Proofpoint

Capstone Update and Object Detection Literature Review

1st October 2020

Duke

Annotations

- Annotated approx. 870 images from approx. 470 companies.
- Issues with some images (approx. 70 or 7.5% of screenshots):
  - Access denied
  - Darkened/blocked logos due to popup boxes (accept cookies)
  - Blank screenshot
  - Wikipedia page (no logo or logo blocked)
  - No image (weblink broken)

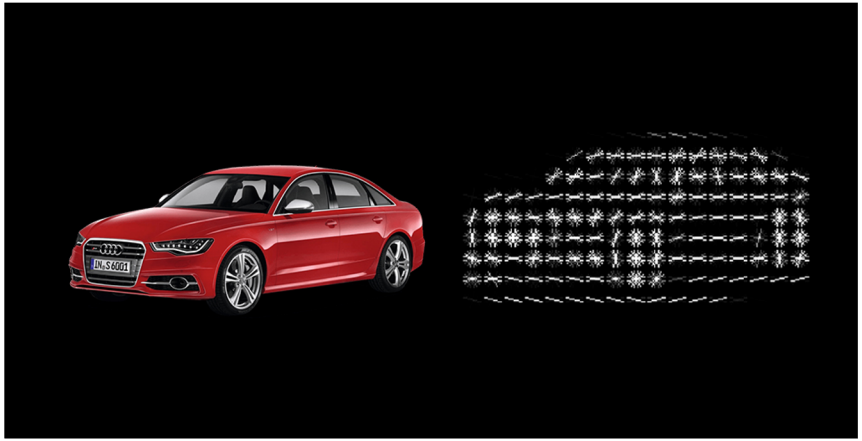
Duke

Literature Review: Object Detection

Duke

Context

- Previous object detection used methods such as HOG and SIFT
  - Extract features from image using gradients
  - Computationally expensive
  - Ensemble models
- Object detection requires better localization within the image
- Use of deep learning (CNNs) can provide more efficient feature extraction
  - Improved object detection

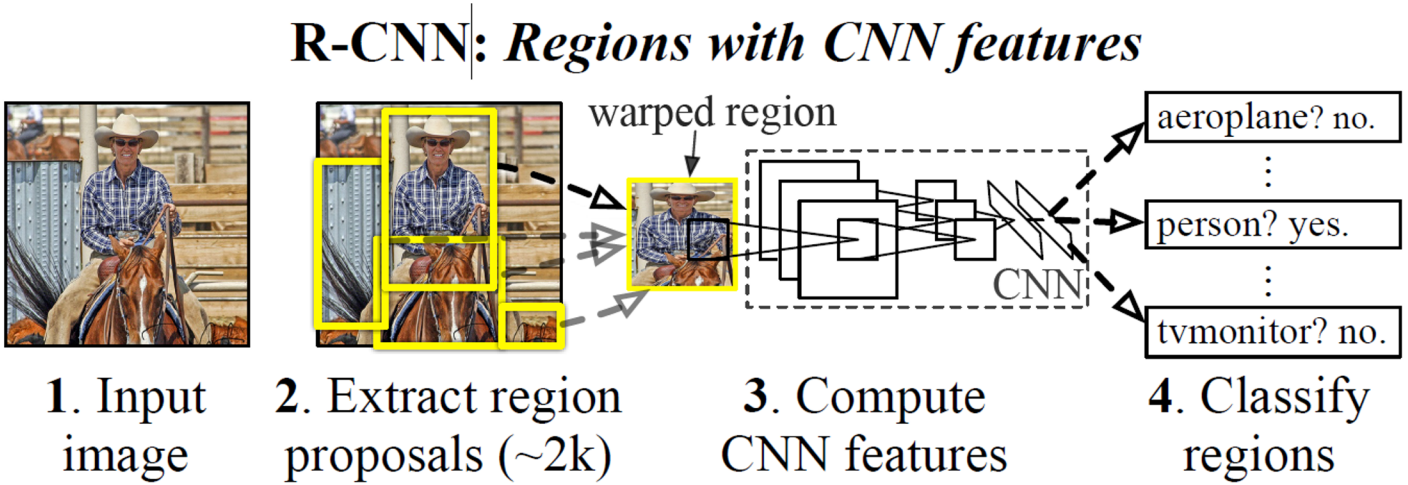


Duke

Part 1: R-CNN - Region Based Convolutional Neural Networks

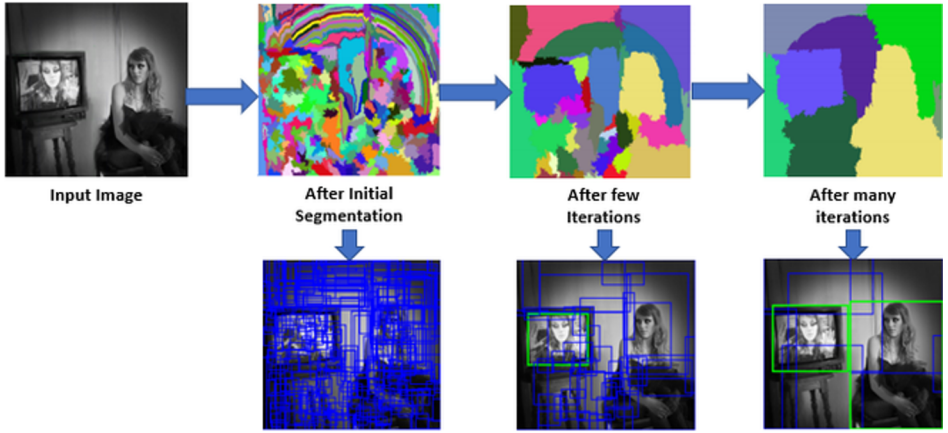
Duke

R-CNN (2014) Overview



Key Network Features

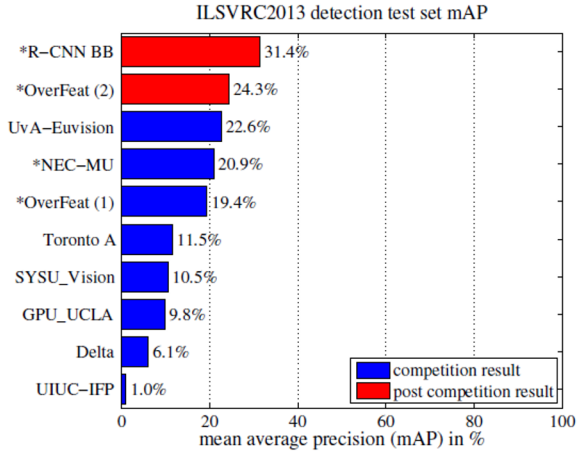
- Uses selective search for generating region proposals
  - Can work with other methods
- Efficiency
  - CNN parameters shared across all classes
  - Low dimensional feature vectors compared to prior models (360k vs 4k dimensions)
  - Several orders of magnitude faster

