Detecting Cars in a Parking Lot using Deep Learning

Samuel Ordonia
May 2019

Overview

1. Introduction
2. Background
3. Implementation
4. Experiments & Results
5. Skynet Demo
6. Conclusions
7. Q & A

Introduction

- **Objective**: Detecting cars in a parking lot
- **Solution**: Convolutional Neural Net (CNN) single-shot detector

Technical Challenges

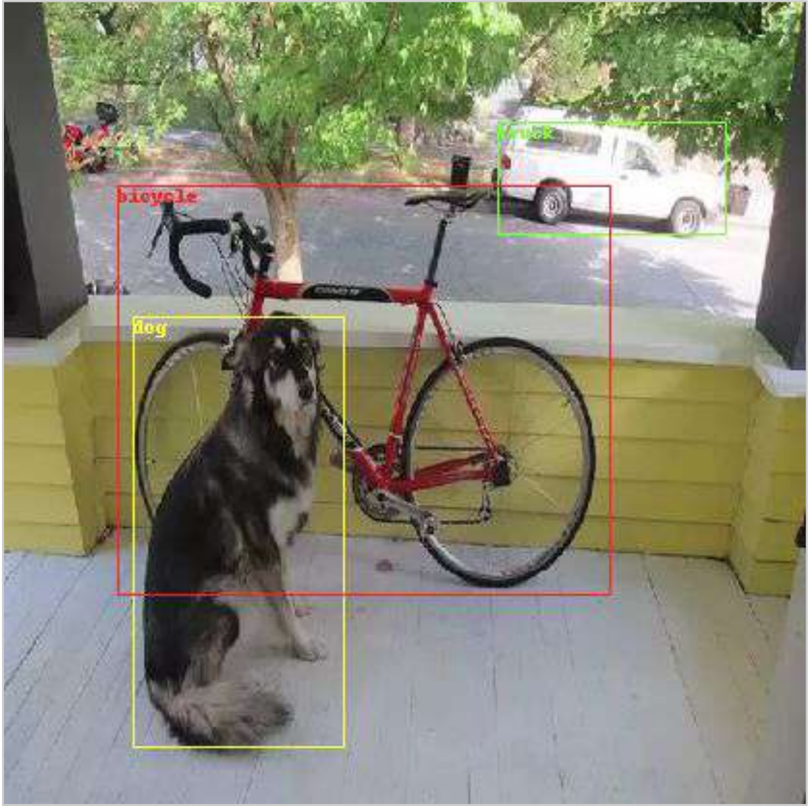
1. Detection of dozens of objects in an image with variable size, color, pose, depth, and occlusion
2. Obtaining thoroughly annotated and accurate training data
3. Effective data compilation and validation tool
4. Implementation and maintenance of a complex detection model
5. Practical application of a parking lot car detector

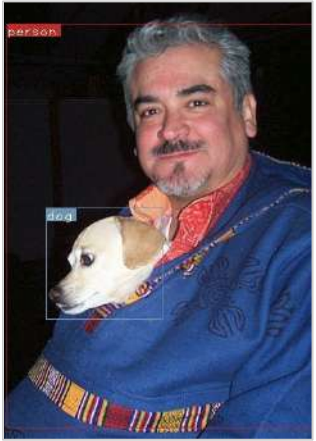
Car Detection

- Two part problem
  - Classification - there is a car in these pixels
  - Localization - these pixels are significant
- Detection - these pixels within this image = car

Car Detection: Iconic Example

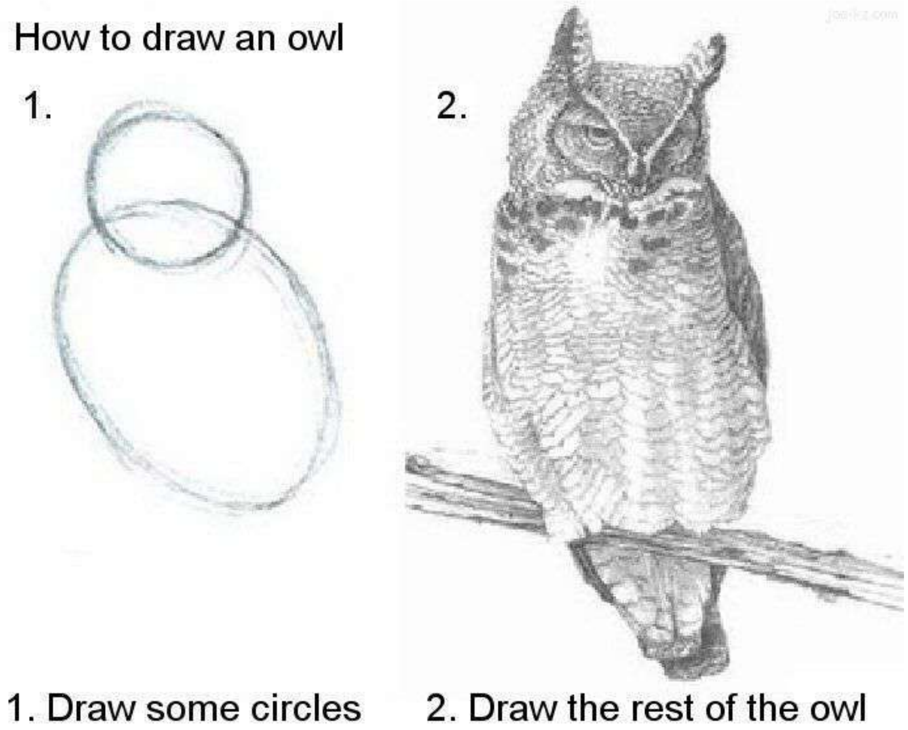
Other Iconic Detection Examples





Background

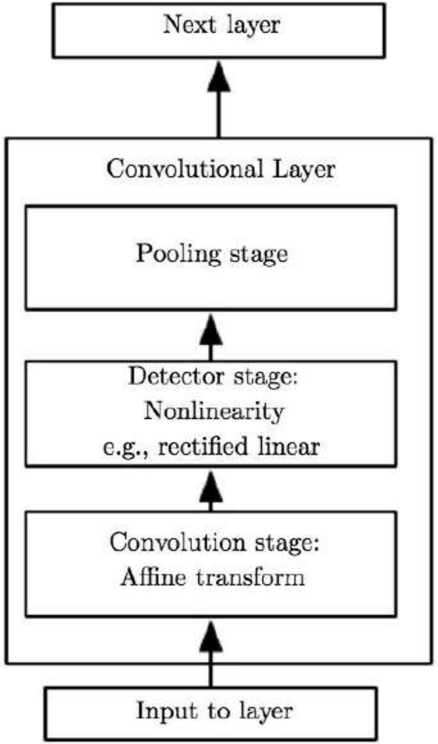
CNN Engineering



Building Blocks of the CNN

- Convolution → apply a weighted kernel (i.e., filter) across an input tensor to derive feature maps.
- Convolutional Layer
  - Convolution transformation
  - Non-linear activation function
  - Pooling Layer
- Kernel
- Stride
- Padding

