Comparative Evaluation of Human and Object Detection Models on Low-light Images

By Group A: Anket Sah, Amala Chirayil, Kriti Gupta, Ksheeraj Vepuri, Sanmesh Bhosale

CS 256

11/04/19

Outline

➢ Architecture Diagram

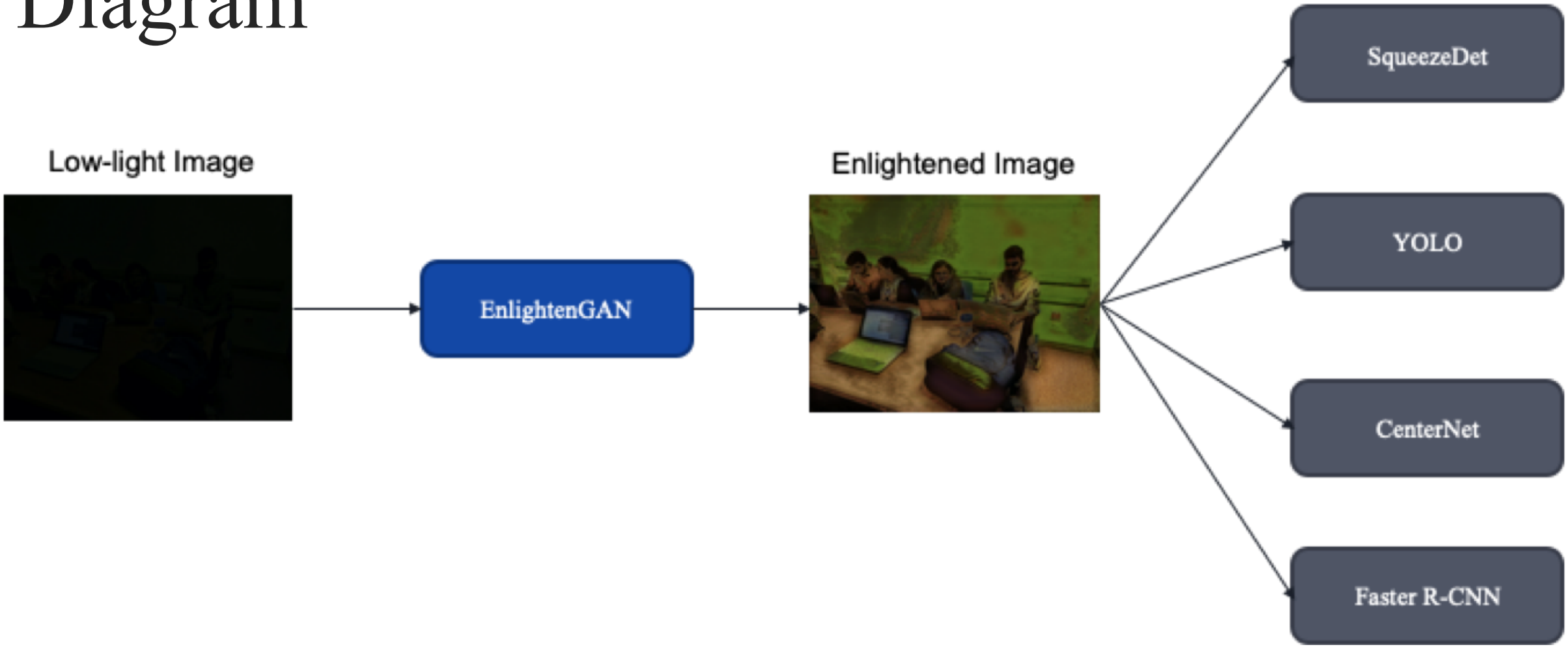
➢ Low-light image results of the following models:

- SqueezeDet
- YOLO
- Faster R-CNN
- CenterNet

➢Comparison between the models based on:

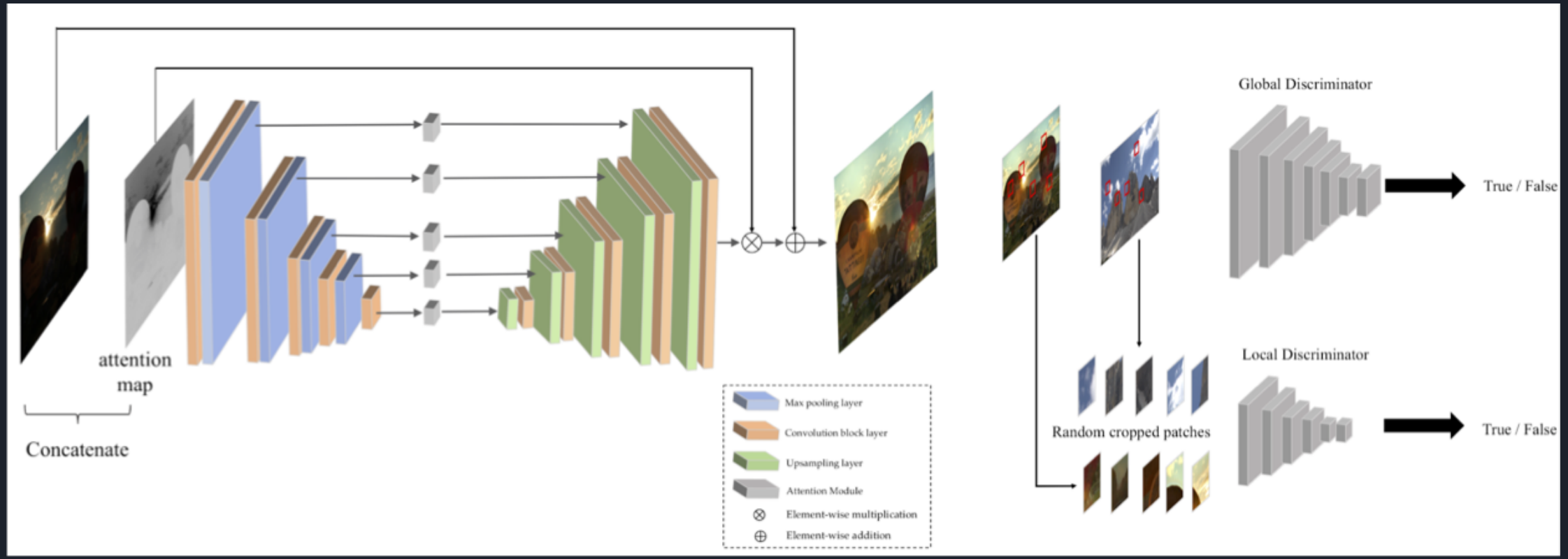
- Detection speed
- Accuracy obtained for low-light images

Architecture Diagram



EnlightenGAN

EnlightenGAN Architecture



EnlightenGAN Implementation Details

➢ Tools used:

Amazon EC2 Ubuntu Instance of type p2.8xlarge with specifications:

- GPUs: 8
- vCPUs: 32
- Mem (GiB): 488
- GPU Memory (GiB): 96
- Network Performance: 10 GB

