Detect Corns in Drone Images with Deep Learning

Goup name: VT_CS_CS
Zhiyi Li
Ph.D. candidate in CS Department
Haidong Yan
Ph.D. candidate in School of Plant & Environmental Sciences
Advised: Dr. Song Li
School of Plant & Environmental Sciences
2019 VT IBM ARC Hackthon, Oct 11th, 2019

Deep Learning Addressable market

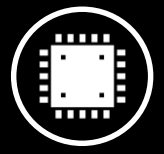
AUTOMOTIVE
ADAS, Maintenance

LAW & DEFENSE
Threat analysis - social media monitoring

CONSUMER GOODS
Sentiment analysis

FINANCIAL SERVICES
Risk analysis
Fraud detection

RESEARCH
Physics Modeling



MANUFACTURING
Line inspection, Defect analysis

LIFE SCIENCES
Sequence Analysis, Radiology

MEDIA/ENTERTAINMENT
Advertising effectiveness

CUSTOMER SERVICE
Chatbots, Helpdesk Automated Expenses

HEALTH CARE
Patient sensors, monitoring, EHRs

OIL & GAS
Exploration, sensor analysis

RETAIL
Reco. Engines, Precision Mktg

TRANSPORTATION
Optimal traffic flows, Route planning

UTILITIES
Smart Meter analysis, Capacity planning

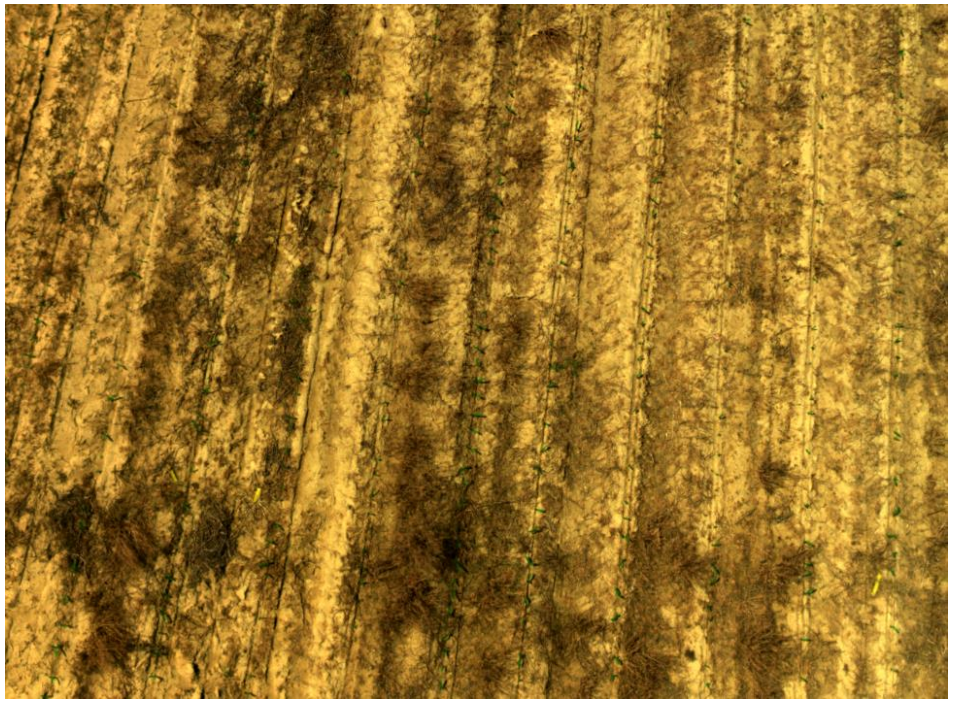
OTHERS
Agriculture, Remote Sensing

Summary

- Objective
- Background and previous work
- Problems and Solutions in Current work