Convolution Layer Breakdown

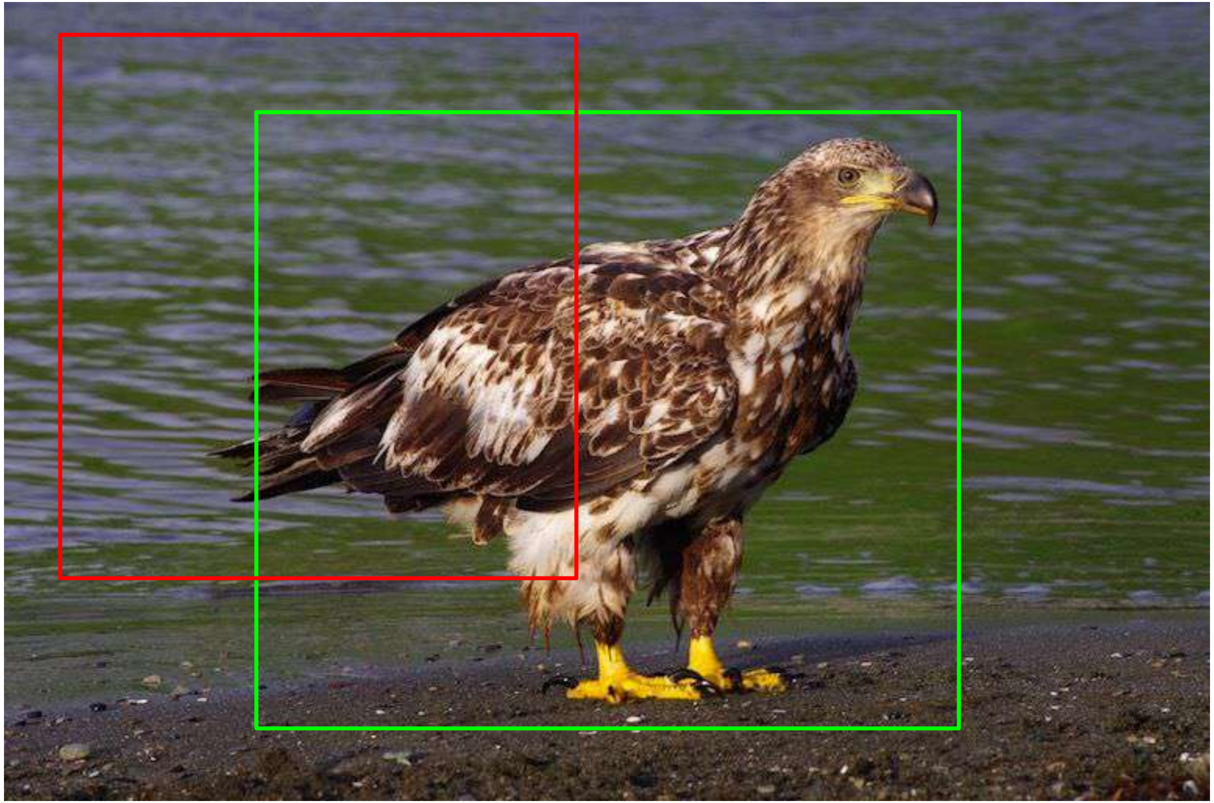


Intersection over Union (IOU)

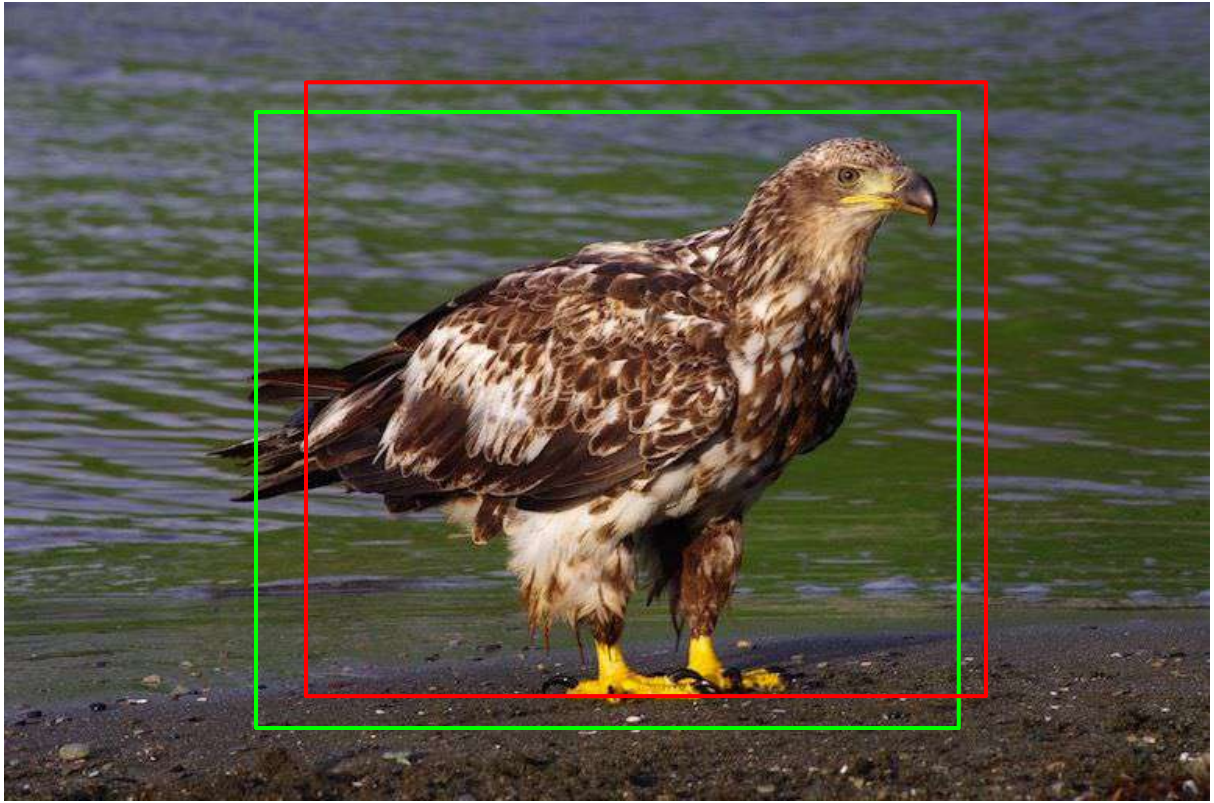
- Objectness score.
- Comparison of the predicted box to the ground truth one.

IOU = Area of Intersection / Total Area

Poor IOU



Excellent IOU



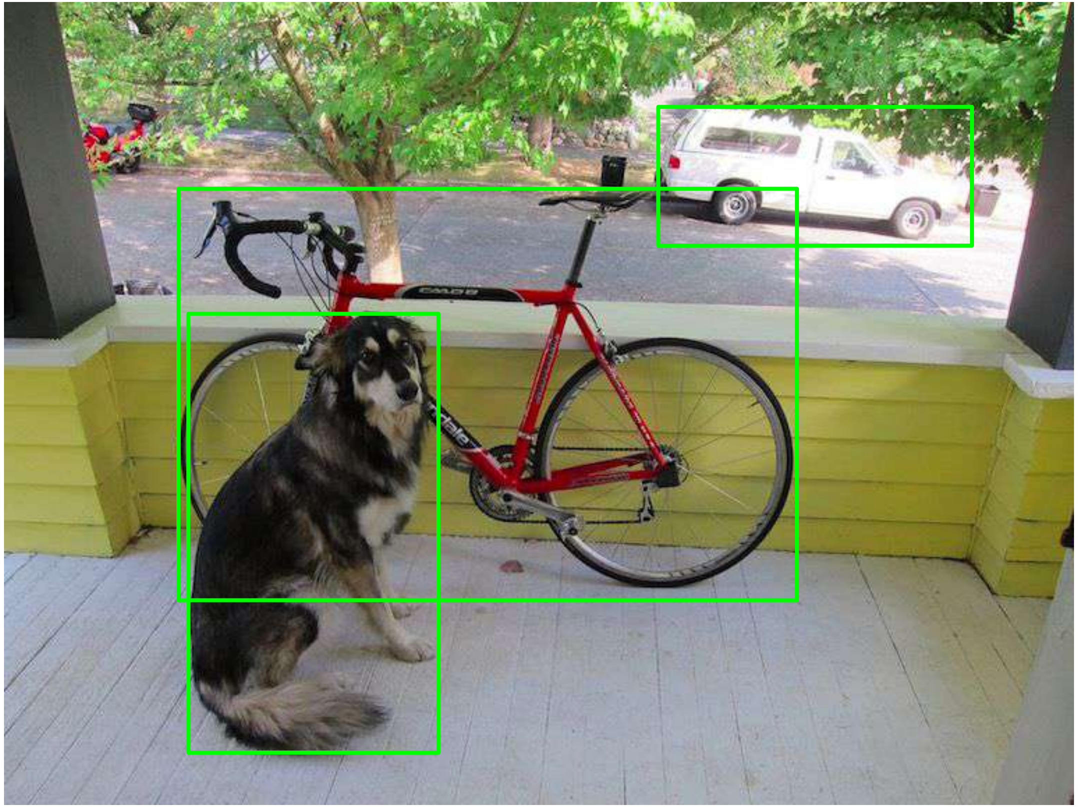
Anchor Boxes

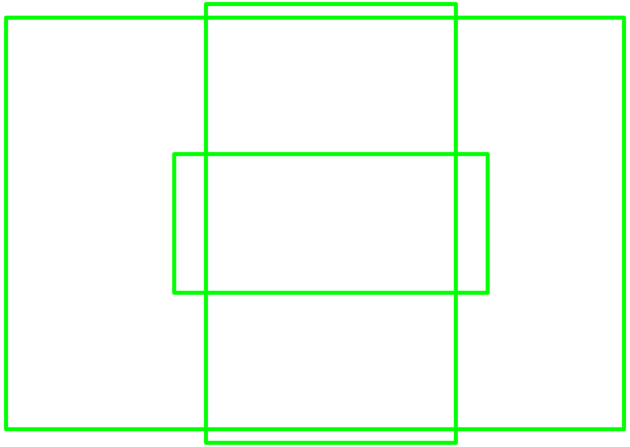
- Simplify bounding box regression.
- Compute IOU based on predetermined anchor boxes instead of the ground truth boxes.
- Predict multiple classes at the same centroid pixel.