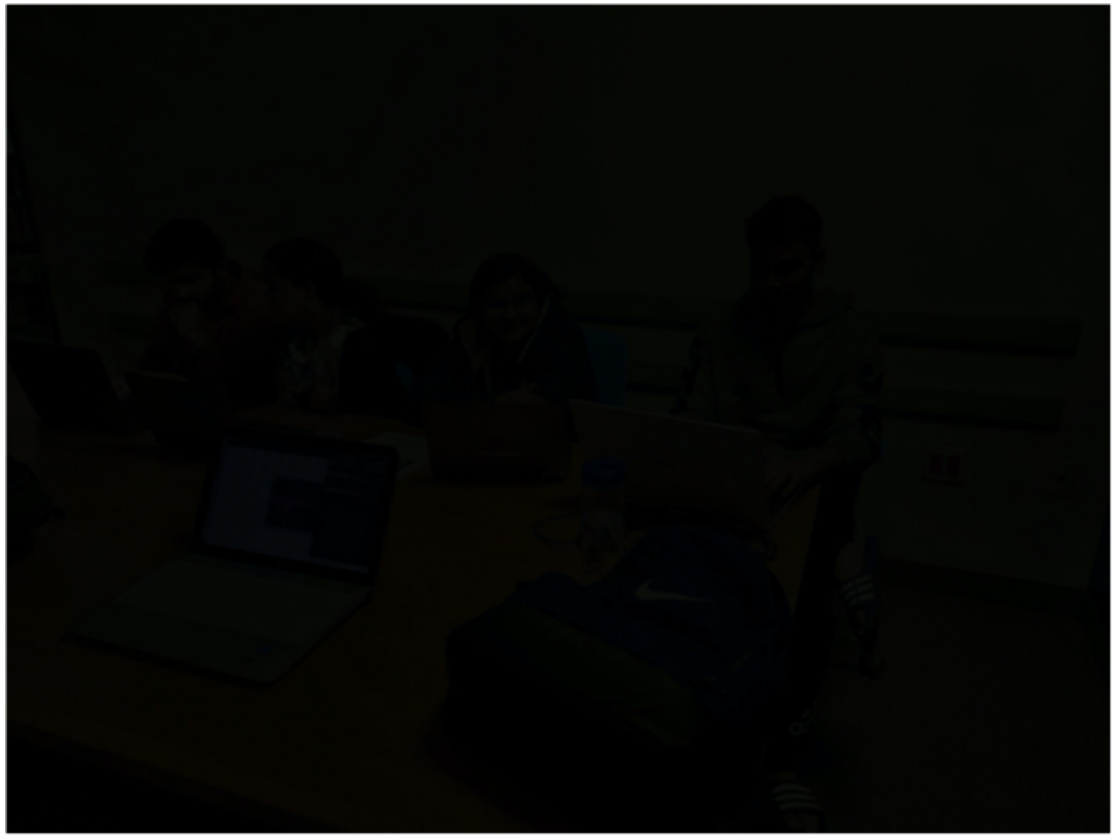
Software Used

- Torch 0.3.1
- Torchvision 0.2.0
- Visdom
- Dominate

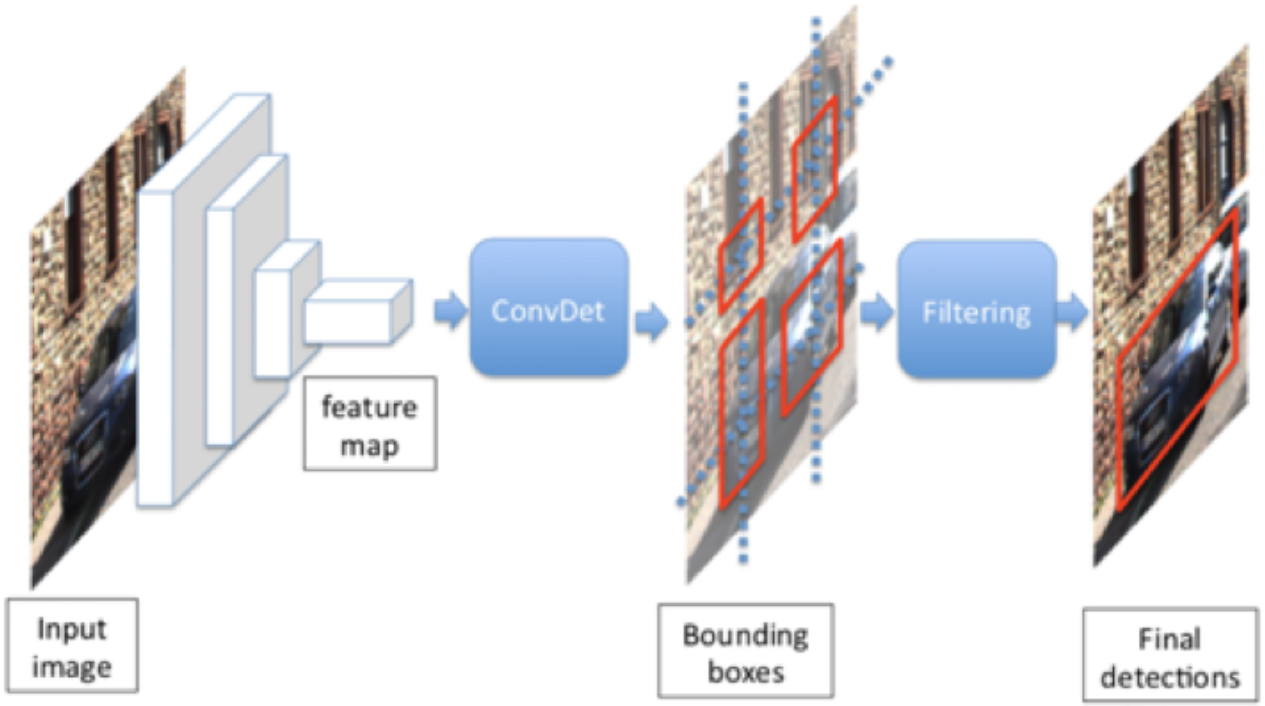
➢Code Reference: https://github.com/TAMU-VITA/EnlightenGAN

Low-light Training Data Image Result:

Low-light Image Result:



SqueezeDet:CNN for Autonomous Driving



Dataset - KITTI

Dataset specification

- Object categories - 8
- Images - 7520 (7483 labeled)

Training SqueezeDet on KITTI

- Object categories - 3 (Car, Cyclist, Pedestrian)
- Training images - 3691
- Validation images - 3690

Implementation Details

➢ AWS EC2 Instance with Deep Learning Base AMI (Ubuntu 16.04) and version 20.0 with p2.xlarge GPU instance was setup in order to run this model

- p2.xlarge specifications
  - GPU -1
  - vCPUs - 4
  - RAM - 61GB

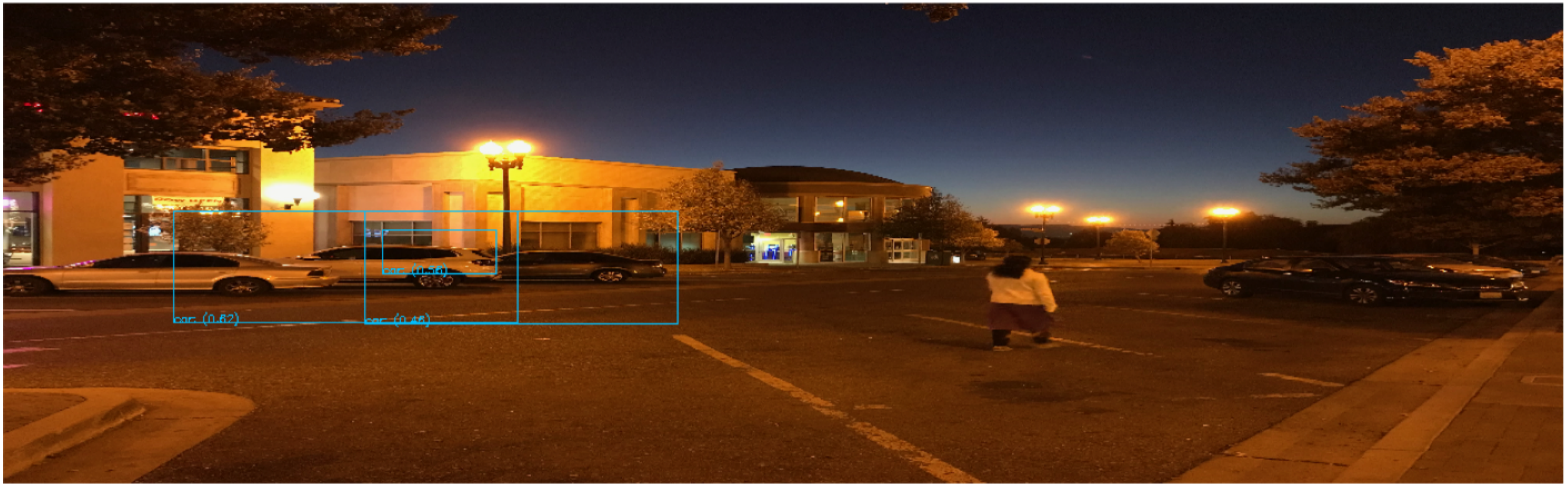
➢ Software specifications

- easydict==1.6
- joblib==0.10.3
- numpy==1.12.0
- opencv-python==3.2.0.6
- Pillow==4.0.0
- tensorflow-gpu==1.0.0