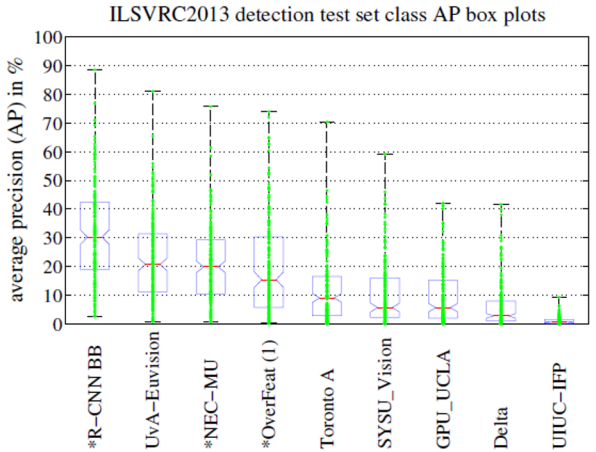


Duke

Results

VOC 2010 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
DPM v5 [20]†	49.2	53.8	13.1	15.3	35.5	53.4	49.7	27.0	17.2	28.8	14.7	17.8	46.4	51.2	47.7	10.8	34.2	20.7	43.8	38.3	33.4
UVA [39]	56.2	42.4	15.3	12.6	21.8	49.3	36.8	46.1	12.9	32.1	30.0	36.5	43.5	52.9	32.9	15.3	41.1	31.8	47.0	44.8	35.1
Regionlets [41]	65.0	48.9	25.9	24.6	24.5	56.1	54.5	51.2	17.0	28.9	30.2	35.8	40.2	55.7	43.5	14.3	43.9	32.6	54.0	45.9	39.7
SegDPM [18]†	61.4	53.4	25.6	25.2	35.5	51.7	50.6	50.8	19.3	33.8	26.8	40.4	48.3	54.4	47.1	14.8	38.7	35.0	52.8	43.1	40.4
R-CNN	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	50.2
R-CNN BB	**71.8**	**65.8**	**53.0**	**36.8**	**35.9**	**59.7**	**60.0**	**69.9**	**27.9**	**50.6**	**41.4**	**70.0**	**62.0**	**69.0**	**58.1**	**29.5**	**59.4**	**39.3**	**61.2**	**52.4**	**53.7**





Duke

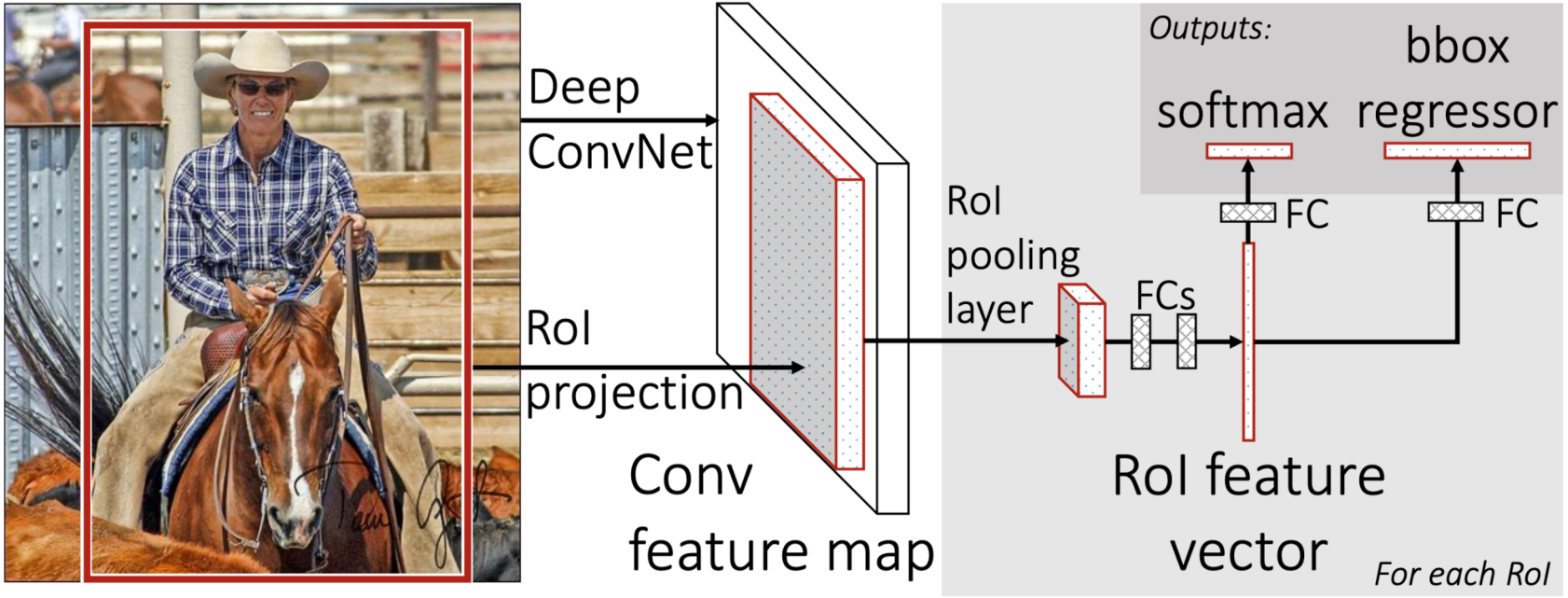
Limitations of R-CNN

- Major limitations of R-CNNs:
  1. Training is a multi-stage pipeline
  2. Training is expensive in space and time
  3. Object detection is slow (47 seconds on test data)
- R-CNN is slow primarily because it performs a convolutional network forward pass for each object proposal, without sharing computation.
- R-CNN also generates 2,000 region proposals per full image making it slow to train and test.