Objective

Find corns in drone images

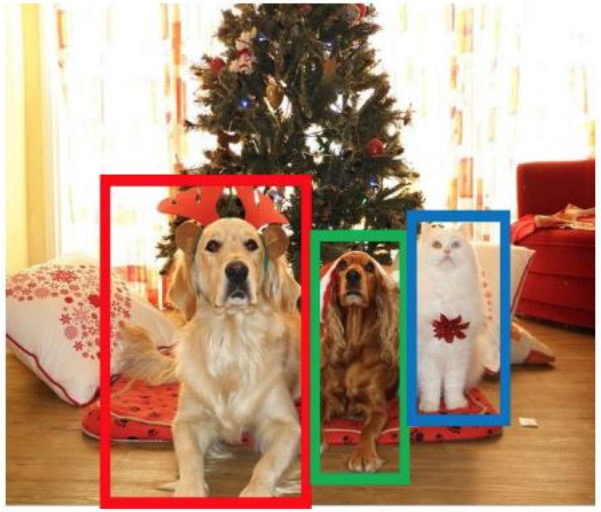


Original corn field images

Where?

How many?

Object Detection in Computer Vision



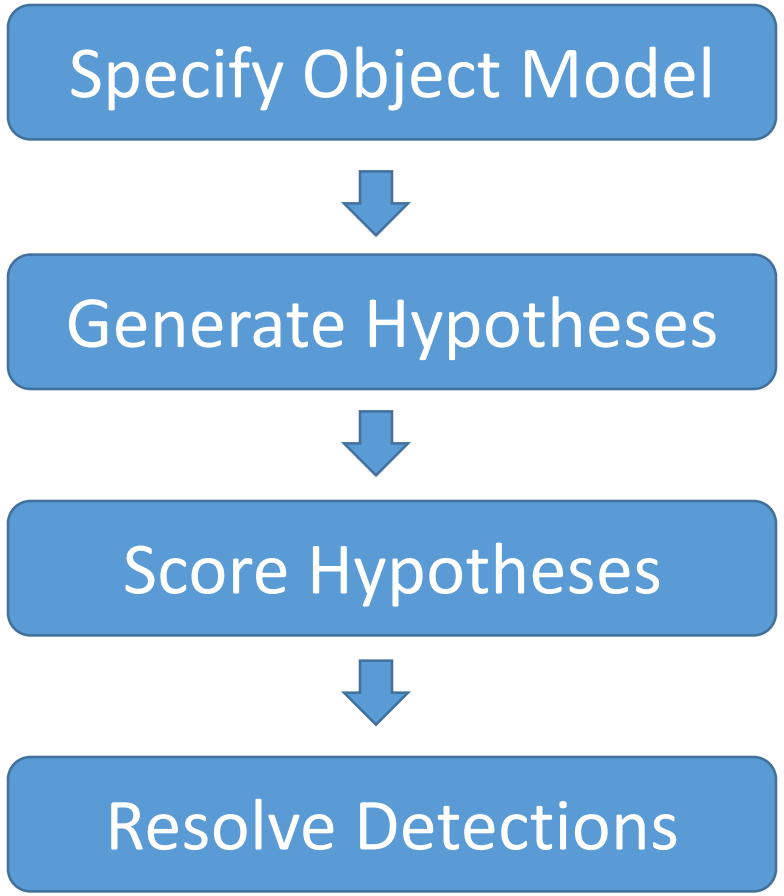
Slide credited from VT ECE 5554 taught by Jia-bin Huang

Object Detection in Computer Vision

- Traditional methods
  - Dalal-Triggs detector (basic concept)
  - Viola-Jones detector (cascades, integral images)
- Deep learning methods
  - Two-stage: R-CNN
  - One-stage: YOLO, SSD, Retina Net

Slide credited from VT ECE 5554 taught by Jia-bin Huang

Traditional General Process of Object Recognition



What are the object parameters?