➢Code Reference: https://github.com/BichenWuUCB/squeezeDet

Low-light Image Result

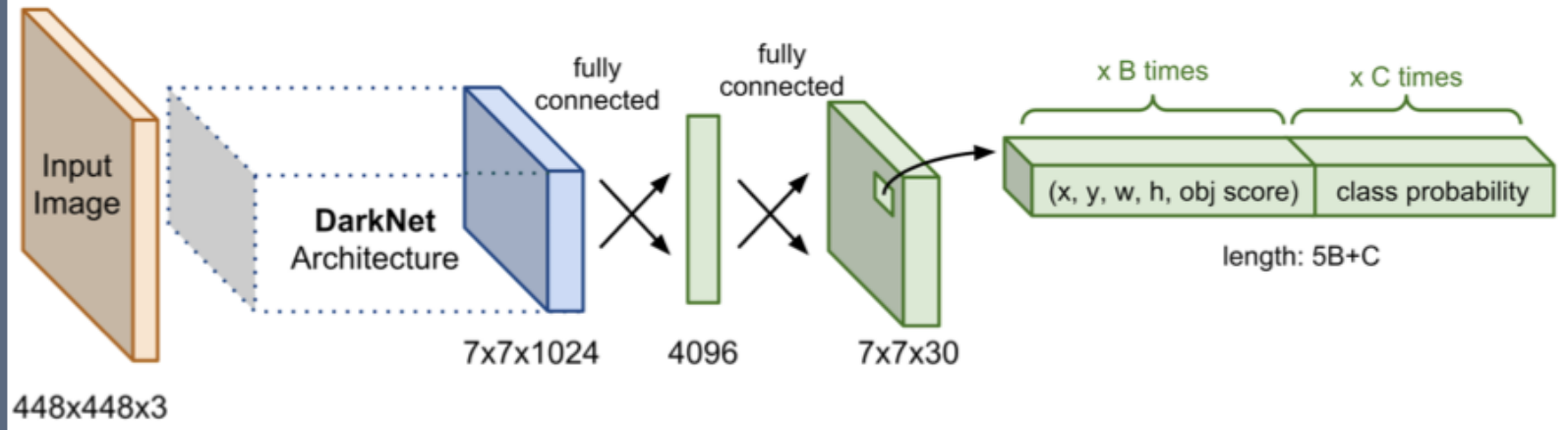
Enlightened Image Result



Low-Light Image Result

Enlightened Image Result

Yolo: Real time Object detection



YOLO

➢ YOLO processes an image in 3 major steps:

Resizes the image into 448 x 448

Runs a convolution network on the image

Threshold the resulting detection by model’s confidence

Dataset - Pascal Voc 2012

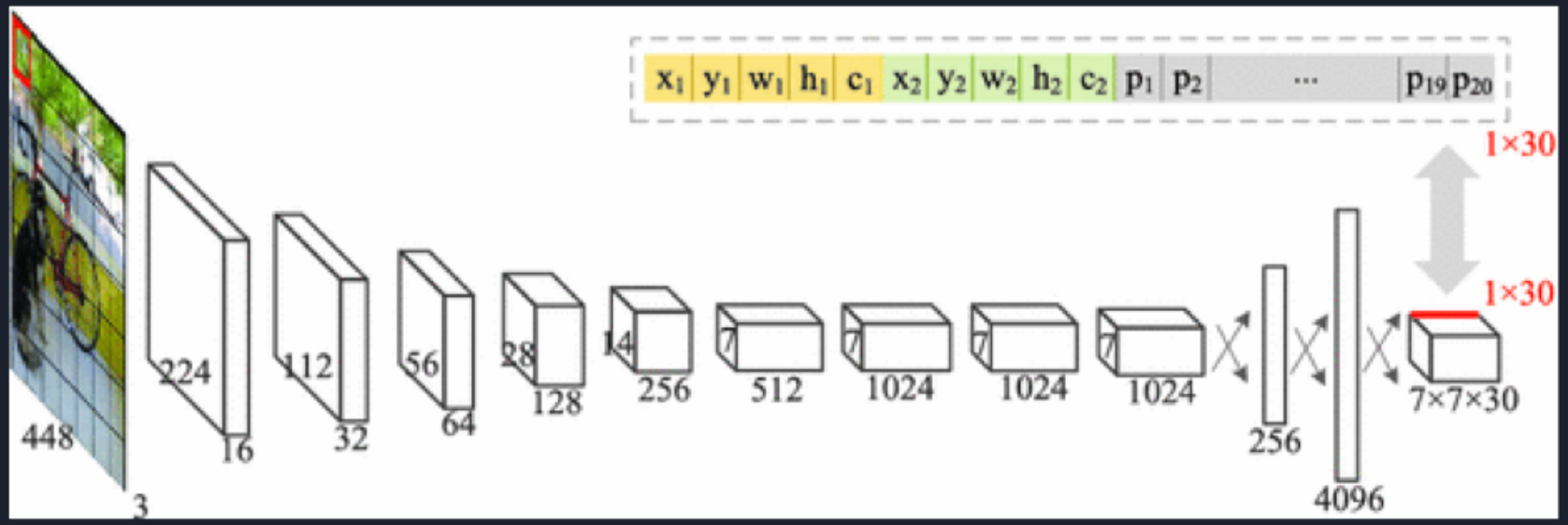
Dataset specification

- Object categories - 32
- Images - 110k (>100k labeled)
- Object instances - 448k