Anchor Boxes Derivation

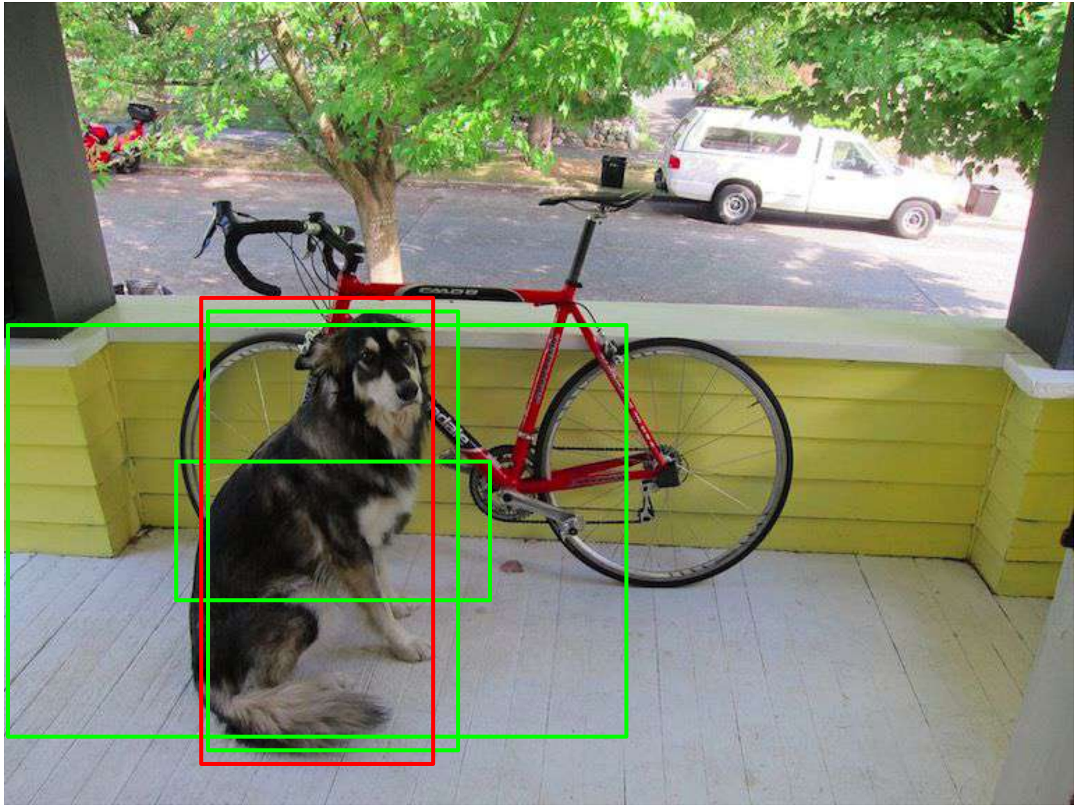
- Derived from k-means clustering of ground truth bounding boxes.
- Determines centroids based on the labeled dataset's height and width parameters.
- No association of (x, y) positions in the image are accounted for.

Anchor Boxes Example





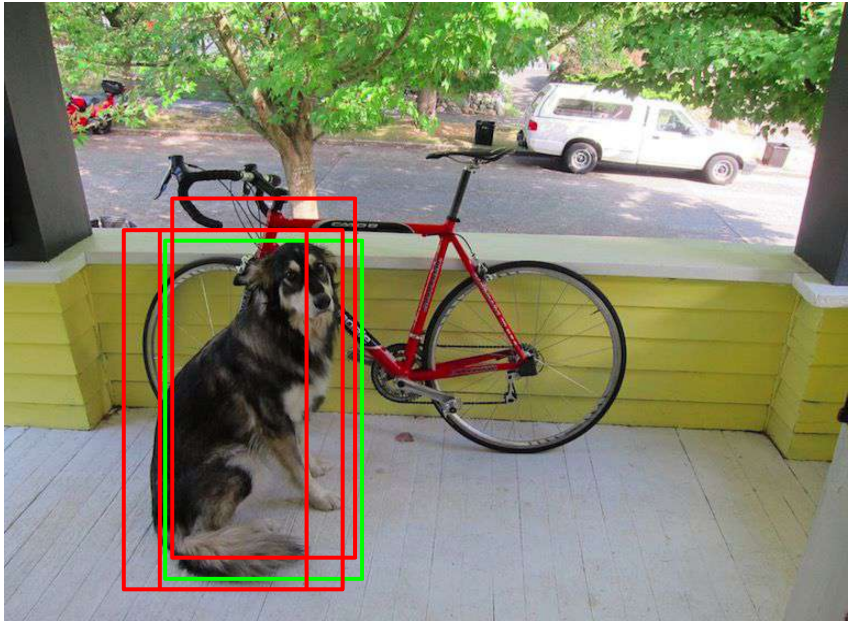
Anchor Boxes Example

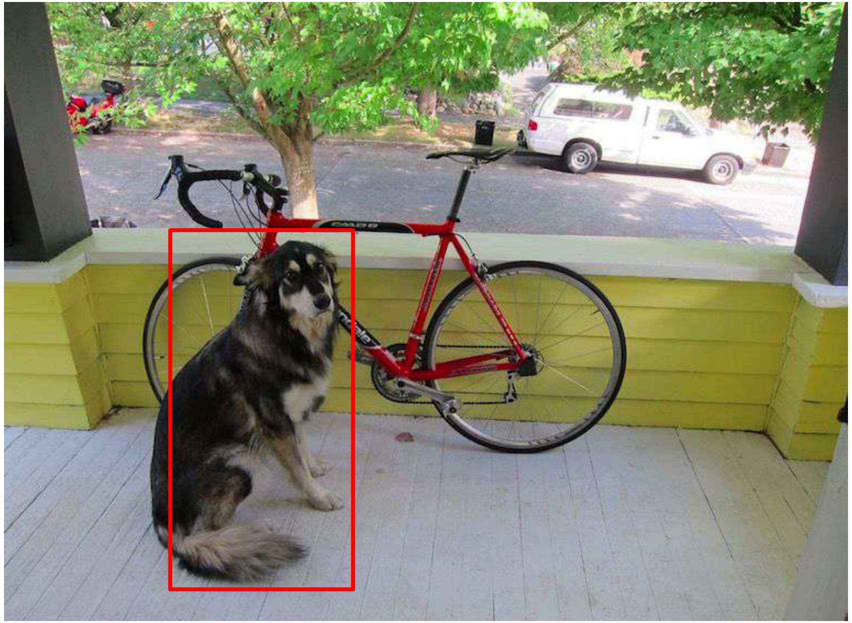


Non-max Suppression

- Bounding box regression approach.
- Maximize intersection of predicted over ground truth bounding boxes.
- Discard excess bounding boxes of high similarity (IOU).

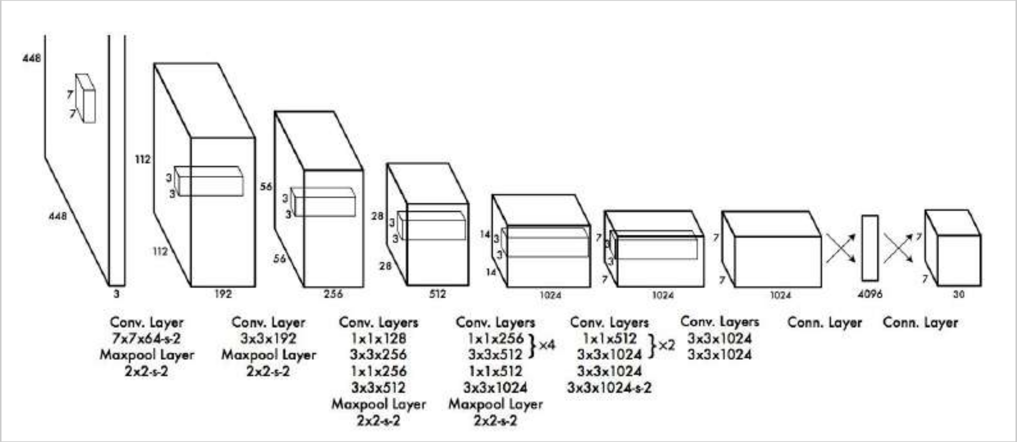
Non-max Suppression





You Only Look Once (YOLO)

- Fully convolutional net → faster and less memory heavy than a fully connected net.
- Performs classification and localization in the same step → faster than multi-stage deep learning models.



