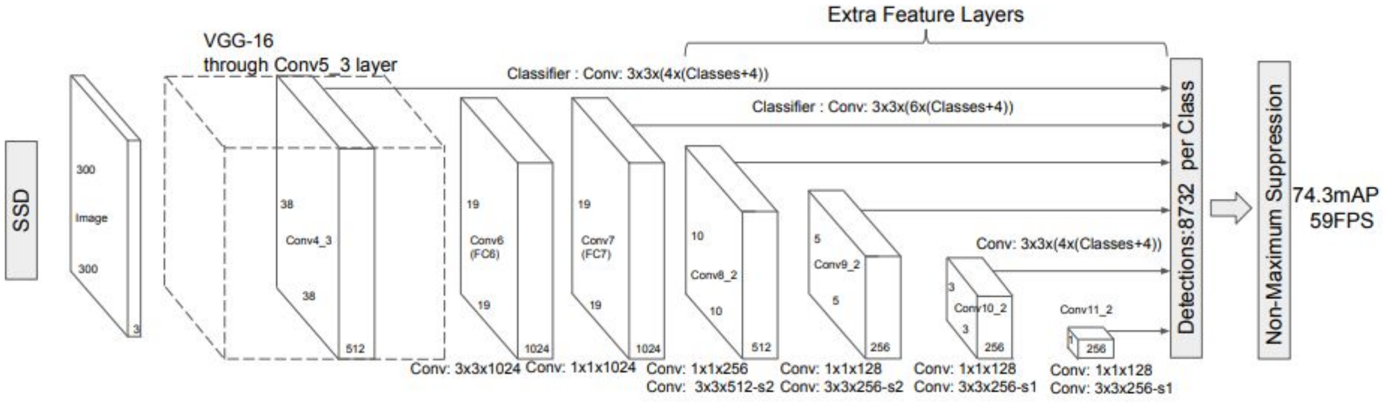
Advantages of SSD:

- Doesn’t utilize sliding windows => Faster computation.
- Utilizes feature maps of multiple scales => Can better detect objects of different sizes
- Doesn’t need a separate Region Proposal Network.
- Only uses 1 Convolutional Deep Neural Network, predicts objects in 1 single pass.

Single Shot MultiBox Detector (SSD)

Model consists of 2 parts:

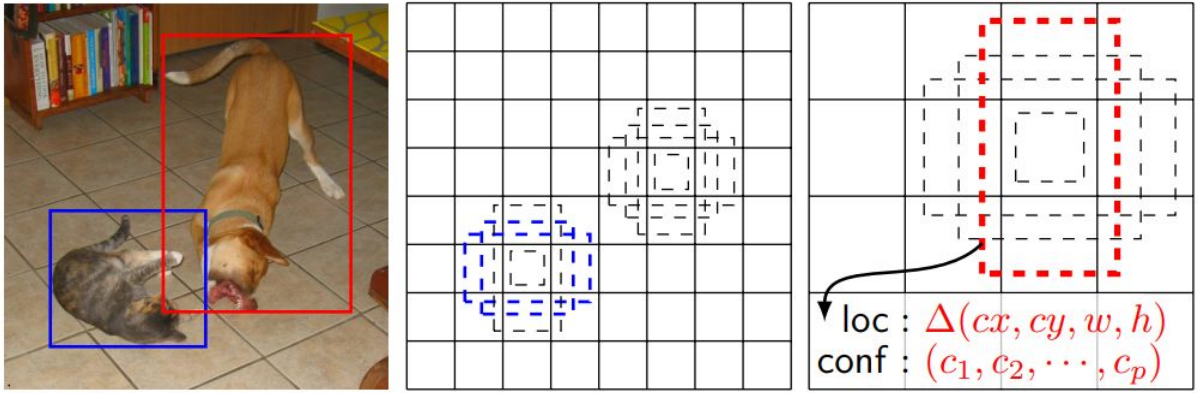
- A truncated backbone network to extract feature maps.
- An auxiliary structure to produce predictions.



Single Shot MultiBox Detector (SSD)

Each feature map is divided into cells. Each cell is associated with a set of default bounding boxes similarly to YOLO's anchors.

The model outputs the offsets for the predicted bounding boxes (i.e. how much the nearest default bounding box must be adjusted to match the prediction), and the per-class confidence scores for each box.