Training CenterNet on MS COCO

- Training images - 25k
- Validation images - 6k
- Test images - 12k

Yolo detection architecture



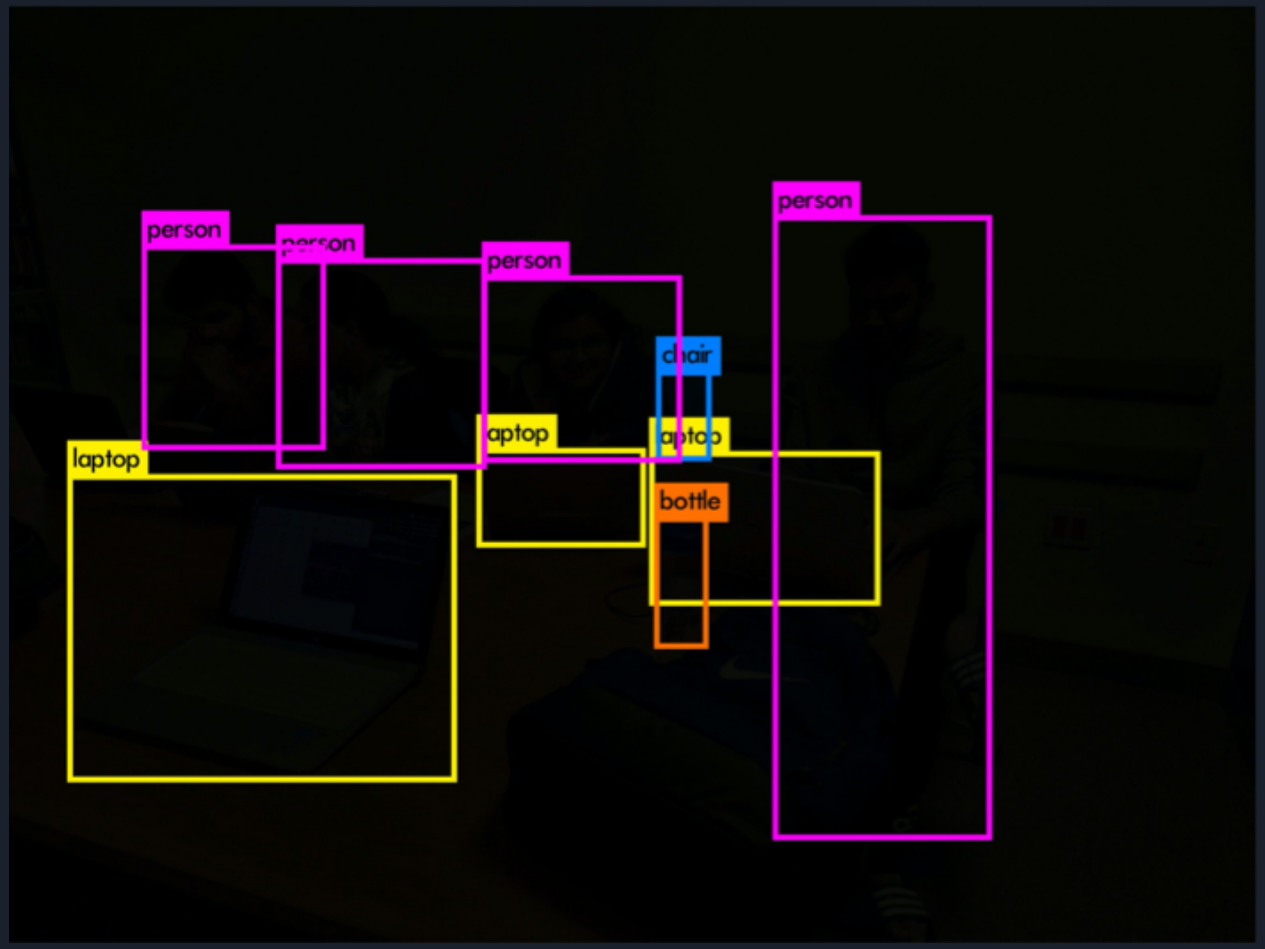
YOLO Implementation Details

- Tools used:
  - The YOLO model was executed on a local Linux system with the following specifications:

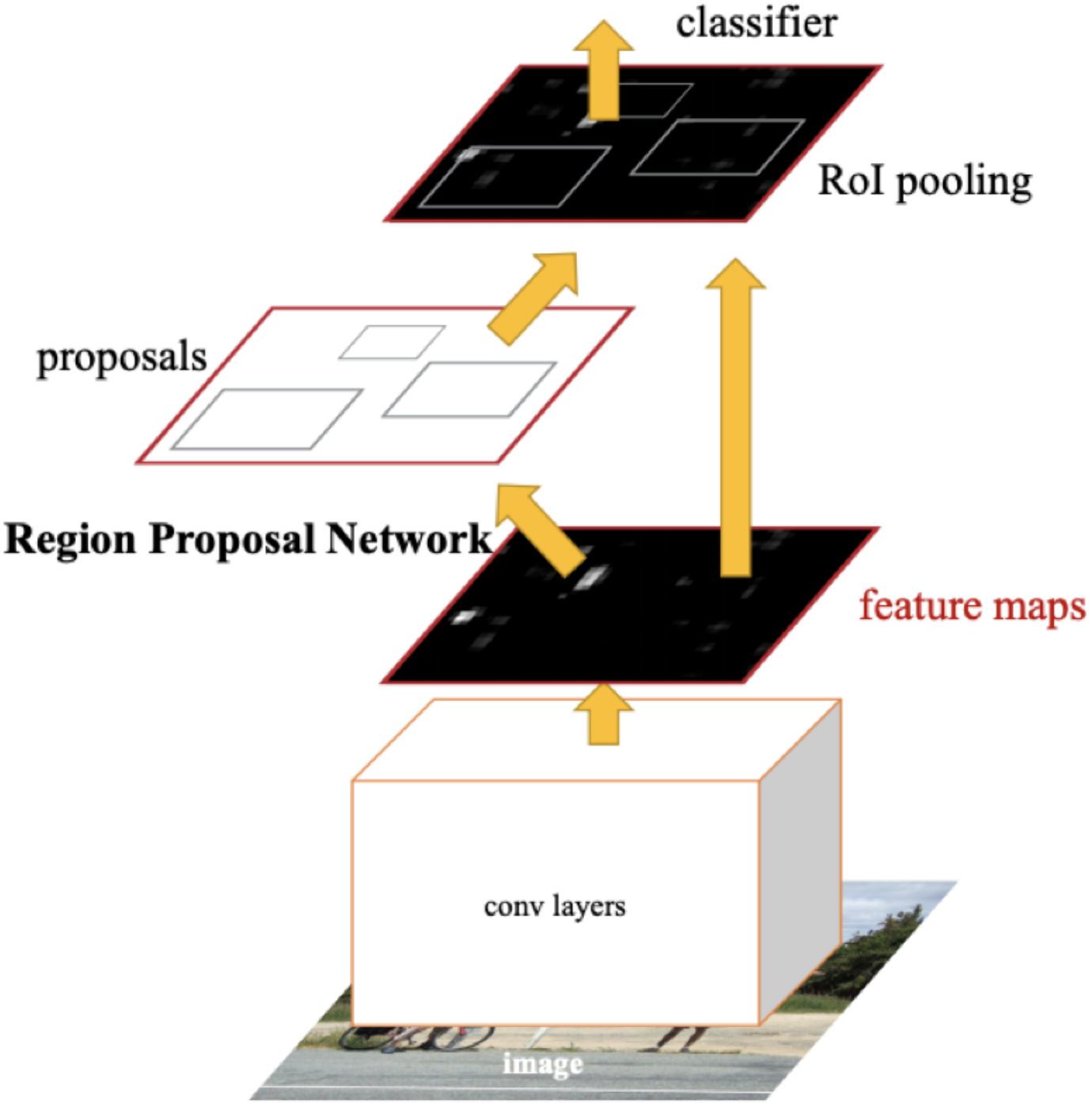
    - Hardware:
      - CPU: Intel Core i7 6th gen,
      - RAM: 16GB,
      - GPU: 8GB NVIDIA GeForce gtx – 960x

    - Software:
      - Operating system: Linux Ubuntu
      - Python 3

Low-light Image vs. Enlightened Image Results



Faster R-CNN


classifier
RoI pooling
proposals
Region Proposal Network
feature maps
conv layers
image

Dataset - PASCAL VOC 2007

Dataset specification

- Object categories - 20
- Images - 9,963
- Object instances - 24,640

Training Faster R-CNN on PASCAL VOC 2007

- Training images - 5k
- Test images - 5k

Implementation Details

➢ AWS EC2 Instance with Deep Learning Base AMI (Ubuntu 16.04) and version 20.0 with p2.xlarge GPU instance was setup in order to run this model.

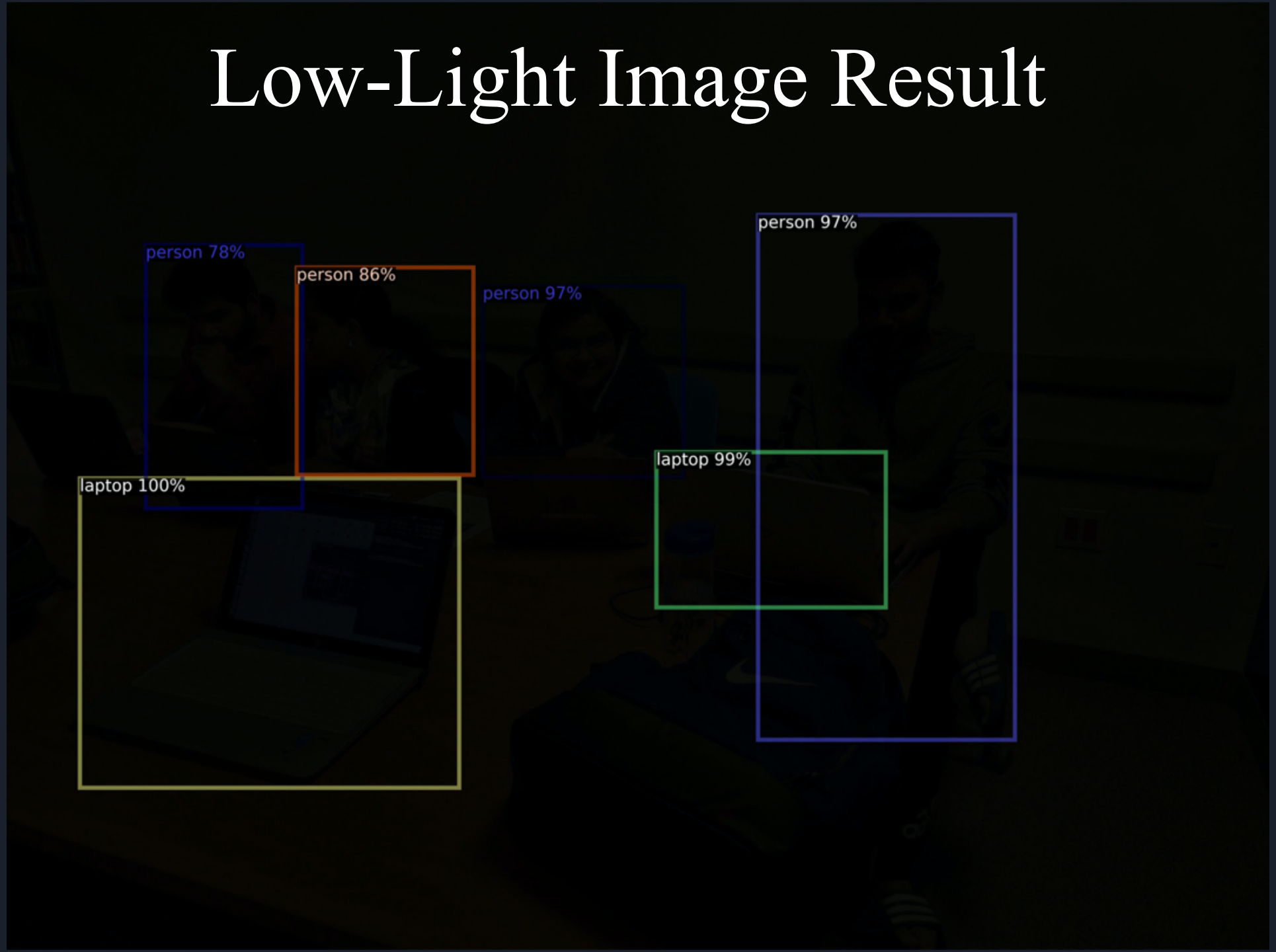
➢ The implementation code that we incorporated used Detectron2.