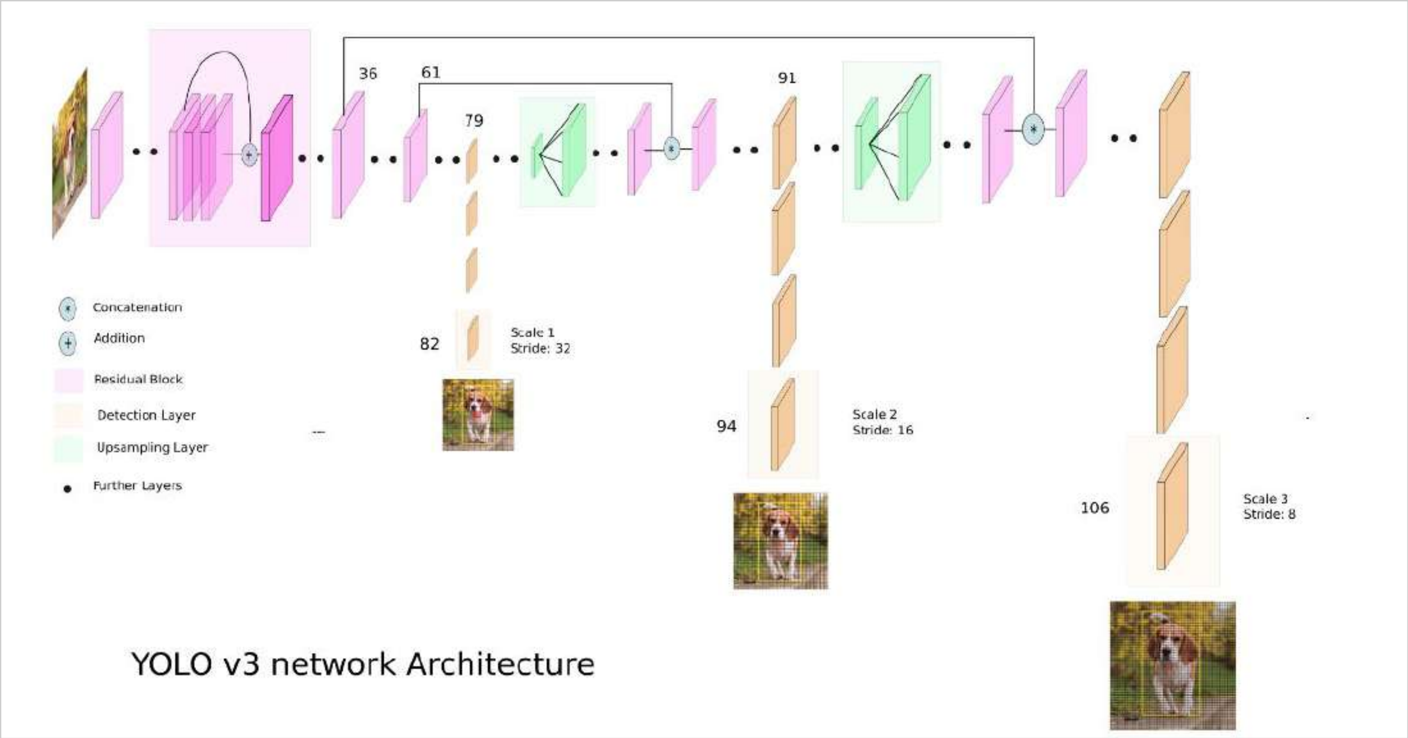
YOLO v2 - Faster, Better

- Switched to anchor boxes and NMS for detection regression.
- Optimal runtime with Darknet-19 feature extraction CNN.
- Single detection layer for cars

YOLO v3 - Three Layers of Detection

- Much deeper feature extraction → higher accuracy, but slower runtime.
- Three detection layers routing different feature maps from the Darknet-53 feature extraction portion.
- Detects cars at different depths.

YOLO v3



36
61
79
91
82
Scale 1
Stride: 32
94
Scale 2
Stride: 16
106
Scale 3
Stride: 8
Concatenation
Addition
Residual Block
Detection Layer
Upsampling Layer
Further Layers
YOLO v3 network Architecture

Implementation

Vision

- Open CV Docker application for parking lot data compilation, processing, and validation.
- Compilation → uniform format for training the CNN models
- Validation → verify accuracy of the ground truth labeling

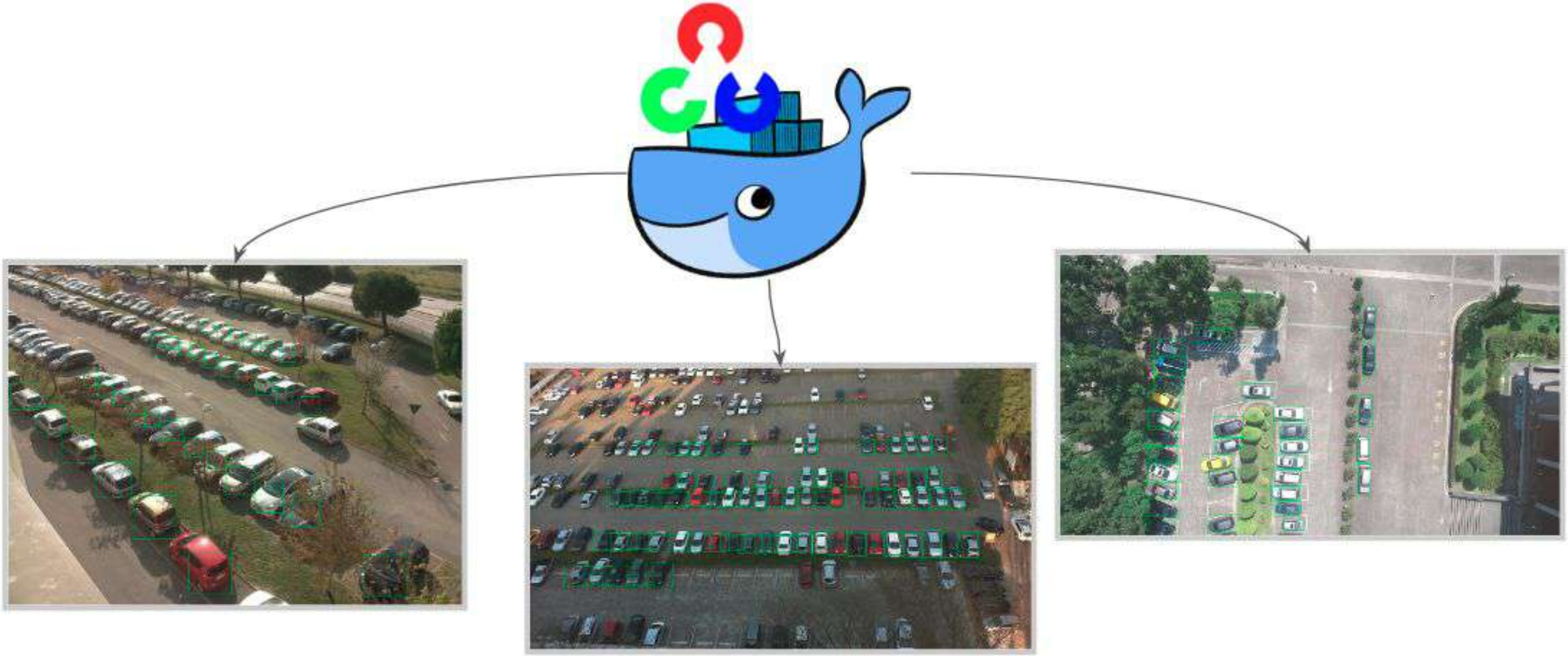
Vision: Open CV

- Computer Vision framework with Python bindings.
- Highly optimized in C.
- Installed binaries from the framework → wreak havoc with the host machine's potentially existing versions of those executables or libraries.

Vision: Docker

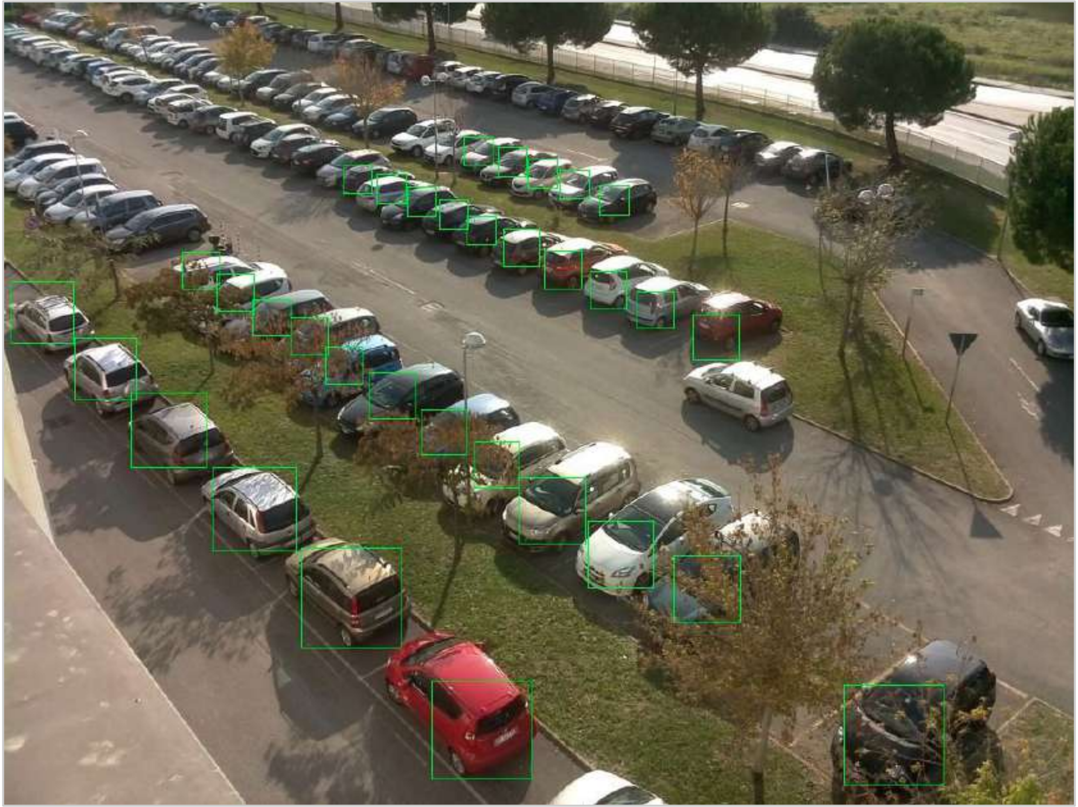
- Linux containerization as a service.
- Safely install and execute Open CV within the Docker application. Host machine's existing programs are untouched.
- Concise syntax and well documented features.