(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

Single Shot MultiBox Detector (SSD)

At training time:

- To match the prediction to the ground truth: use IoU => results in *positive matches* and *negative matches*.

$$L_{conf} = \frac{1}{n_{positives}}\left(\sum\nolimits_{positives} CE\ Loss + \sum\nolimits_{hard\ negatives} CE\ Loss\right)$$

$$L_{loc} = \frac{1}{n_{positives}}\left(\sum\nolimits_{positives} Smooth\ L_1\ Loss\right)$$

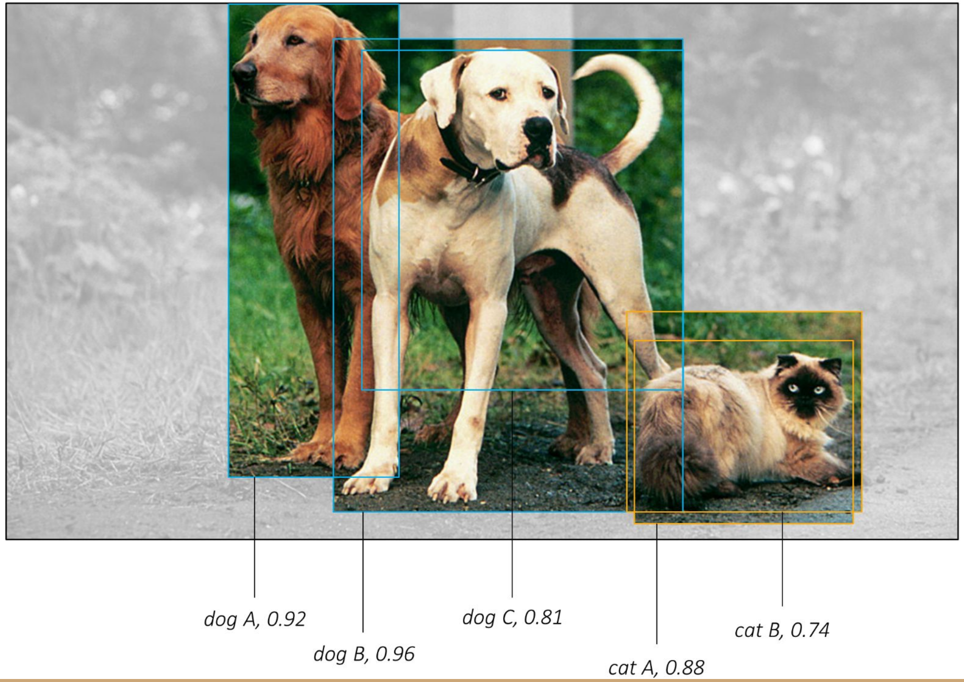
$$L = L_{conf} + \alpha \cdot L_{loc}$$

Single Shot MultiBox Detector (SSD)

At inference time: Non-Maximum Suppression is applied to remove overlapping predictions.

Overlapping predictions are identified using IoU. Among them, only the one with the highest confidence score is kept.

There are usually multiple predictions for the same object



SSD implementation

- Implemented using PyTorch library
- Original vs. Augmented Data
- Resized and Normalized input images (using mean and standard deviation of the ImageNet dataset)
- Optimizer: Adam optimizer
  - lr = 1e-4
  - beta1 = 0.9
  - beta2 = 0.999
  - weight decay = 0.05

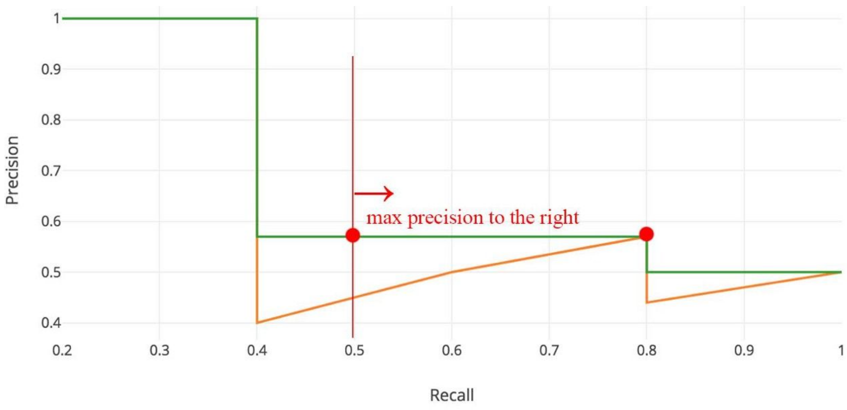
Evaluation and Results

Evaluation metric

Average precision (AP):

- Map each detection to its most-overlapping ground-truth instance
- All of the detection with ≥ 50% IoU with its ground-truth instance are true-positives (TPs), all other detections as false-positives
- Recall: TP detections / ground-truth instances
- Precision: TP detections / All detections
- The PR curve are filled in (interpolated)