Propose an alignment of the model to the image

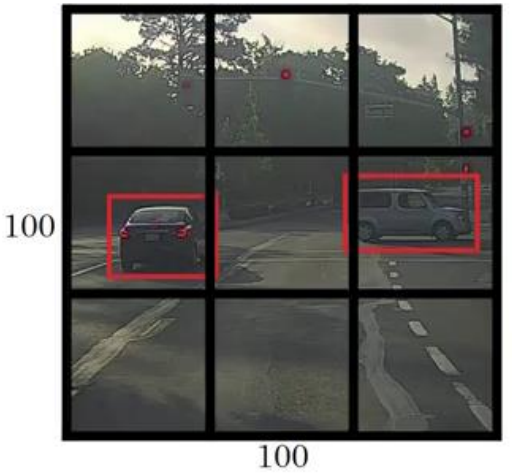
Mainly-gradient based or CNN features, usually based on summary representation, many classifiers

Slide credited from VT ECE 5554 taught by Jia-bin Huang

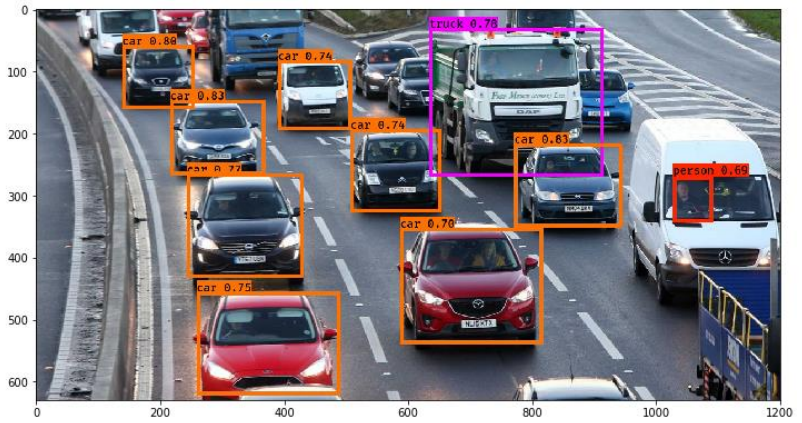
Previous work: Object Detection by YOLO(You only See Once)

- Model detection as regression method.

- Take an input image, divides the input image into grids say a 3 x 3 grid)



- Image classification and localization are applied on each grid). Predict bounding box and their corresponding class probabilities of objects.

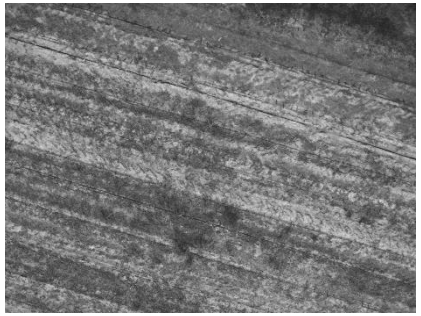


Credited from web: https://www.analyticsvidhya.com/blog/2018/12/practical-guide-object-detection-yolo-framewor-python/

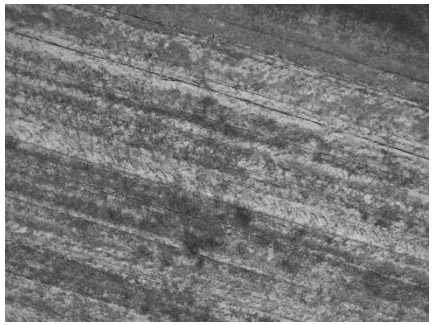
Previous work: Data Preparation for Deep Learning Model

- Align and Merge different channels of Images with MicaSense program.