From R-CNN to Fast R-CNN

- Fast R-CNNs improve on R-CNNs in terms of space, time, and accuracy.
  1. Training is single-stage, using a multi-task loss function
  2. Training can update all network layers
  3. No disk storage is required for feature caching
  4. Higher detection quality (mean average precision) than R-CNNs
  5. Comparable performance to SPPnet (immediate successor to R-CNNs)
- Generates region proposals on the convolutional feature maps instead of on the full image. This drastically reducing training time.

Fast R-CNN Architecture



* RoI: Region of Interest
* bbox: Bounding Box
* FC: Fully Connected

Duke

Information Sharing

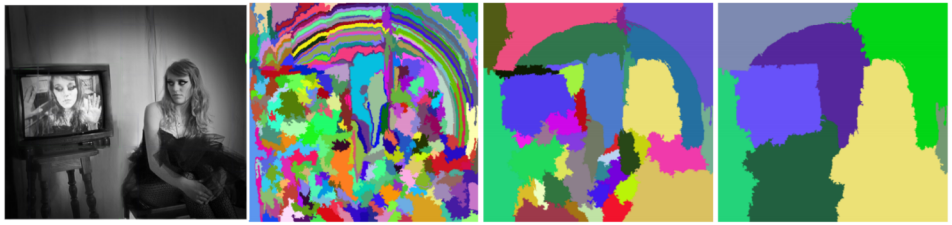
- Both the regressor and classifier share two fully connected layers.
- Information is shared between both parts of the models.
- Learning happens sequentially due to the multi-task loss function.
- Improves performance overall and makes softmax predictions marginally better the R-CNN's support vector machine with big gains in speed.

Fast R-CNN Benchmark Performance Gains

- Trains very deep VGG16 network 9x faster than R-CNN and is 213x faster at test time.
- 65.7% mAP on PASCAL VOC 2012 compared to 62% achieved by R-CNN.
- Performance in general is better than R-CNN and comparable to SPPnet but training and testing is much faster than both of these methods.

Drawbacks of Fast R-CNN

- Test Region Proposal is slow
  - Selective Search (2s/img)
  - EdgeBoxes (0.2s/img)



Input Image | Initial Segmentation | After some iterations | After more iterations

- Despite Fast-RCNN giving real-time object detection, obtaining region proposals in the beginning is a ‘bottleneck’

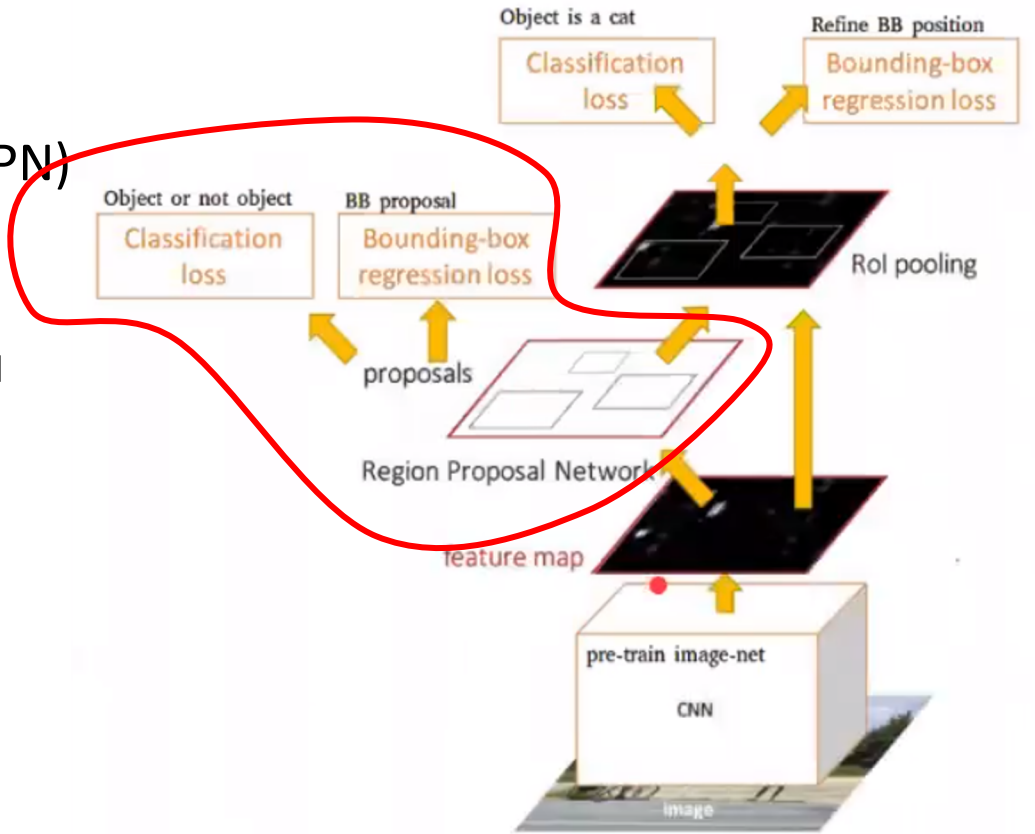
- Region Proposal Time ~ Detection Time

Duke

Faster R-CNN (2016)