Vision Architecture



Vision: Bounding Box Annotation

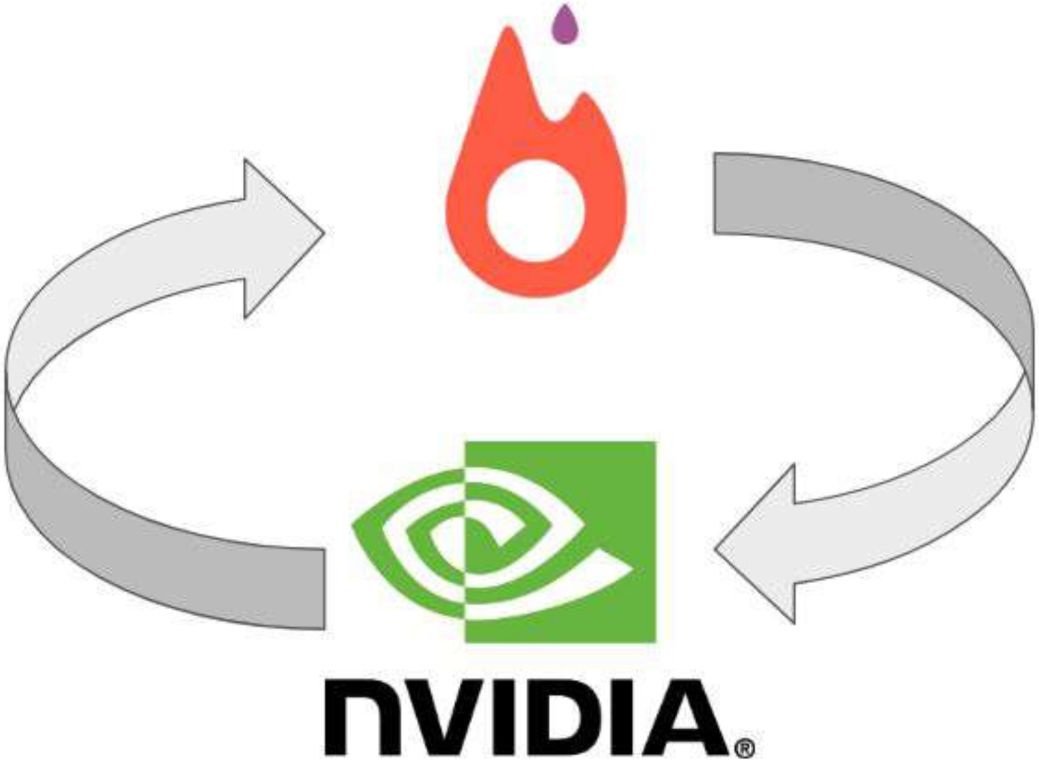
Vision: Bounding Box Annotation



Research Environment

- Pytorch deep learning application powered by NVIDIA's CUDA GPU integration.
- Models defined in configuration (cfg) files.
- Factory-like Pytorch logic generates a model and its layers.

Research Environment Architecture



Research Environment: Pytorch

- Pytorch is a Python deep learning framework for research and production.
- Flexible & concise syntax abstracts away enough lower-level logic involving neural nets.
- Imperative execution → easily debug as deep learning scripts are executed.
- Seamless & straightforward integration with NVIDIA CUDA framework.

Research Environment: Model Configurations

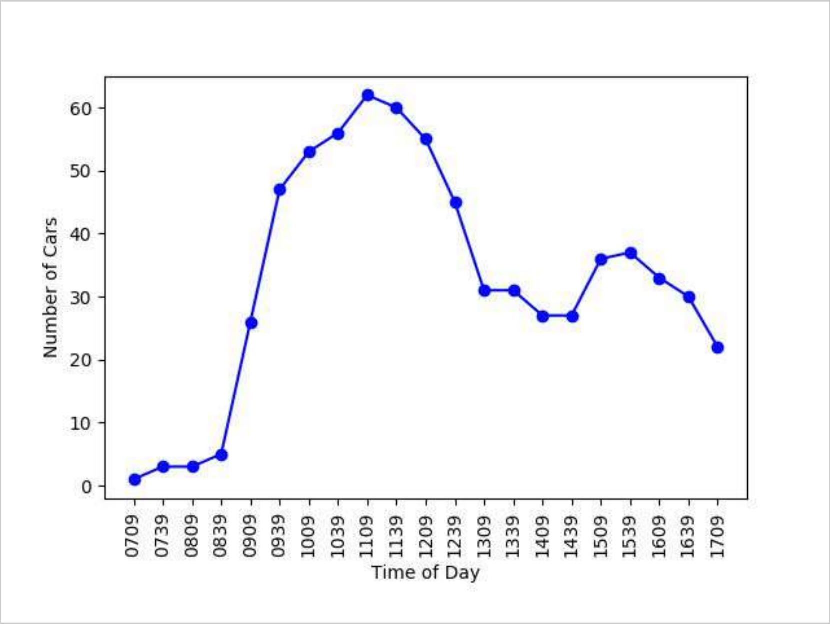
- Standardized file format (cfg) to represent model definitions.
- Blocks contain hyperparameters or layer-specific parameters.
- Used throughout the deep learning community → independent of language or framework

Skynet: Practical Application

- Docker application to execute statistical analysis on detection results.
- Invokes the Research Environment's Pytorch Detector, though it is decoupled from the core training environment.
- Executable on a device with or without a GPU.

Skynet: Car Count Time Series

- Car detection across several images over time.
- Plots car counts in the parking lot over time.