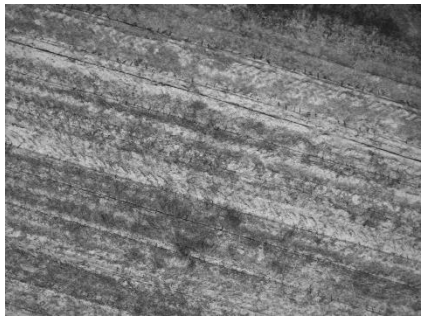
5 channel
MicaSense camera



R channel

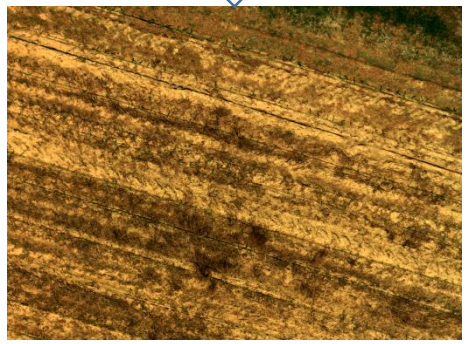


G channel



B channel

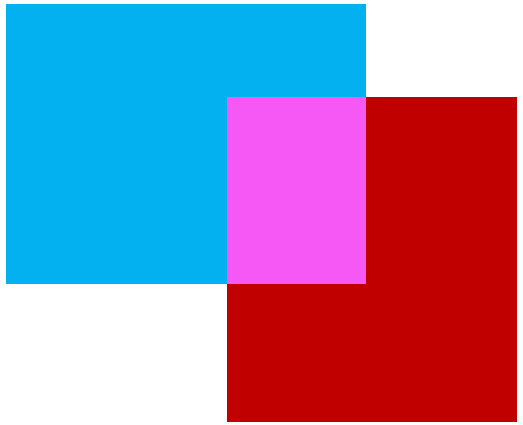
Align/Merge



RGB image

- Label image with imglabel tool
  39 training images
  18 testing images were labeled.

IoU: Matching detections to ground truth(bounding box)



$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Slide credited from web: *www.cs.cornell.edu › courses › lec36-obj-detn*

Matching detections to ground truth

- Match detection to most similar ground truth
  - highest IoU
- If IoU > 50%, mark as correct
- If multiple detections map to same ground truth, mark only one as correct
- **Precision** = #correct detections / total detections
- **Recall** = #ground truth with matched detections / total ground truth

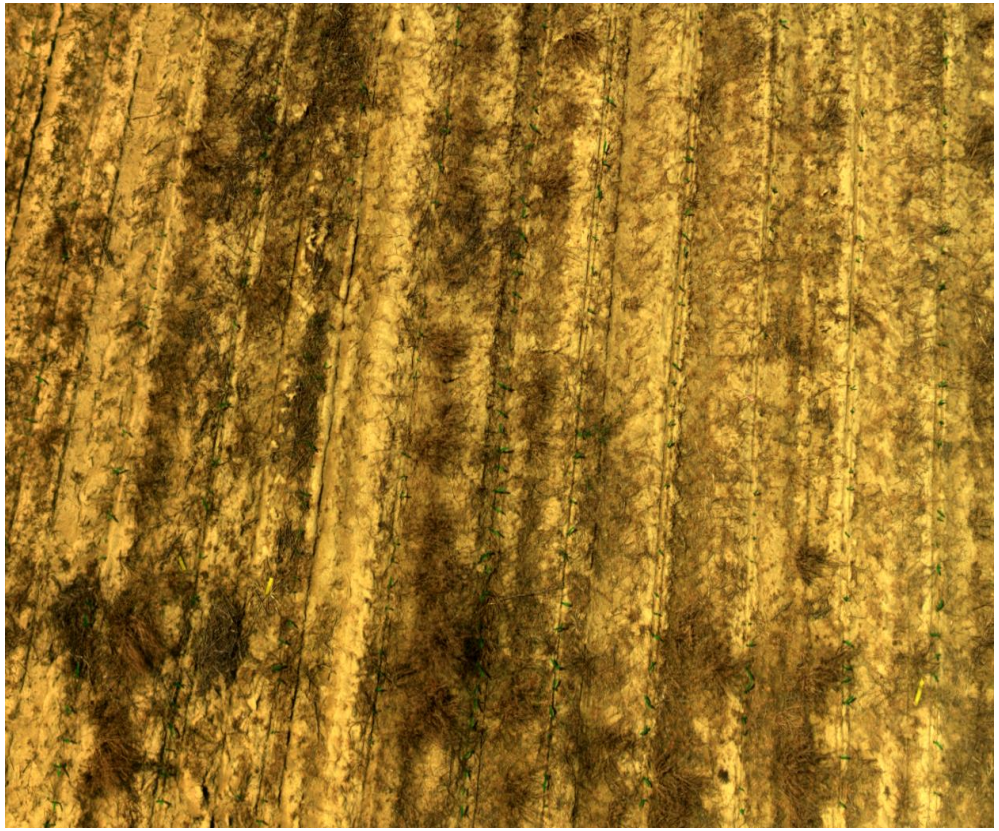
Slide credited from web: *www.cs.cornell.edu › courses › lec36-obj-detn*