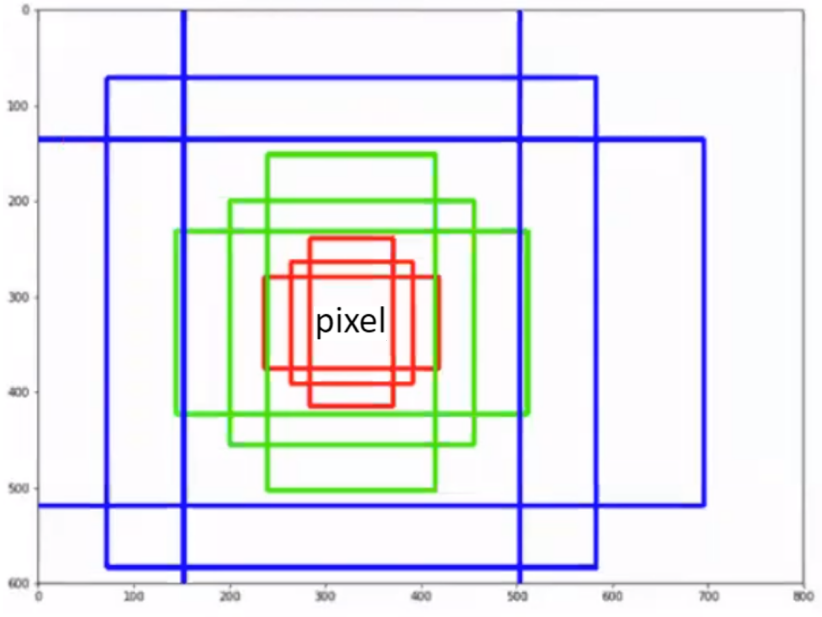
- Region Proposal Network (RPN)
  - **Initial Guess**
  - Fully Convolutional Network
  - Outputs :
    - Rectangular Object Proposal
    - Objectness Score

- **RPN + Fast R-CNN**



How does RPN Work?

- Anchor Boxes
  - **Initial Initial guess** (Input to RPN)
  - Anchors are shapes and aspect ratios we place at every pixel on image
  - They are not meant to be close to the ground truth bounding box. They are just a starting point **proposal.** Hence Region Proposal Network (RPN)



k = *9* anchor boxes per pixel

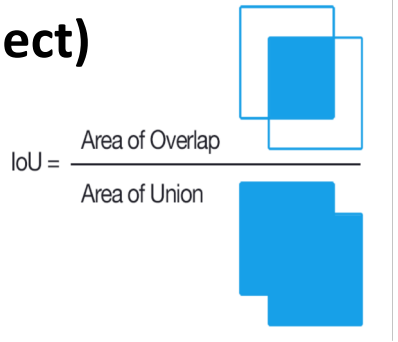
Total Anchor Boxes = W * H * k

- **RPN prunes anchor boxes**

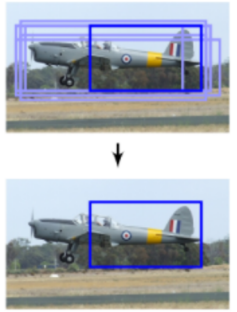
Duke

How does RPN Prune Anchor Boxes?

- Gives binary label to each anchor box **(Object or No Object)**
  - **Condition for positive label:**
    - Anchor with an IoU > 0.7 with the ground-truth box
  - **Condition for negative label:**
    - Anchor with an IoU < 0.3 with the ground-truth box