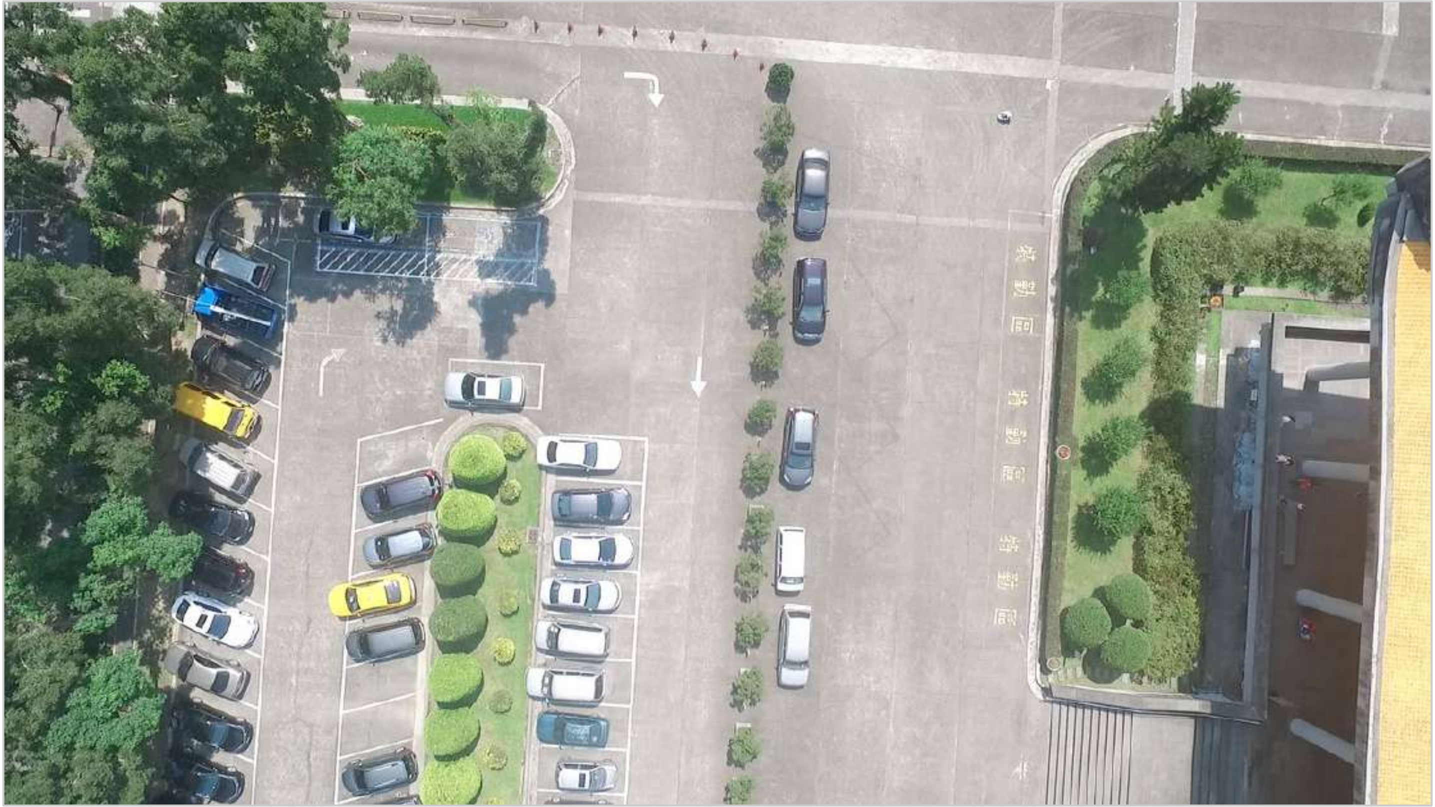


Experiments & Results

Training Data

- CARPK
  - Top-down drone captured parking lot images
  - Cars have different poses but same sizes
  - Minimal occlusion
  - Fully annotated ground truth bounding boxes
- CNR
  - Fixed camera angle covering the parking lot
  - Cars have different poses and depth
  - More occlusions behind trees and other cars
  - Not all cars are labeled with bounding boxes

Input Image Example



Input Image Example

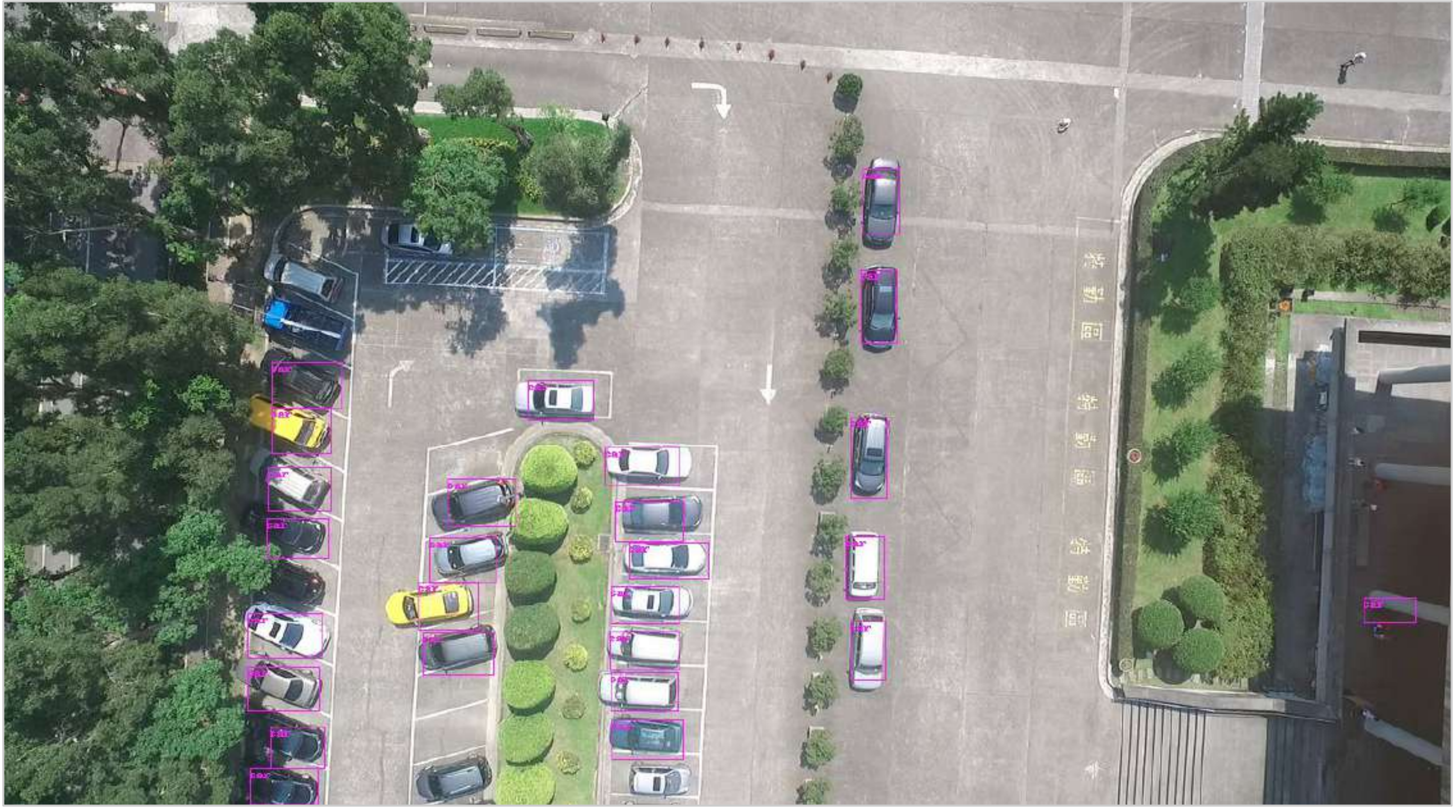
Models

- YOLO v2 (Darknet-19 + one detection layer)
- YOLO v3 (Darknet-53 + three detection layers)
- Modified YOLO v3 (Darknet-53 + four detection layers)

YOLO v2: One Detection Layer

- Single detection layer with anchor boxes.
- Fastest of the CNN models in this project due to having the fewest layers.
- Least accurate because of single detection layer.