➢ p2.xlarge specifications
- GPU -1
- vCPUs - 4
- RAM - 61GB

➢ Software specifications
- Python 3.7
- CUDA 10.0
- cnDNN 7.6.4
- PyTorch 1.3.0
- TensorFlow 1.5.0rc2
- Keras 2.2.5
- MxNet 1.6.0

➢Code Reference: https://github.com/facebookresearch/detectron2

Low-Light Image Result



Enlightened Image Result

CenterNet

➢ Uses keypoint estimation to find center points and regresses all object properties such as size, 3D location, orientation, and even pose.

➢ End-to-end differentiable, simpler, faster, and more accurate than corresponding bounding box based detectors

➢ Model is trained using standard dense supervised learning

➢ Inference is a single network forward pass, without NMS for post-processing

Dataset - MS COCO

Dataset specifications

- Object categories - 80
- Images - 330k (>200k labeled)
- Object instances - 1.5 million
- People with key points - 250,000

Training CenterNet on MS COCO

- Training images - 118k
- Validation images - 5k
- Test images - 20k

Implementation Details

➢ AWS EC2 Instance with Deep Learning Base AMI (Ubuntu 16.04) and version 24.3 with p2.xlarge GPU instance was setup in order to run this model

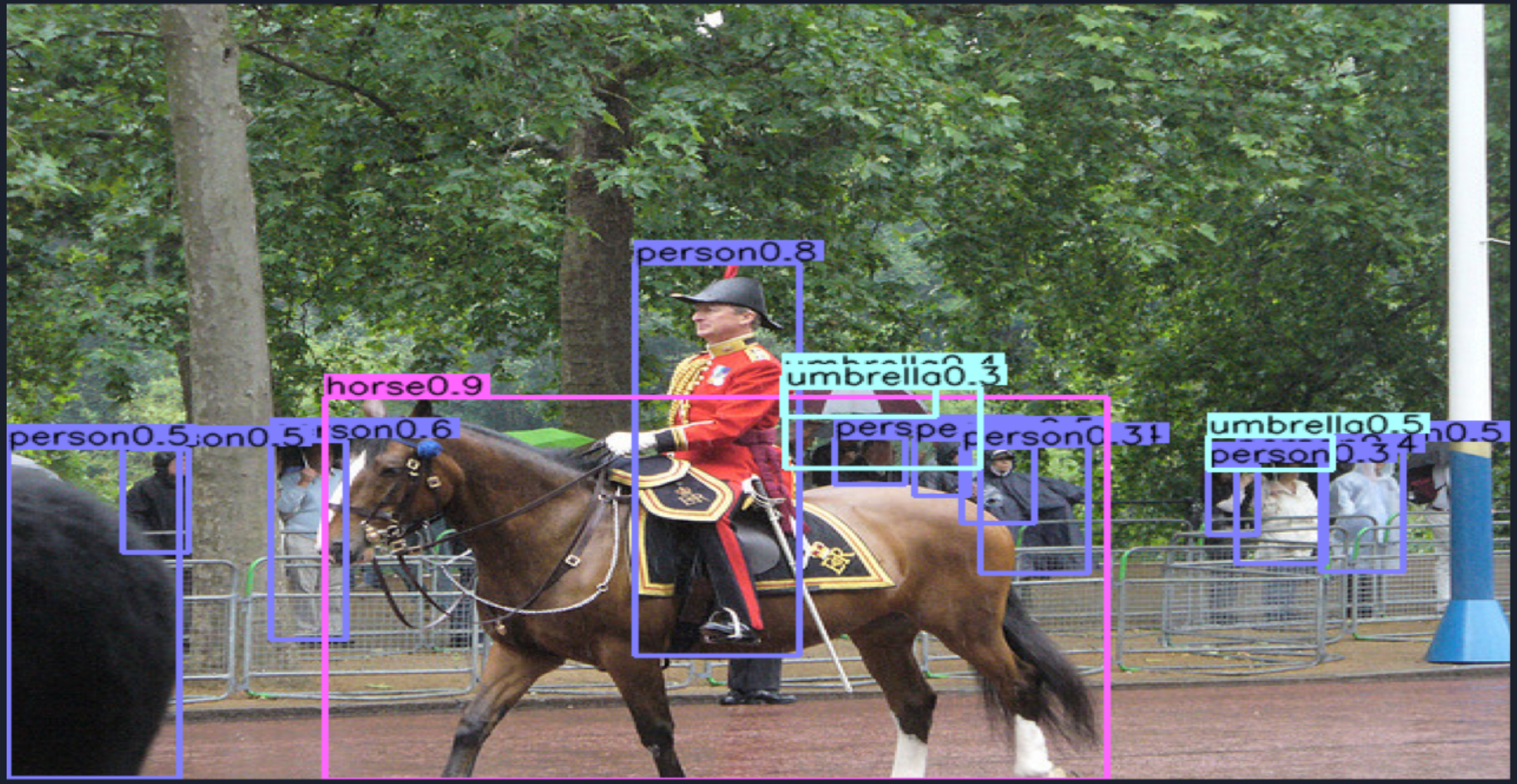
- p2.xlarge specifications
  - GPU -1
  - vCPUs - 4
  - RAM - 61GB
- Software specifications
  - Python - 3.6
  - CUDA - 9
  - cudNN -7.1
  - PyTorch - 0.4.1