=> AP = Area under interpolated PR curve



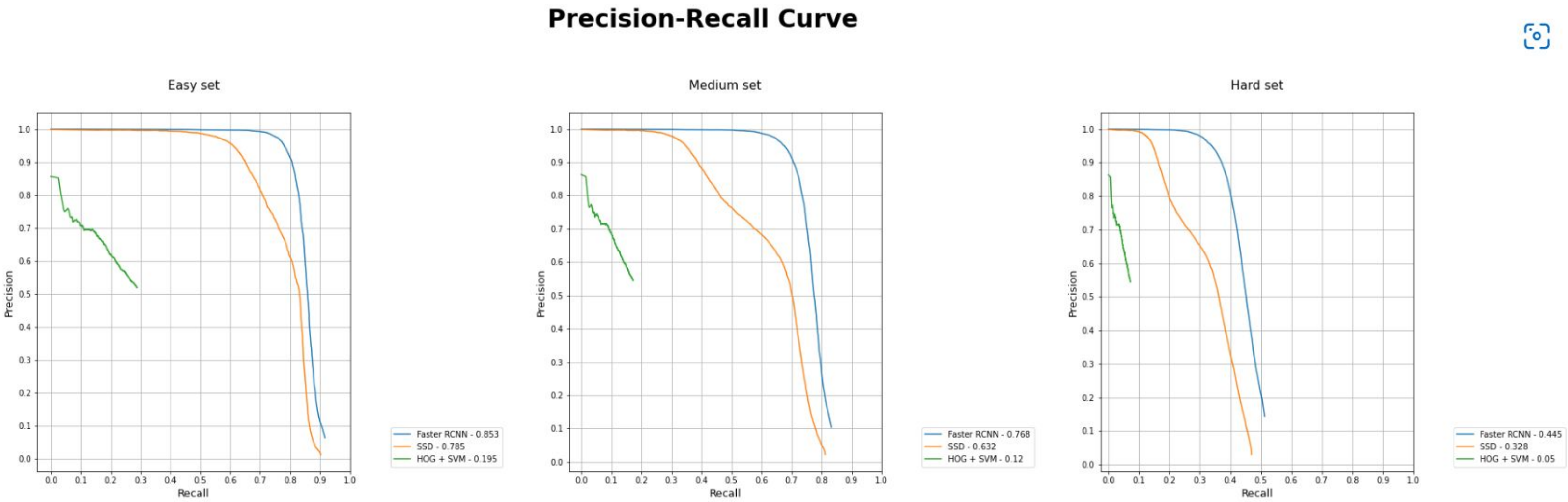
Results

- HOG SVM gets outperformed.
- Faster R-CNN is the best out of the three.

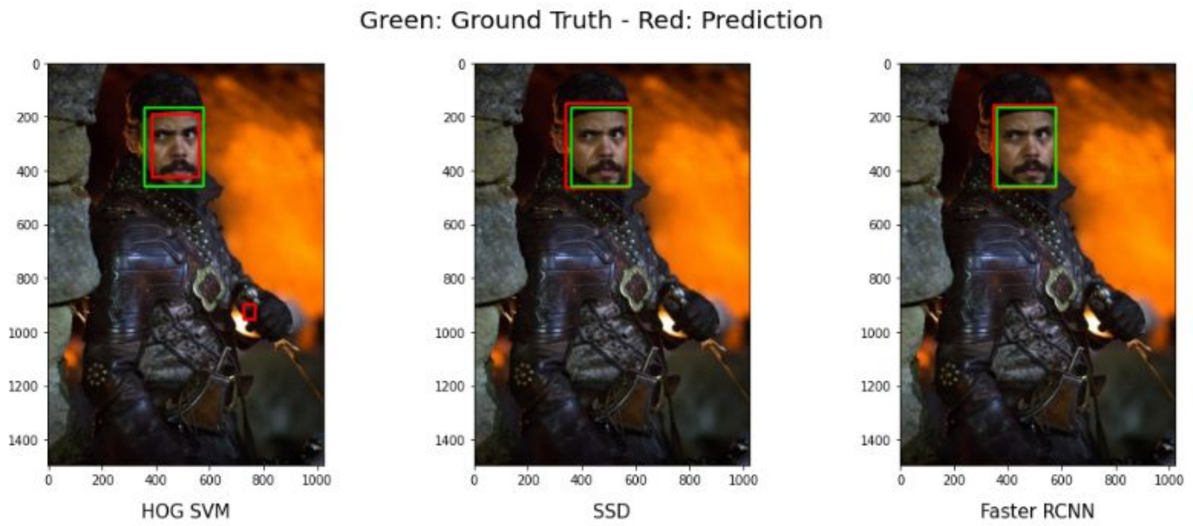
HOG SVM	Easy	Medium	Hard
Baseline	0.004	0.003	0.001
Hard Negative Mining	0.195	0.12	0.05
SSD	**Easy**	**Medium**	**Hard**
No Augmentation	0.744	0.572	0.282
Augmented	0.785	0.632	0.328
Faster R-CNN	**Easy**	**Medium**	**Hard**
Default Weights	0.846	0.758	0.442
COCO’s Weights	**0.853**	**0.768**	**0.445**