CNN Detection Result: YOLO v2

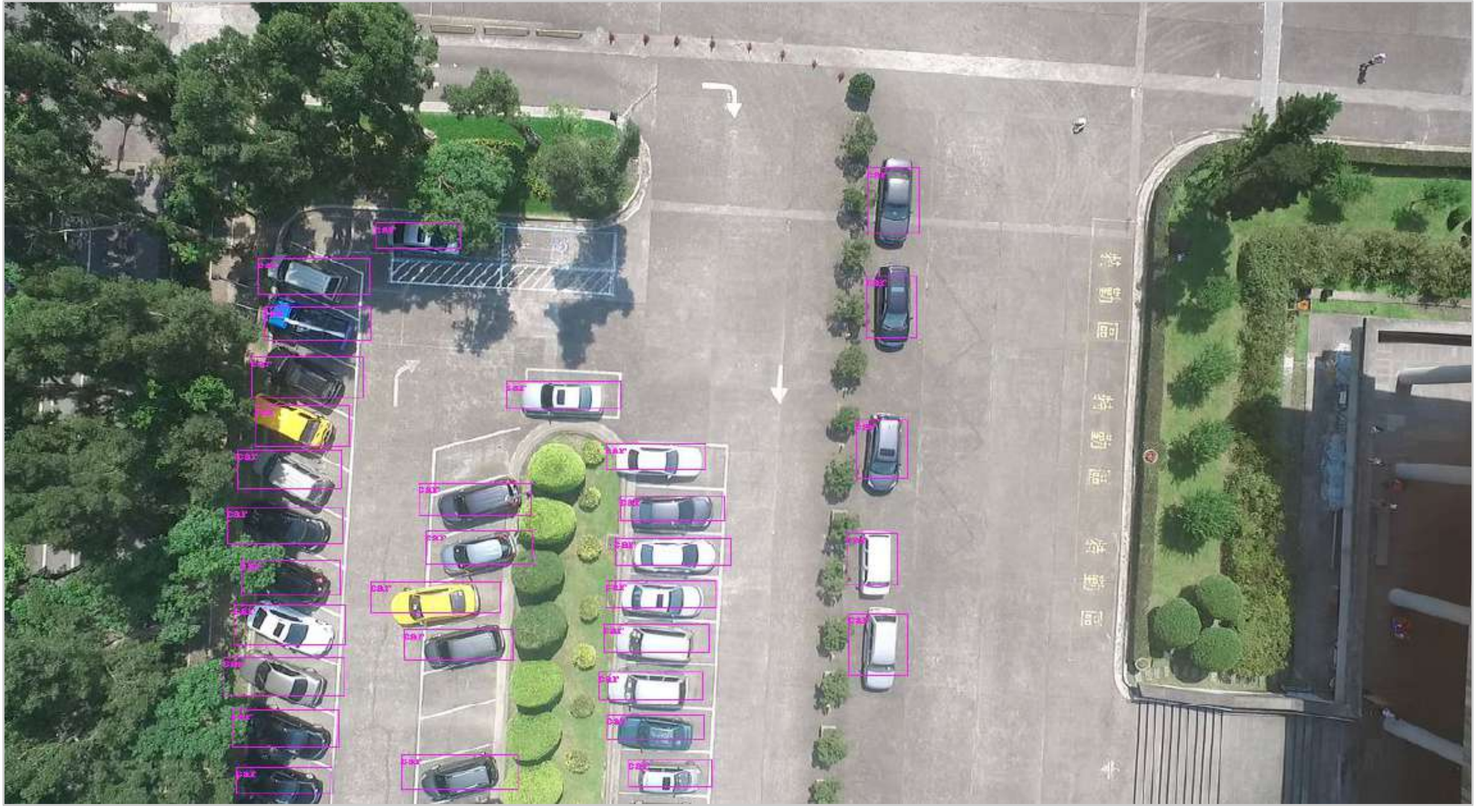


CNN Detection Result: YOLO v2

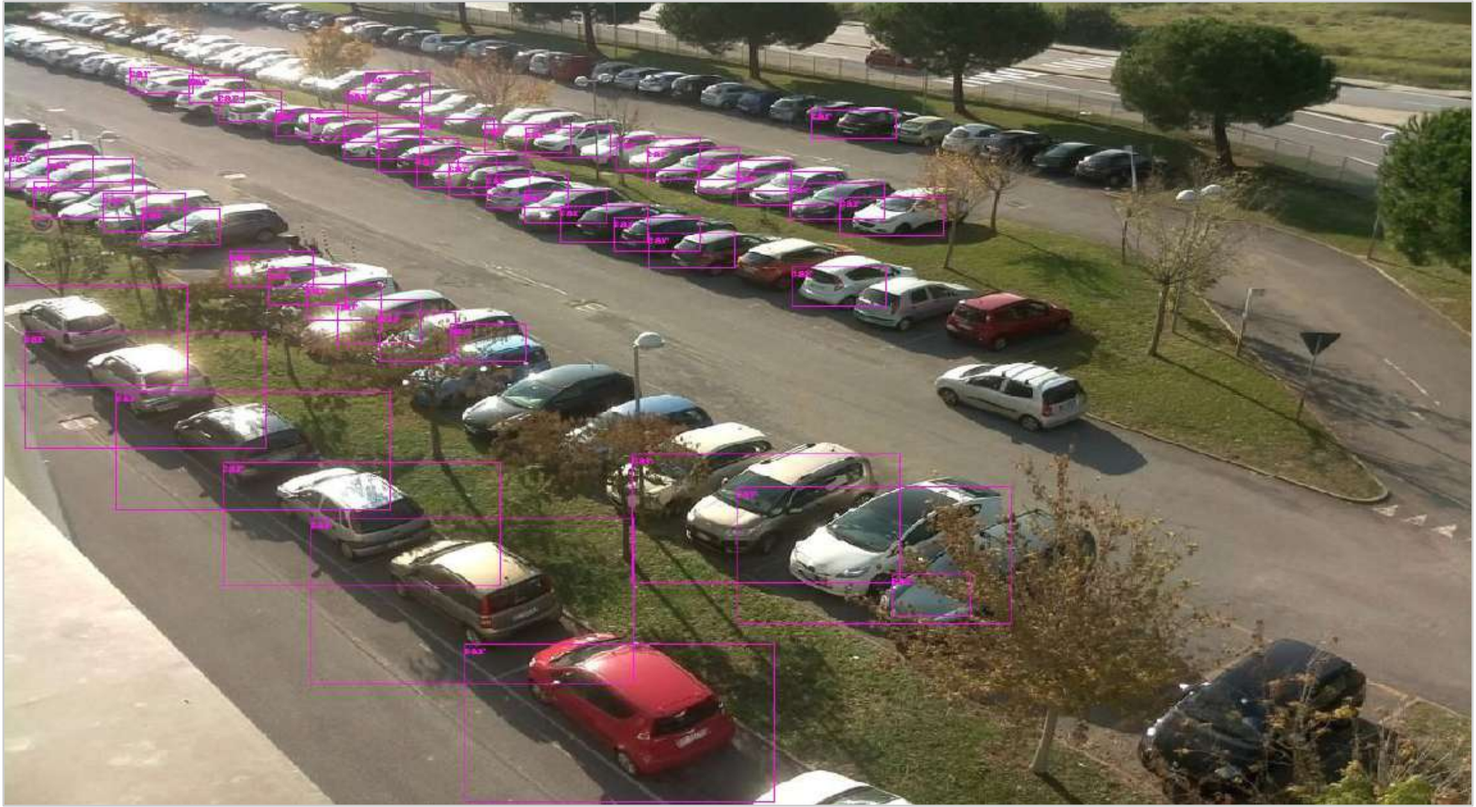
YOLO v3: Triple the Detection Layers

- Designed to detect a range of differently sized objects.
- Three detection layers at different levels of upsample.
- Routing layers concatenate a feature map with the “current” tensor.

CNN Detection Result: YOLO v3



CNN Detection Result: YOLO v3

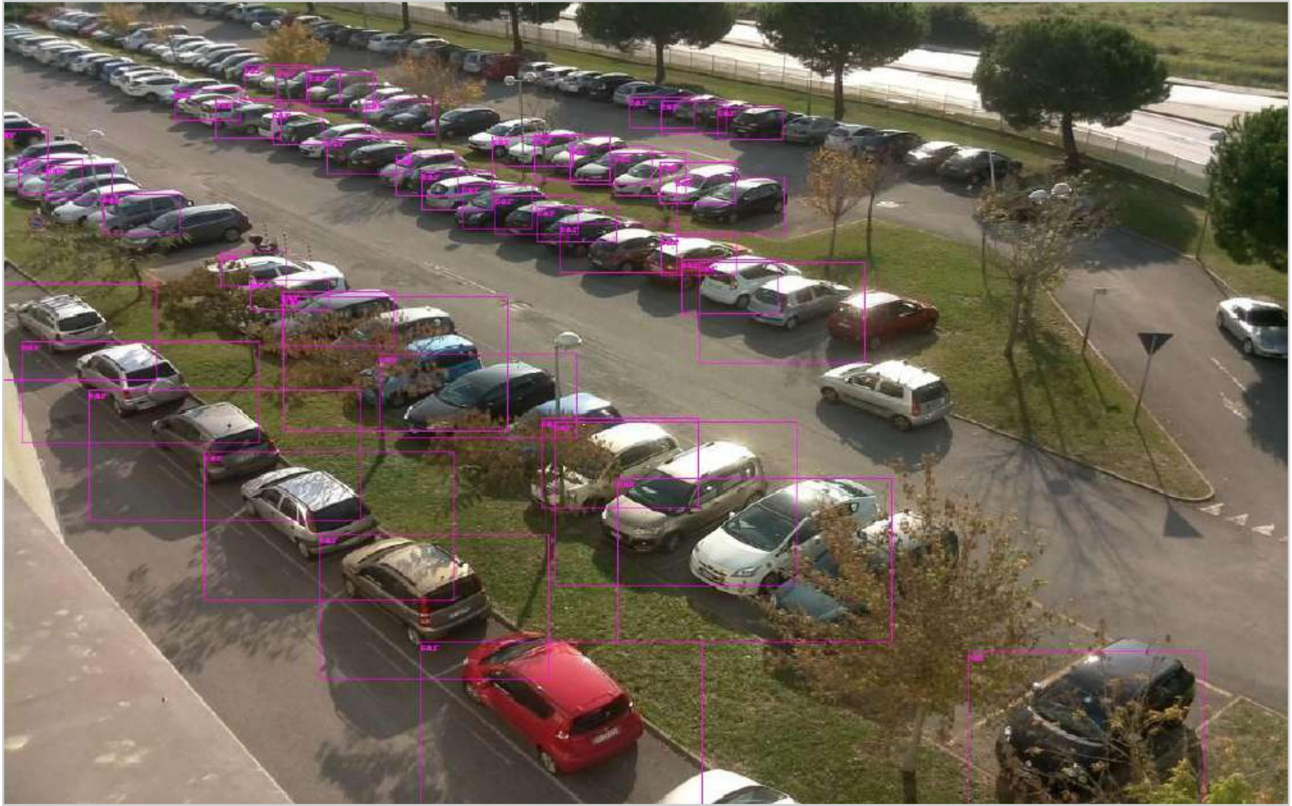


Modified YOLO v3: Fourth Detection Layer

- Same Darknet-53 feature extraction portion.
- Routed a fourth feature map to a corresponding fourth detection layer.
- Computed additional anchor boxes.