CenterNet with 3 different backbone networks

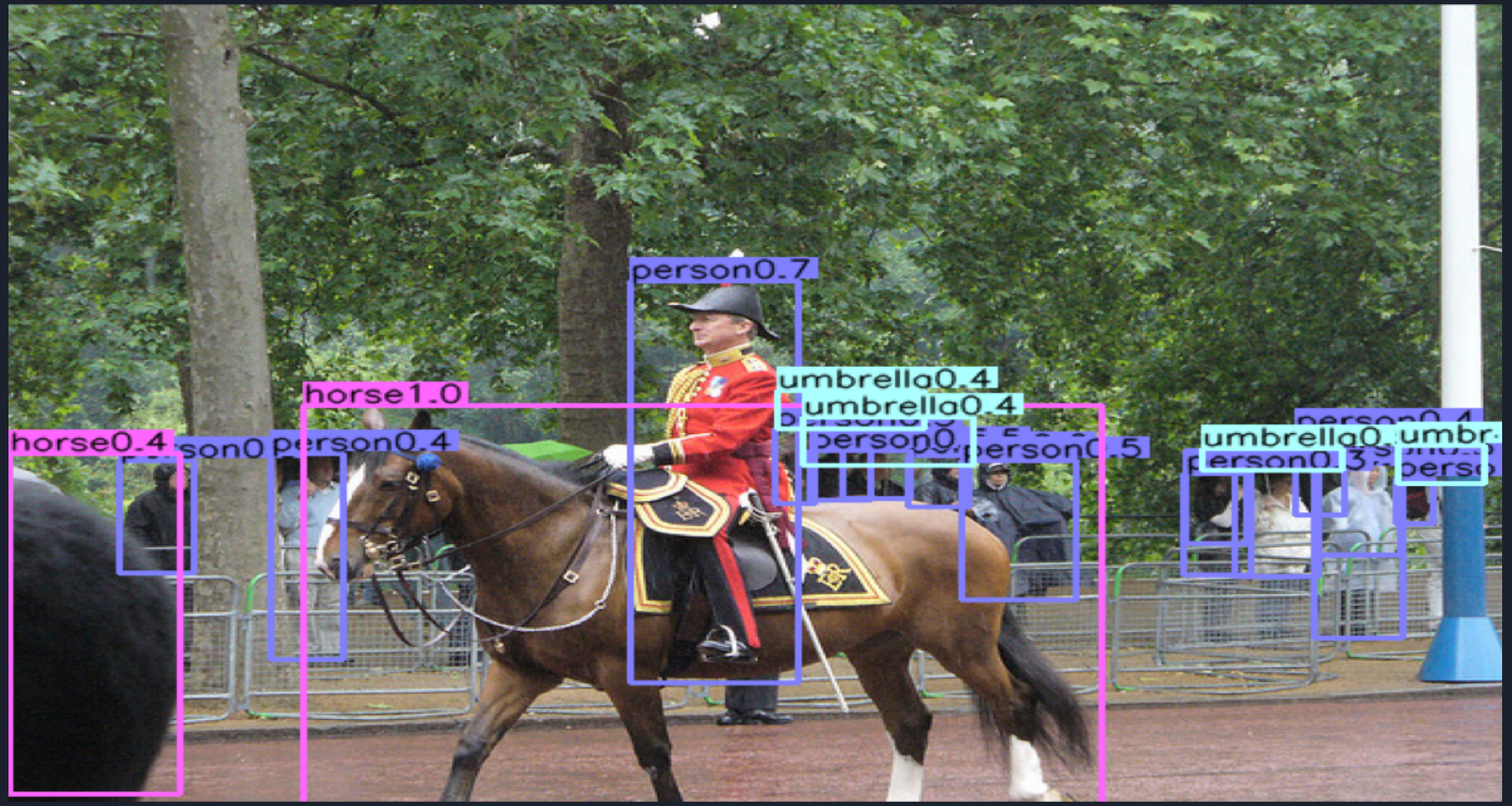
Experiments conducted on sample Image shown in next slides

Backbone network	Inference Time (seconds)	Objected detected(Sample image)
Hourglass - 104	0.628	19
DLA - 34	0.229	16
ResNet - 101	0.208	14