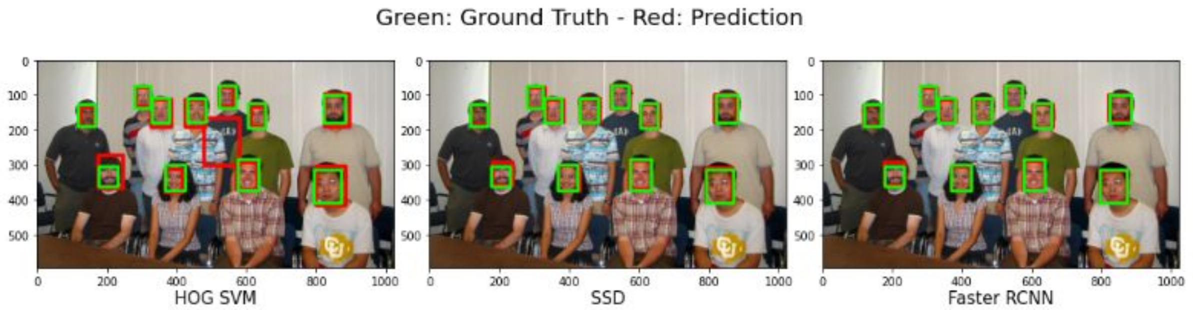
Results

The dataset split up the official test set and validation set into 3 subsets based on the detection rate of EdgeBox: Easy, Medium and Hard. In our case the official validation set is our test set. We will compare our model performance on these 3 subset.