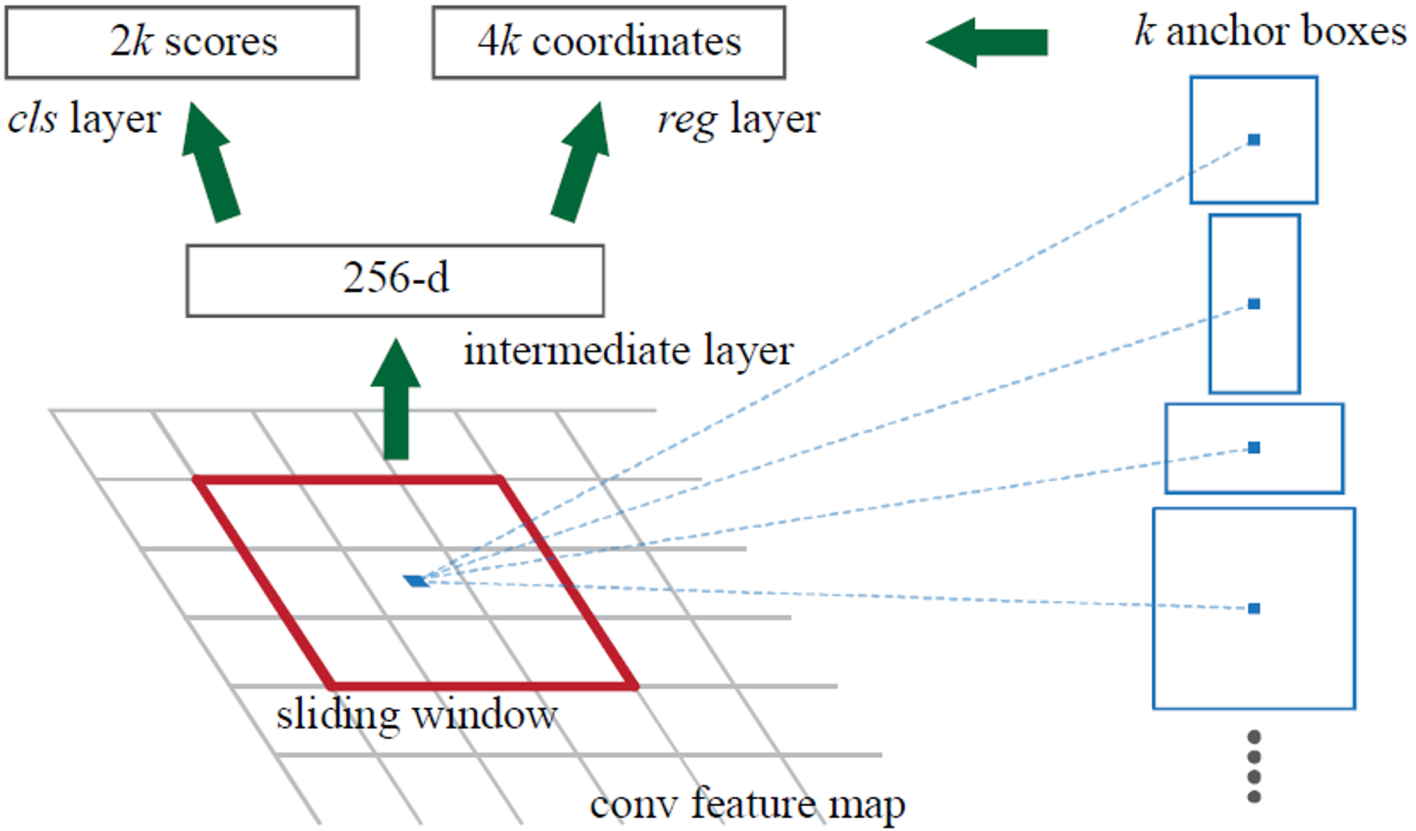


- Single ground truth box can give positive label to multiple anchor boxes



- Non-Maximal Suppression (NMS)

Duke


2k scores
4k coordinates
k anchor boxes
cls layer
reg layer
256-d
intermediate layer
sliding window
conv feature map

Selective Search (2s/img)

EdgeBoxes (0.2s/img)

RPN (0.01s/img)

Potential Drawbacks of RCNNs

- Too many hyper parameters to choose(k, IoU threshold etc.)
- Requires training different part of the networks sequentially (aka freezing weights)
- Slow (~5 fps)

Part 2: YOLO
You Only Look Once

Duke

You Only Look Once (YOLO)

- YOLO - Single Regression Problem
  - Pixels ⇨ Bounding Box + P(Class)

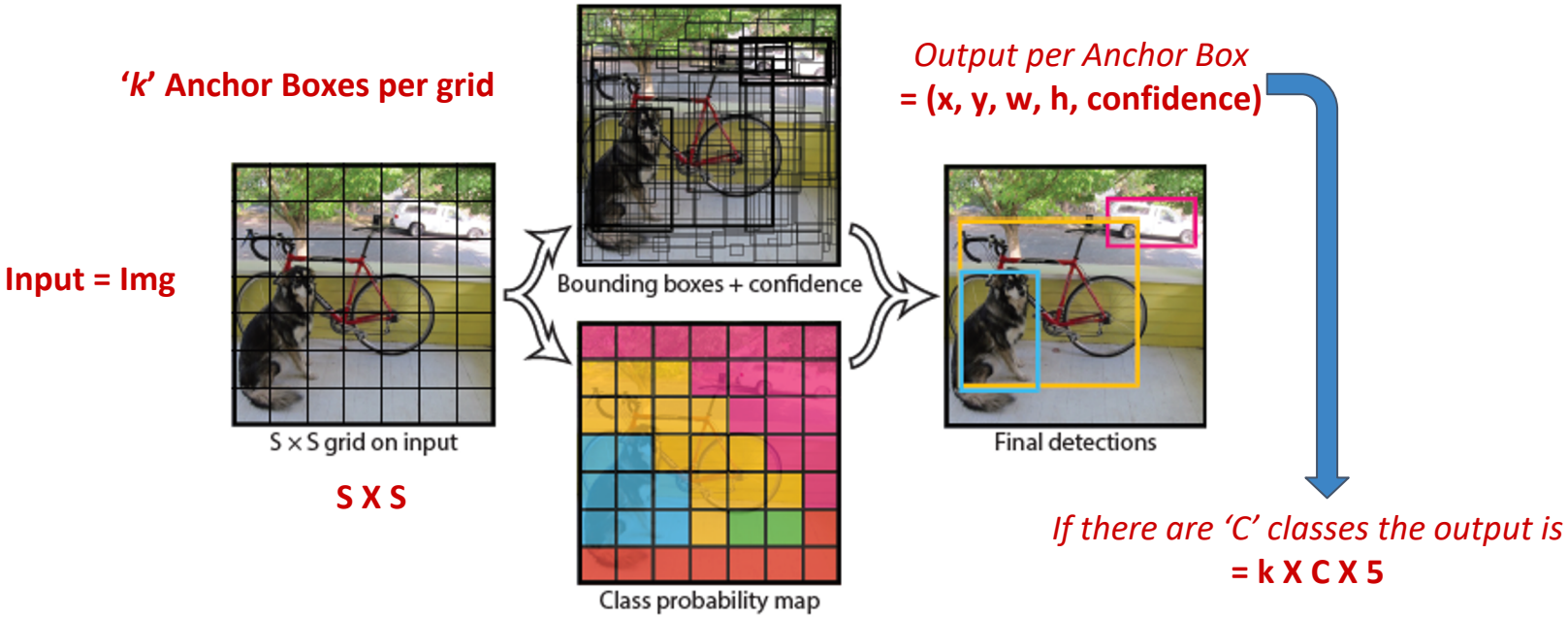
- Very Fast (~45 fps)

- Sees entire image - less likely to misclassify a background as object (drawback of Fast R-CNN)

- Predicts bounding boxes across all classes

Duke

Unified Detection



Duke

Results - YOLO Vs RCNN

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

Table 1: Real-Time Systems on PASCAL VOC 2007. Comparing the performance and speed of fast detectors. Fast YOLO is the fastest detector on record for PASCAL VOC detection and is still twice as accurate as any other real-time detector. YOLO is 10 mAP more accurate than the fast version while still well above real-time in speed.

Duke

Results - Combined Fast R-CNN & YOLO

	mAP	Combined	Gain
Fast R-CNN	71.8	-	-
Fast R-CNN (2007 data)	**66.9**	72.4	.6
Fast R-CNN (VGG-M)	59.2	72.4	.6
Fast R-CNN (CaffeNet)	57.1	72.1	.3
YOLO	63.4	**75.0**	**3.2**

Table 2: Model combination experiments on VOC 2007. We examine the effect of combining various models with the best version of Fast R-CNN. Other versions of Fast R-CNN provide only a small benefit while YOLO provides a significant performance boost.