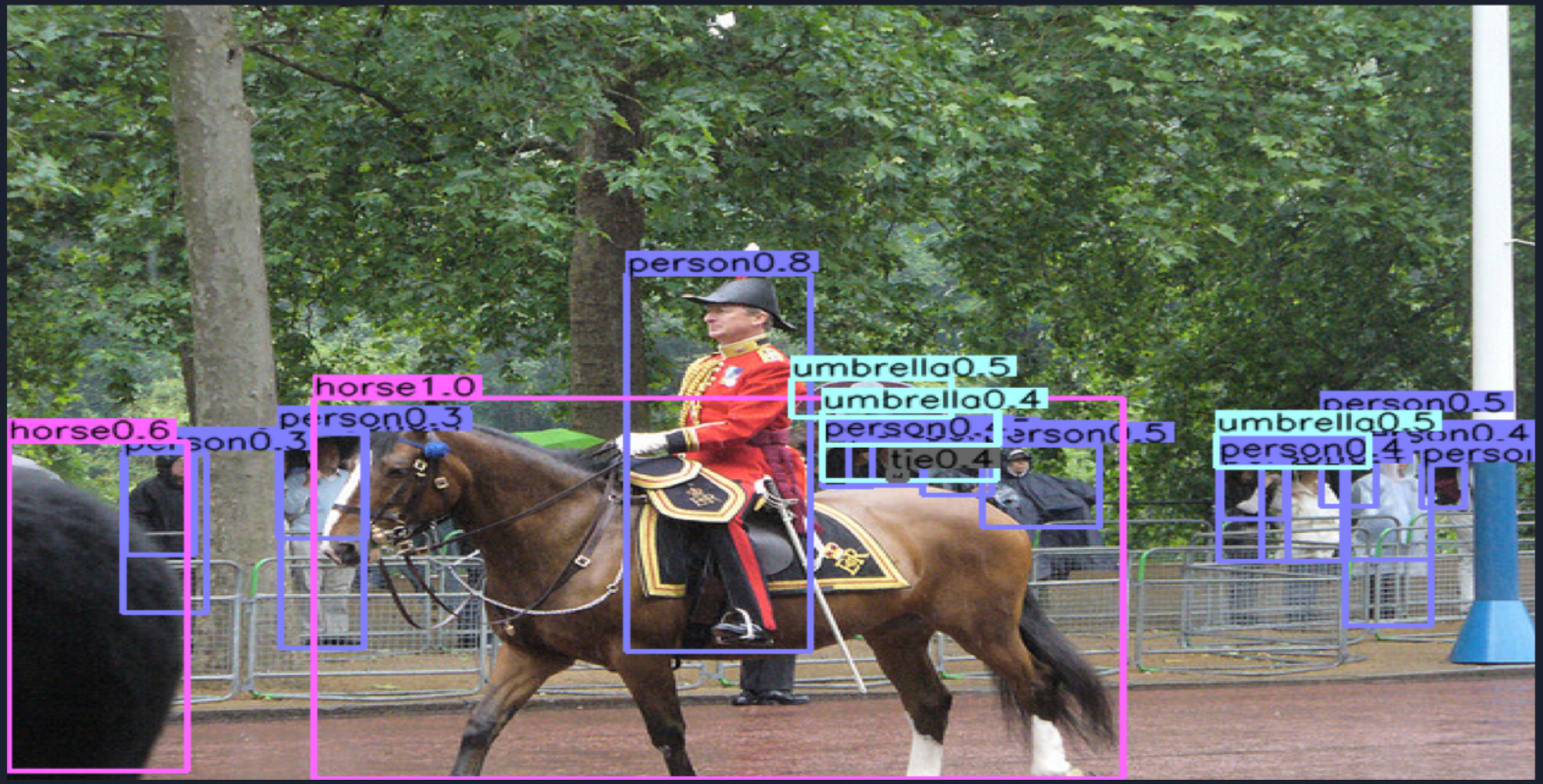
CenterNet with Resnet - 101



CenterNet with DLA - 34

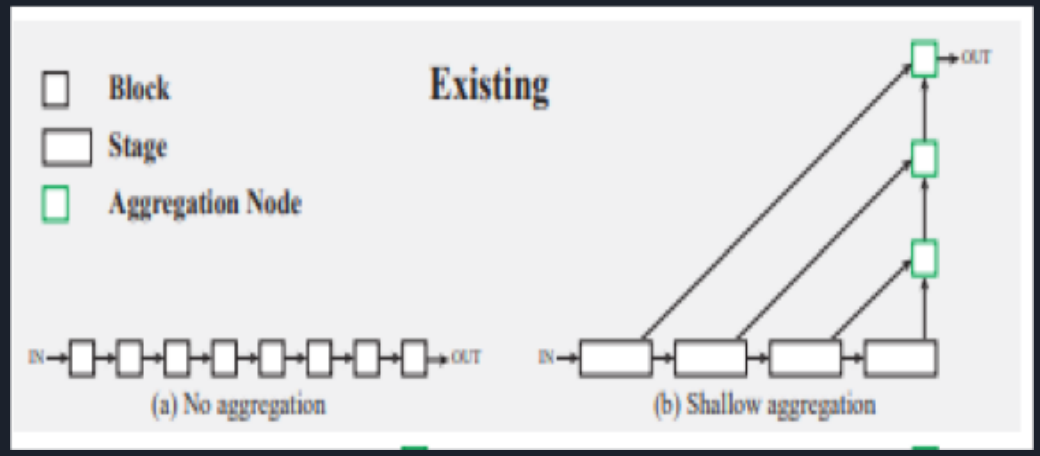


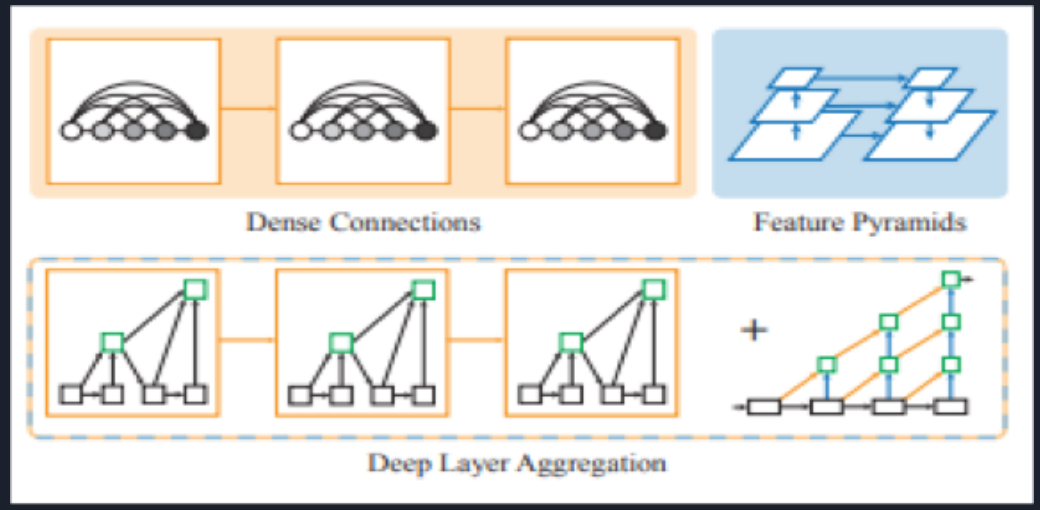
CenterNet with Hourglass - 104



CenterNet with DLA (Deep Layer Aggregation)

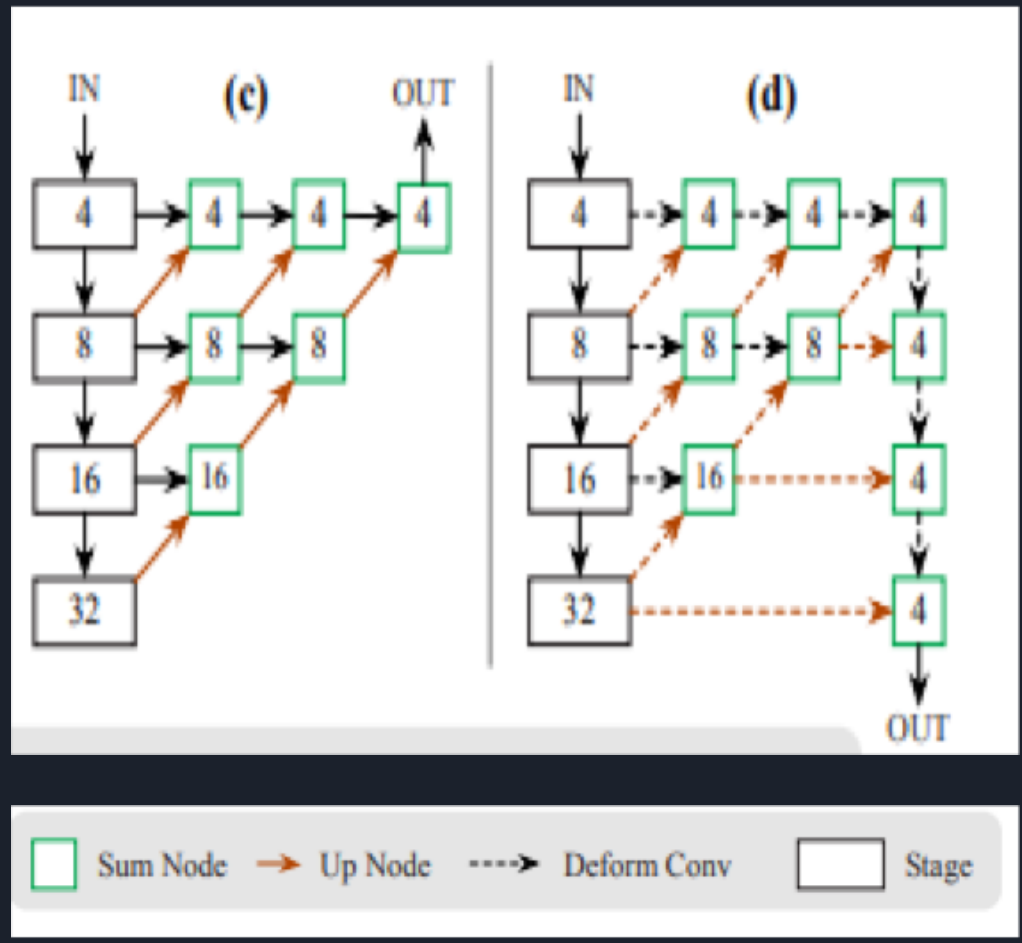
- Aggregates different layers by grouping them into blocks and further grouping them into stages.
- Combines densely connected networks and feature pyramid networks with hierarchical and iterative skip connections.
- Improves resolution and recognition by unifying semantic and spatial information.





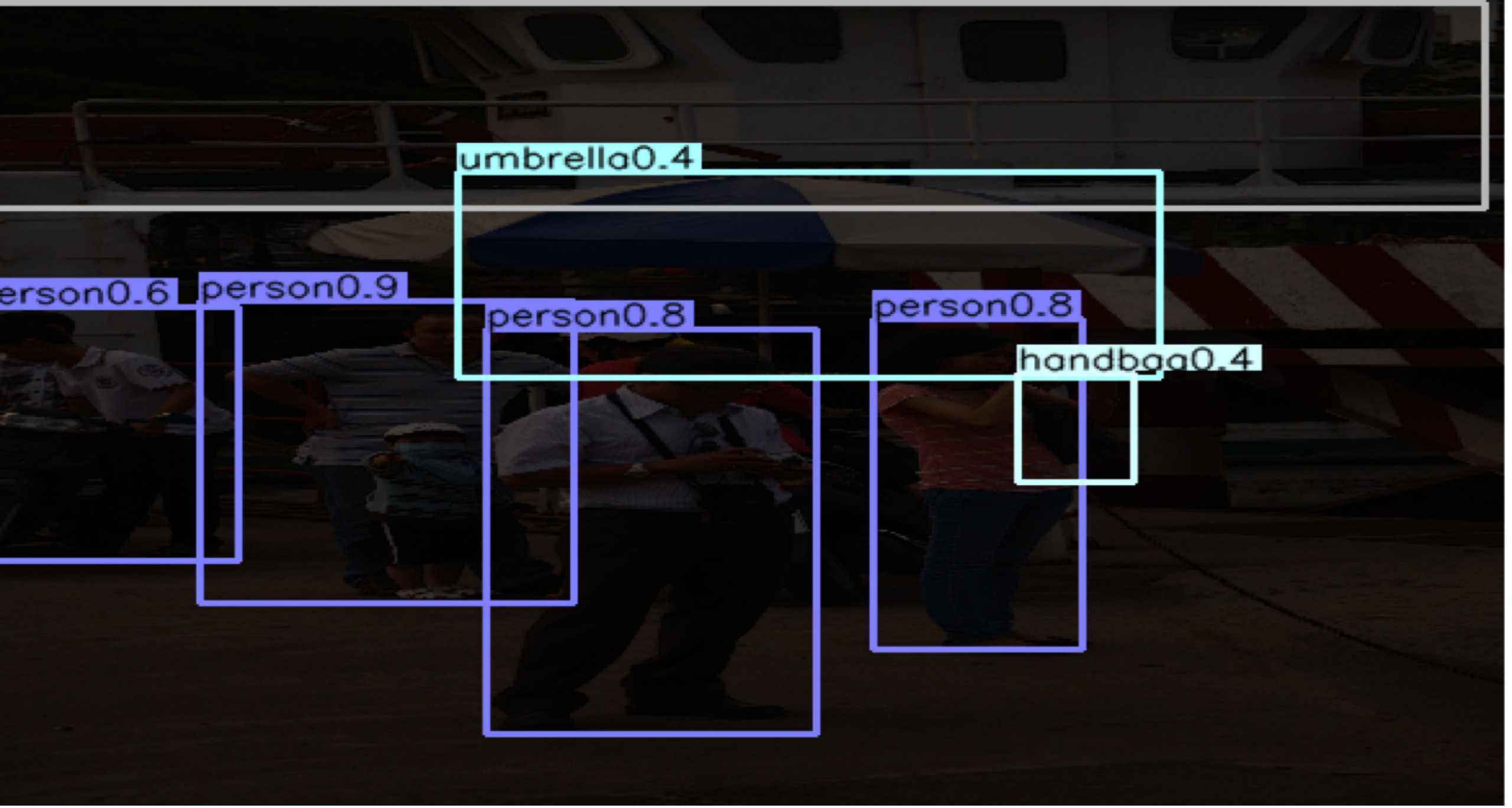
CenterNet with DLA (Deep Layer Aggregation)