CNN Detection Result: 4th Detection Layer

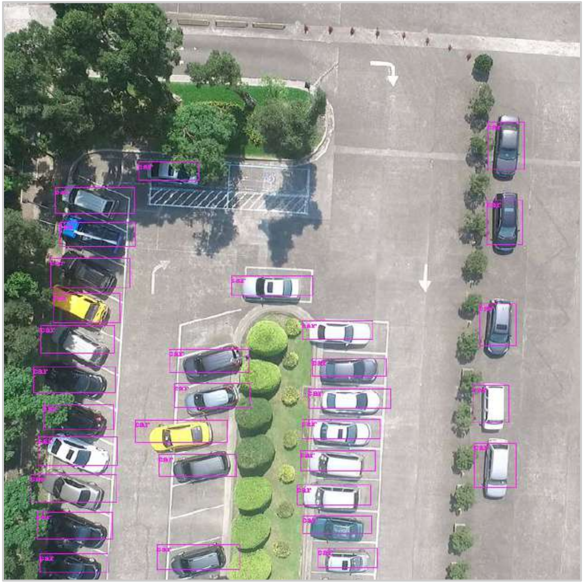
CNN Detection Result: 4th Detection Layer



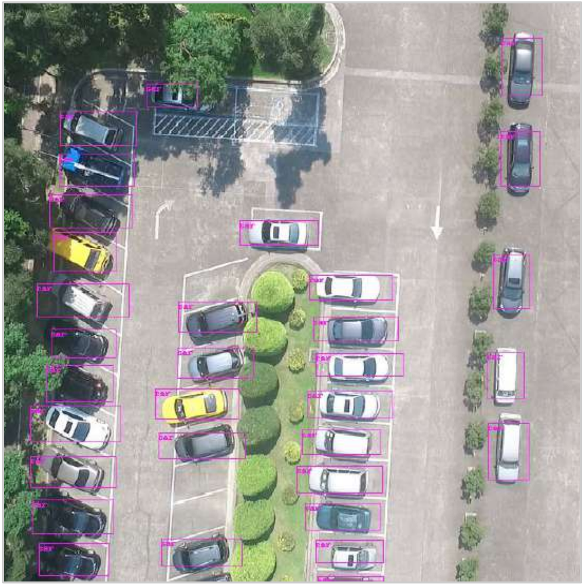
CNN Detection Results Evolution

YOLO v2

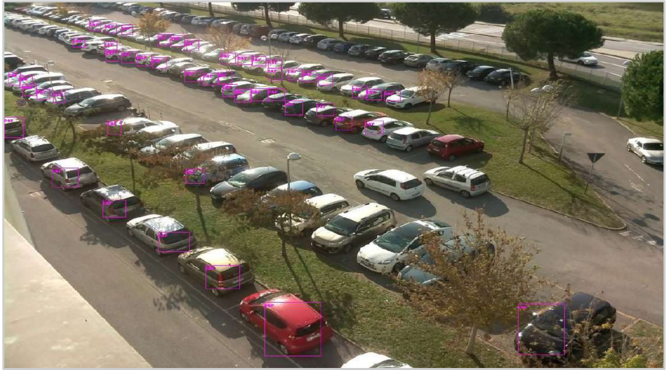
YOLO v3



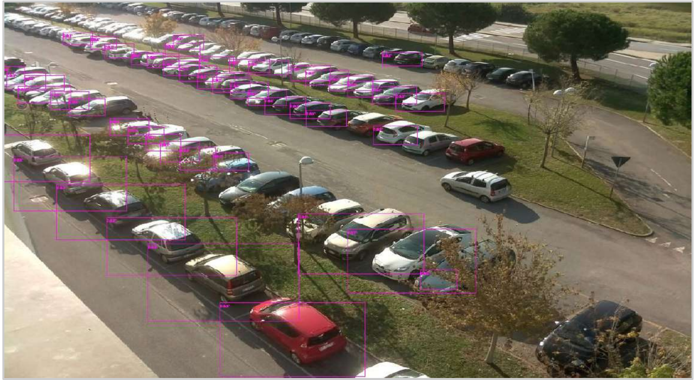
YOLO the 4th



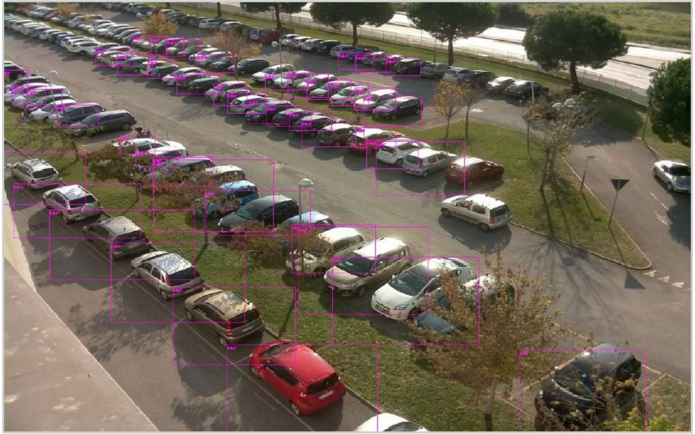
YOLO v2



YOLO v3



YOLO the 4th



Skynet Demo

https://www.youtube.com/watch?v=3brf_r1hnr0&t=38s

Conclusions

- Final results with the fourth detection layer modified YOLO CNN
- Research Environment: Overall parking lot car detection
  - Top-down detection is near-perfect
  - Fixed-camera detection progressively improves with more detection layers
- Skynet: Parking lot capacity monitoring
  - Top-down drone coverage is most ideal in detection accuracy
  - Fixed-angle is effective at parking lot capacity monitoring

Next Steps

- More comprehensively labeled parking lot datasets
- Beyond Darknet-53 → Deeper feature extraction portion
- Skynet video stream support
- Skynet web application
- Beyond parkings lots → inventory management

Thank You

Appendix

Appendix A - References

1. J. Redmon, et al, “You Only Look Once: Unified, Real-Time Object Detection,” arXiv, May 2016. [Online]. Available: https://arxiv.org/pdf/1506.02640.pdf
2.

Appendix B - Convolution Detailed Overview

- Vertical edge detection example
- Detailed look at padding and stride
- Convolution vs cross-correlation

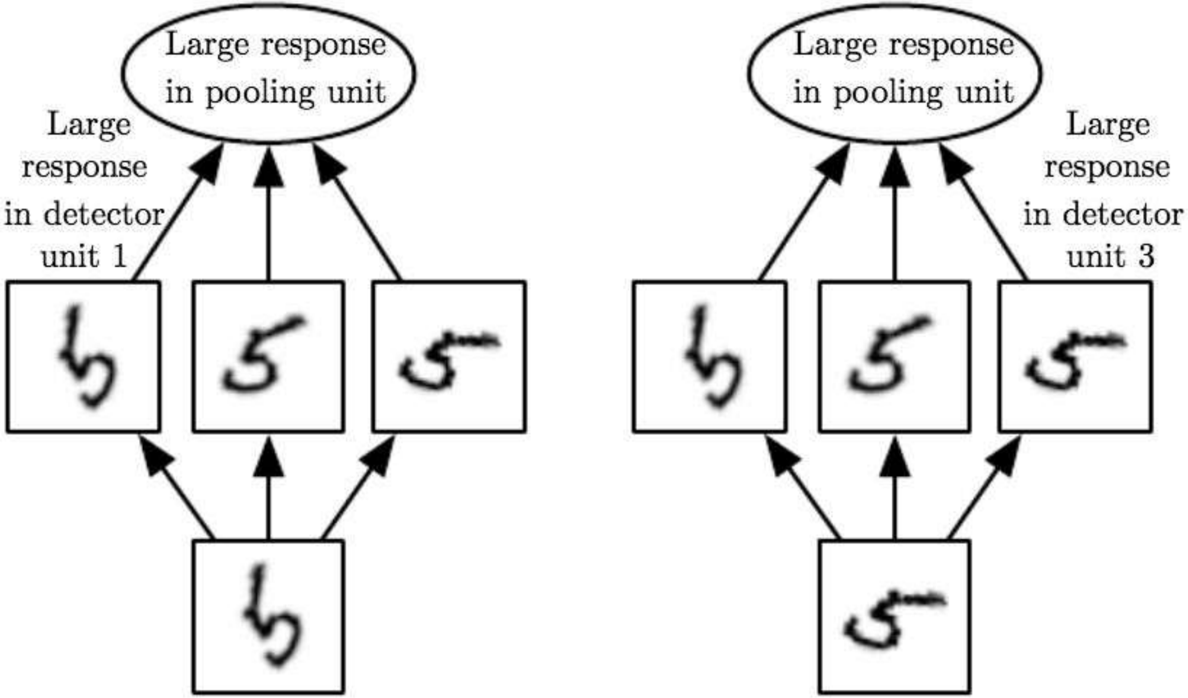
Appendix B - Vertical Edge Detection Example

Appendix B - Padding & Stride

Appendix B - Convolution vs Cross-Correlation

- Convolution in mathematics literature refers to a slightly different operation than Convolution done in deep learning
- Convolution in deep learning ~ Cross-correlation in mathematics

Appendix B: Pooling



Appendix C - Hardware Solution Shortcomings