YOLO has fewer false positives...

Duke

Limitations of YOLO

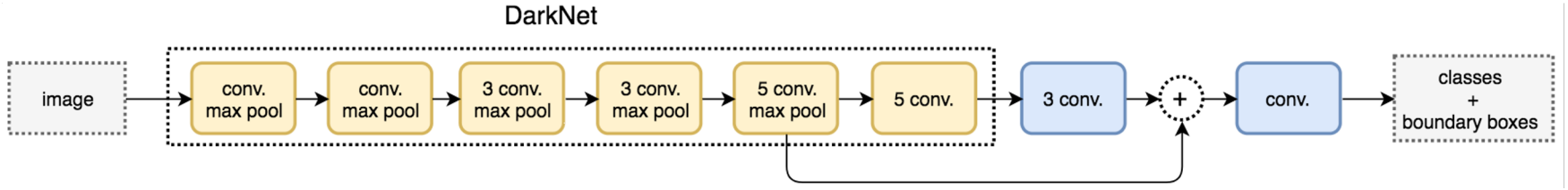
- Lags behind state-of-the-art algorithms in terms of detection accuracy. Faster R-CNN has 73.2 mAP on PASCAL 2007 while YOLO has 63.4 mAP.
- Struggles to identify smaller objects (which may be an issue in terms of logo identification).
- Produces more localization errors than Fast R-CNN.
- Finds it difficult to detect objects of varying sizes.

From YOLO to YOLO v2

- **Higher Resolution Classifier**: Train on ImageNet at 224x224 but resize to 448x448 for additional training. +3.5% mAP.
- **Anchor Boxes**: use k-means to generate dimension clusters. Predict location coordinates (instead of offsets) from dimension clusters rather than hand picked anchor boxes. +5% mAP.
- **Multiscale Training**: training at different aspect ratios. +1.5% mAP.
- **Darknet 19**: maintains YOLO's speed but improves mAP. +0.4% mAP.
- **Batch Normalization**: stabilizes training. +2% mAP.
- **YOLO9000**: training is done jointly for detection and classification using both COCO and Imagenet. Model learns to predict novel detection labels due to its hierarchical structure.

YOLO v2 Architecture

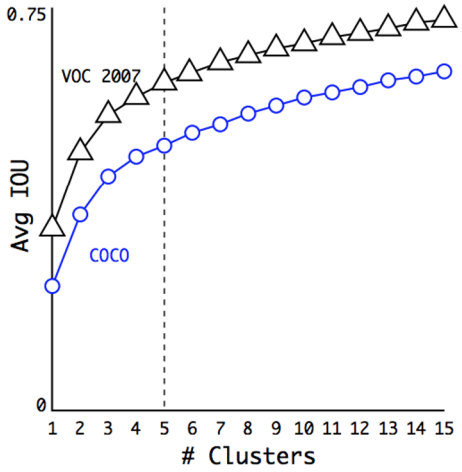
- Fully Convolutional.
- Moves class prediction from cell level to boundary box level.
- Removes one max pooling layer to make spatial output of the network 13x13 instead of 7x7.
- Uses passthrough to help with small object detection.

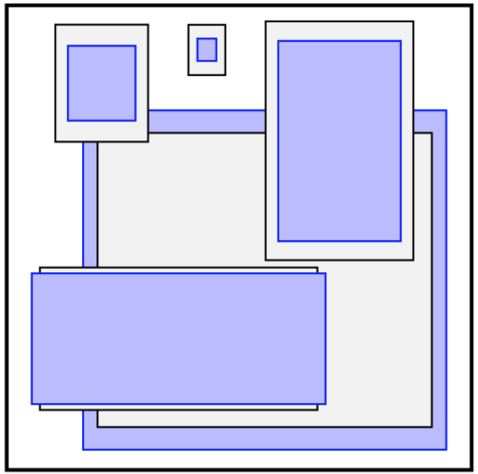


Duke

Anchor Boxes Using K-Means Clustering

- Cluster the bounding boxes in the training data using K-Means.
- Use the clusters to obtain K anchors.
- Predict location coordinates instead of offsets.
- Use custom distance metric to ensure bounding box size does not affect clusters.
- YOLO v2 goes with 5 clusters to balance Avg IOU and prediction time.





Duke

PASCAL VOC 2007 Performance

Detection Frameworks	Train	mAP	FPS
Fast R-CNN [5]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[15]	2007+2012	73.2	7
Faster R-CNN ResNet[6]	2007+2012	76.4	5
YOLO [14]	2007+2012	63.4	45
SSD300 [11]	2007+2012	74.3	46
SSD500 [11]	2007+2012	76.8	19
YOLOv2 288 × 288	2007+2012	69.0	91
YOLOv2 352 × 352	2007+2012	73.7	81
YOLOv2 416 × 416	2007+2012	76.8	67
YOLOv2 480 × 480	2007+2012	77.8	59
YOLOv2 544 × 544	2007+2012	**78.6**	40

Duke

Limitations of YOLOv2

- Was very high-performing until development of models such as RetinaNet
- Lacked newer features such as residual blocks, shortcuts, feature detection at various scales
- Still struggles with smaller objects

Duke

YOLOv3

- Bounding boxes
  - Uses dimension clusters as anchor boxes (like YOLOv2)
  - Predicted using objectness score from logistic regression
- Independent logistic classifiers for each bounding box (multilabel)
  - Improves performance over softmax classifier
- Prediction at three different scales
- Feature extractor
  - Uses Darknet-53 as backbone → Darknet-19 and residual connections

Darknet-53 Architecture

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Table 1. **Darknet-53.**

Duke

Neural Network Backbone Comparison

Backbone	Top-1	Top-5	Bn Ops	BFLOP/s	FPS
Darknet-19 [15]	74.1	91.8	7.29	1246	**171**
ResNet-101[5]	77.1	93.7	19.7	1039	53
ResNet-152 [5]	**77.6**	**93.8**	29.4	1090	37
Darknet-53	77.2	**93.8**	18.7	**1457**	78

Table 2. **Comparison of backbones.** Accuracy, billions of operations, billion floating point operations per second, and FPS for various networks.