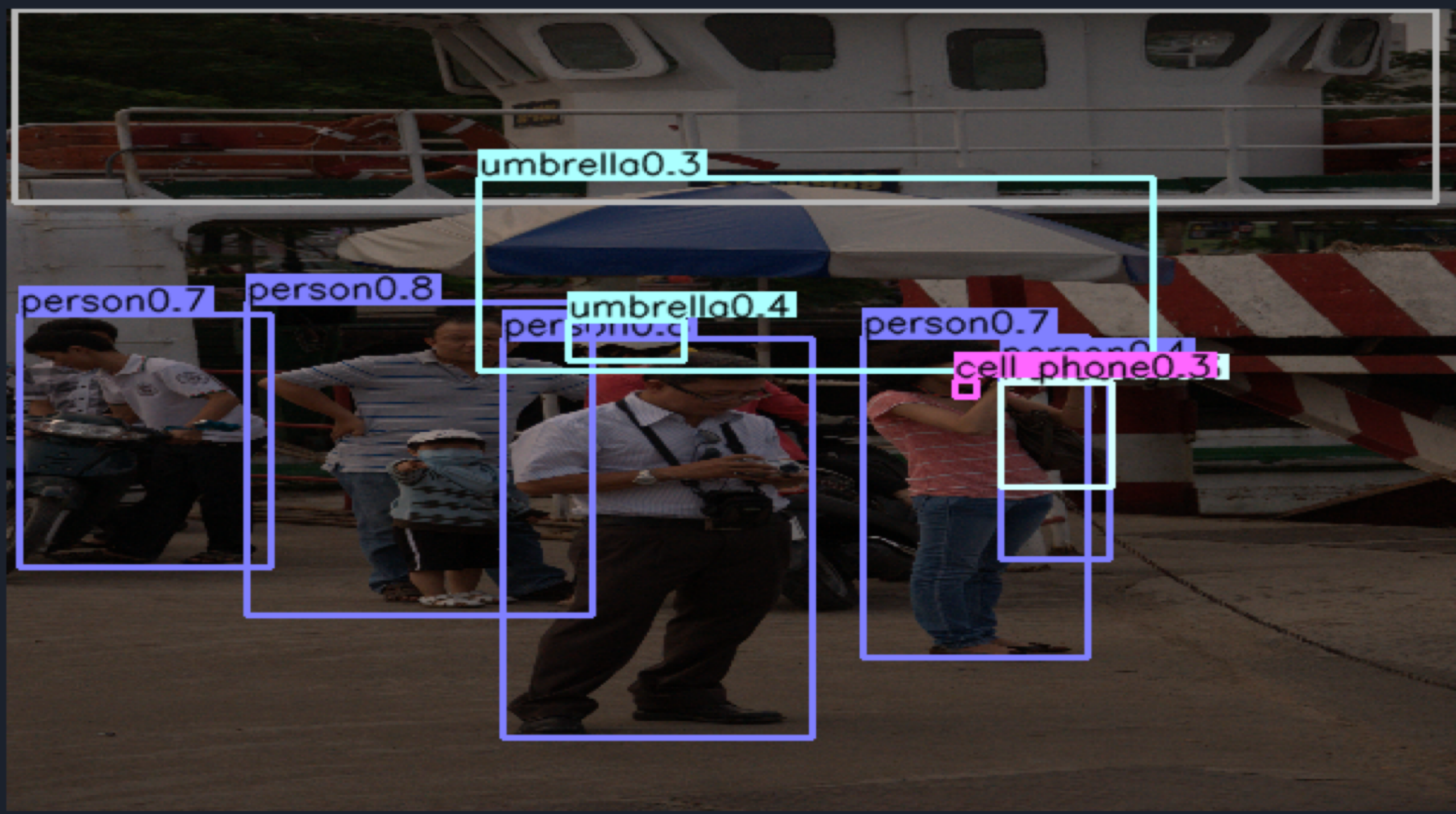
- Utilizes the fully convolutional upsampling version of the DLA for dense prediction
- Augments the skip connections with deformable convolution from lower layers to the output.
- Replaced the original convolution with 3x3 deformable convolution at every upsampling layer.
- Final 1x1 convolution produces the desired output



Lowlight Image vs EnlightenGan Image

- Ran CenterNet with DLA on lowlight images.
- Ran CenterNet with DLA on the brightened images generated by EnlightenGAN.
- Compared the results of the two to verify the improvement in performance.


umbrella0.4
erson0.6
person0.9
person0.8
person0.8
handbag0.4


umbrella0.3
person0.7
person0.8
umbrella0.4
person0.7
cell phone0.3

Model Comparison

Models	Accuracy (low-light)	Accuracy (enlighten)	Time required(seconds)
YOLO	75%	83.33%	30.3
SqueezeDet	20%	40%	4
Faster R-CNN	54%	81%	1
CenterNet	75%	91%	0.30

References

- Yona Falinie, A. Gaus, Neelanjan Bhowmik, Samet Akcay, Paolo M. Guillen-Garcia, Jck W. Barker, Toby P. Breckon, “Evaluation of a Dual Convolutional Neural Network Architecture for Object-wise Anomaly Detection in Cluttered X-Ray Security Imagery,” April 10, 2019.
- Pengchong Jin, Vivek Rathod, Xiangxin Zhu, Google, “Pooling Pyramid Network for Object Detection”, July 9, 2018.
- “*You Only Look Once: Unified, Real-Time Object Detection*”, Joseph Redmon, Santosh Divvalay, Ross Girshick, Ali Farhadi, University of Washington, Allen Institute for AI, Facebook AI Research
- “*Body part detectors trained using 3d human pose annotations*”, L. Bourdev and J. Malik. Poselets, In International Conference on Computer Vision (ICCV), 2009
- D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, “*Scalable object detection using deep neural networks”*. In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, pages 2155–2162. IEEE, 2014
- Bichen Wu, Alivin Wan, Forrest Iandola, Peter H. Jin, Kurt Keutzer, UC Berkeley, DeepScale, “SqueezeDet: Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time Object Detection for Autonomous Driving,” Jun 11, 2019.
- Xingyi Zhou, Dequan Wang, Philipp Krahenbuhl, UT Austin, UC Berkeley, “Objects as Points,” Apr 25, 2019.
- Yifan Jiang, Xinyu Gong, Ding Liu, Yu Cheng, Chen Fang, Xiaohui Shen, Jianchao Yang, Pan Zhou, and Zhangyang Wang, “EnlightenGAN: Deep Light Enhancement without Paired Supervision”, June 17, 2019.

Questions ?