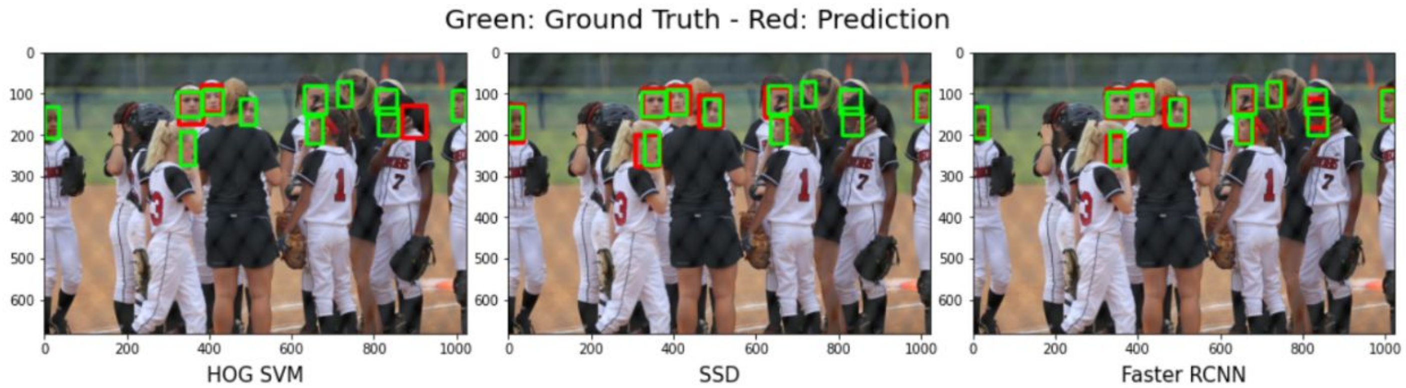


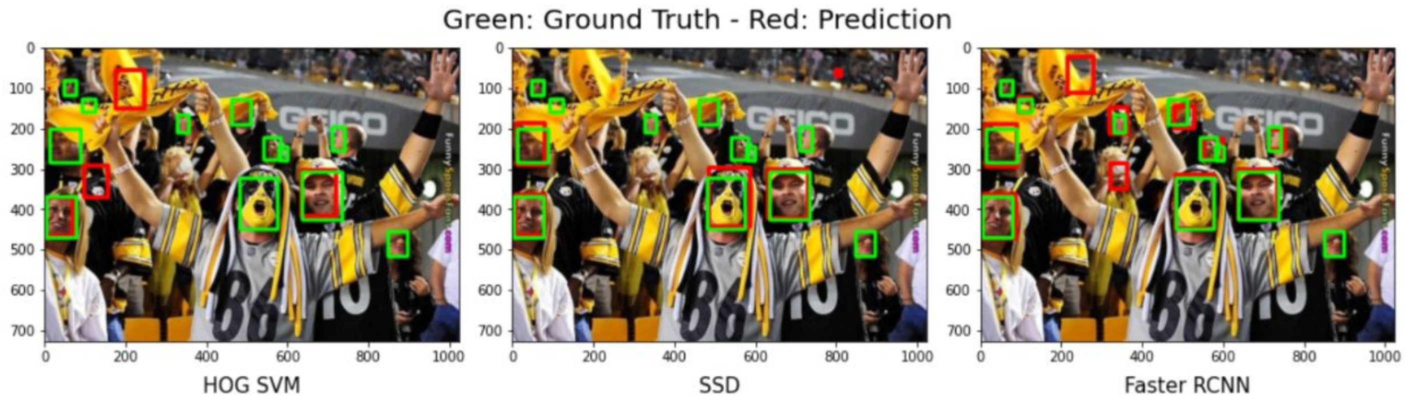
Results: Easy



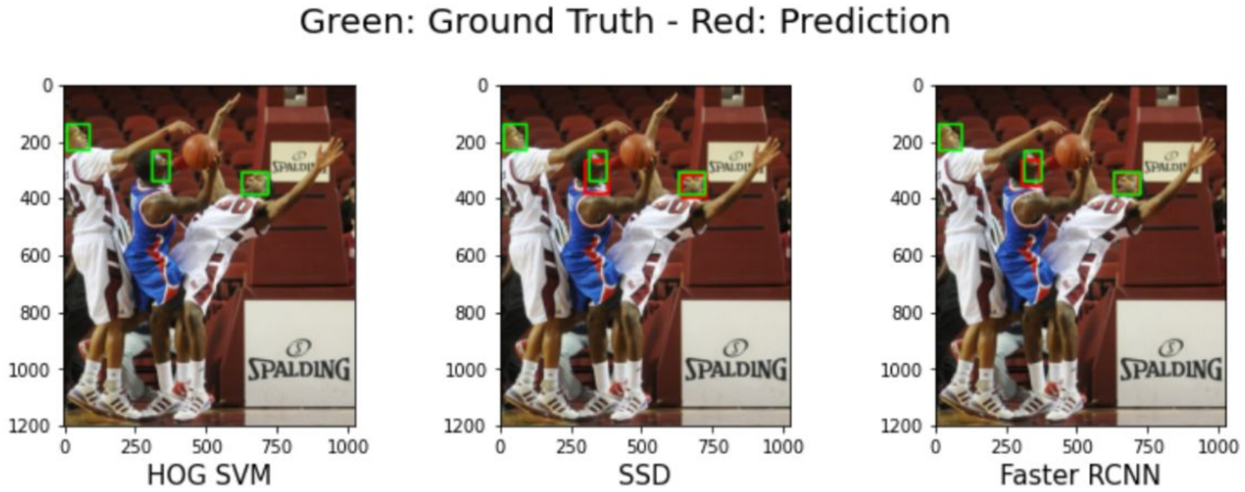


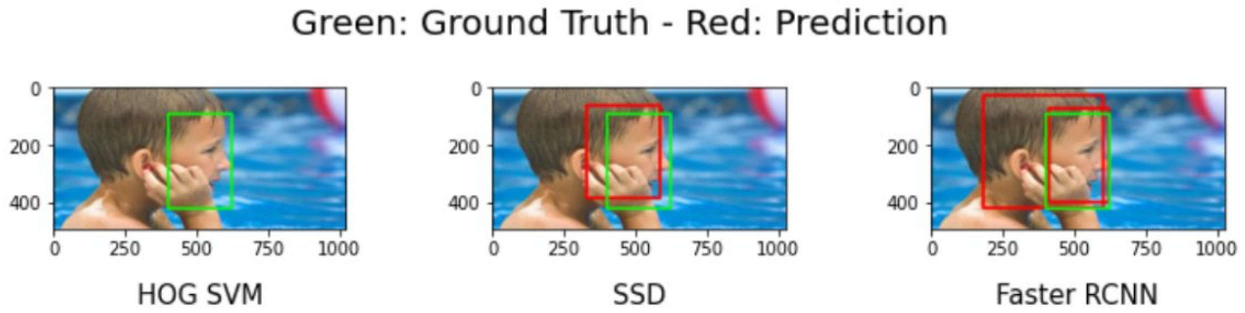
Results: Medium





Results: Hard





Thank you!