Duke

Results on COCO Dataset

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [15]	DarkNet-19 [15]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [9]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608×608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Duke

YOLO v3 Summary

- Improvements
  - Better mAP on smaller objects
  - Reduced localization errors
  - Better prediction at different scales/aspect ratios
- Benefits
  - Faster than methods such as RetinaNet
  - More accurate than YOLO v2
- Limitations
  - Performance is not as strong on medium and large objects
  - Higher IOU thresholds reduce performance, so perfect alignment of bounding boxes is very difficult

Part 3: YOLO vs R-CNN

Duke

Faster R-CNN

1. Versatility (2-stage)
1. Higher accuracy than YOLO
1. Seems to evolving

YOLO v3

1. Faster than Faster R-CNN
1. Dimension clusters for logos
1. Comparable performance to Faster R-CNN

Alternative: We could also combine YOLO with RCNNs

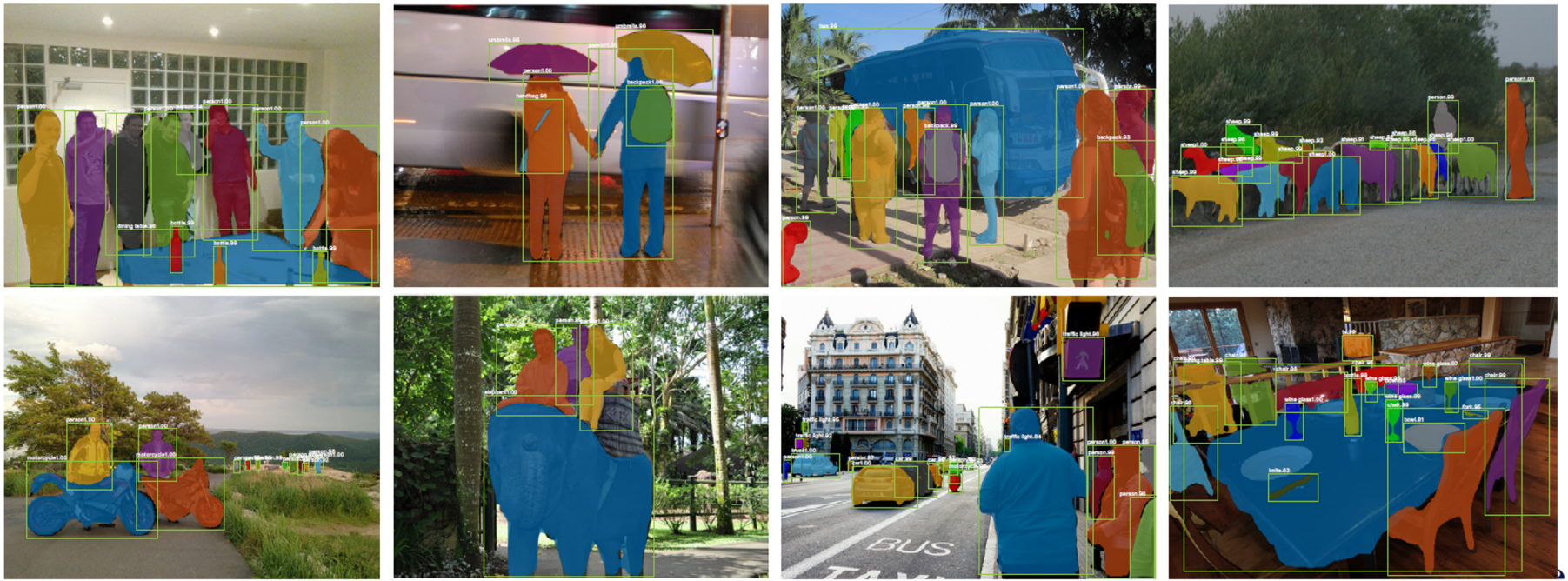
Duke

Part 4: Miscellaneous Algorithms

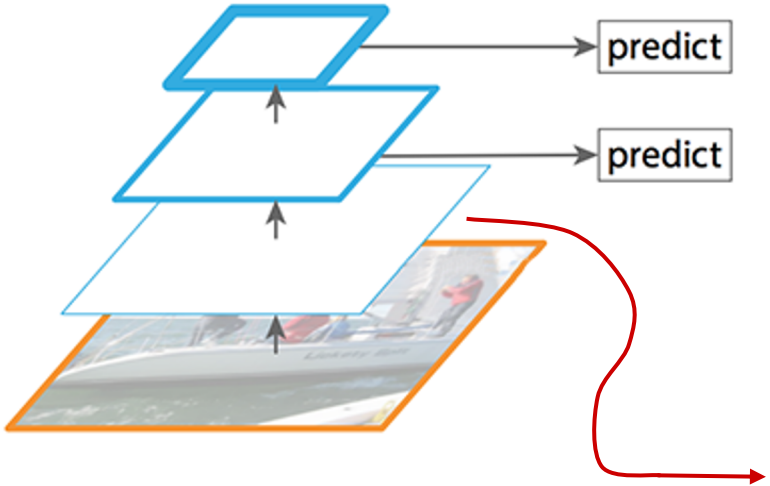
Duke

Mask R-CNN (2018)

- Not viable for our problem



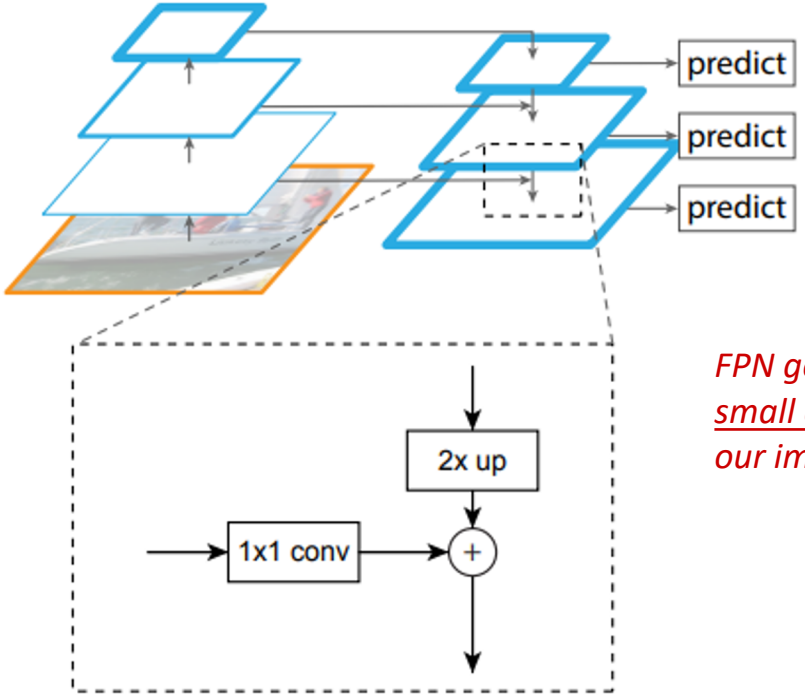
Single Shot Detector (SSD)



Bottom Layers not included. High resolution, but apparently low value

Duke

Feature Pyramid Network (FPN)



FPN good when we have small and big objects in our images

Duke

Challenge is Small Object Detection

Duke

Next Steps

Duke

More Annotations +
Open Set Learning OR Small Object Detection

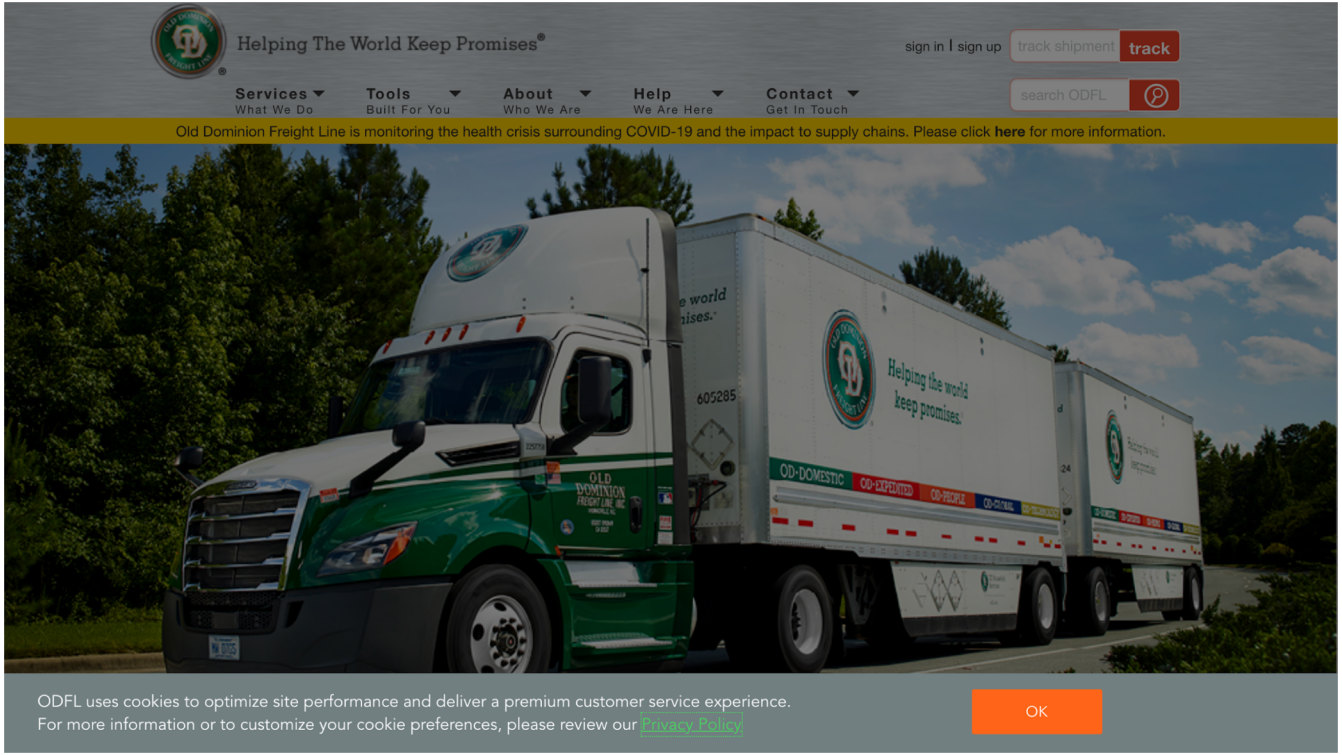
- We intend to scrape and annotate 600-1000 additional images in the next two weeks.
- We intend to do a literature review of Open Set Learning OR Small Object Detection. Which should we prioritize?
- Update emails or messages?

Duke

Appendix

Duke

Example of Darkened Image



Duke

References

1. Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.
2. R. Girshick, Fast R-CNN, IEEE International Conference on Computer Vision (2015), pp. 1440-1448
3. S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, Adv. Neural Inf. Proces. Syst. (2015), pp. 91-99
4. J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.
5. J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.
6. Redmon, Joseph, and Ali Farhadi. "Yolov3: An incremental improvement." *arXiv preprint arXiv:1804.02767* (2018).
7. K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask R-CNN, IEEE Trans. Pattern Anal. Mach. Intell., 42 (2) (2020), pp. 386-397
8. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S.E. Reed, C.-Y. Fu, A.C. Berg, SSD: single shot multibox detector, European Conference on Computer Vision (2016), pp. 21-37
9. Redmon, Joseph, and Ali Farhadi. “YOLO9000: Better, Faster, Stronger.” *ArXiv:1612.08242 [Cs]*, December 25, 2016. http://arxiv.org/abs/1612.08242

Duke