Mean and category-wise AP

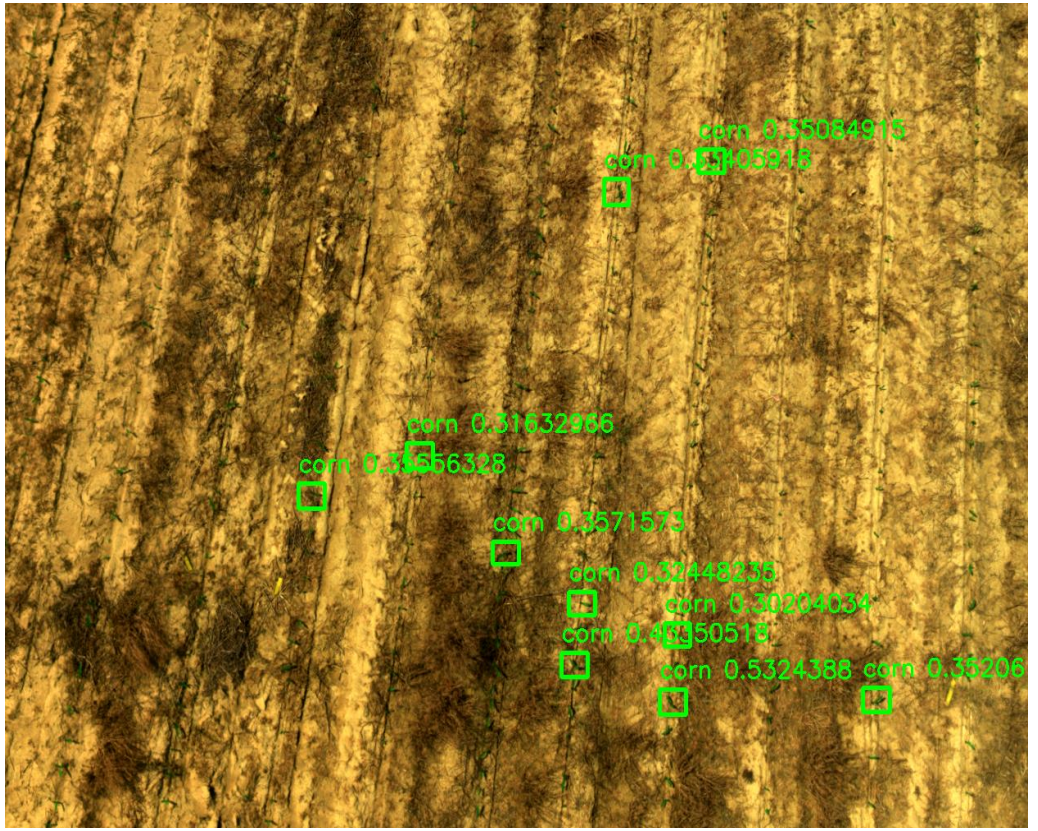
- Every category evaluated independently
- Typically report mean AP averaged over all categories
- Confusingly called “mean Average Precision”, or “mAP”

Slide credited from web: *www.cs.cornell.edu › courses › lec36-obj-detn*

Sample Detection Output with YOLO v2



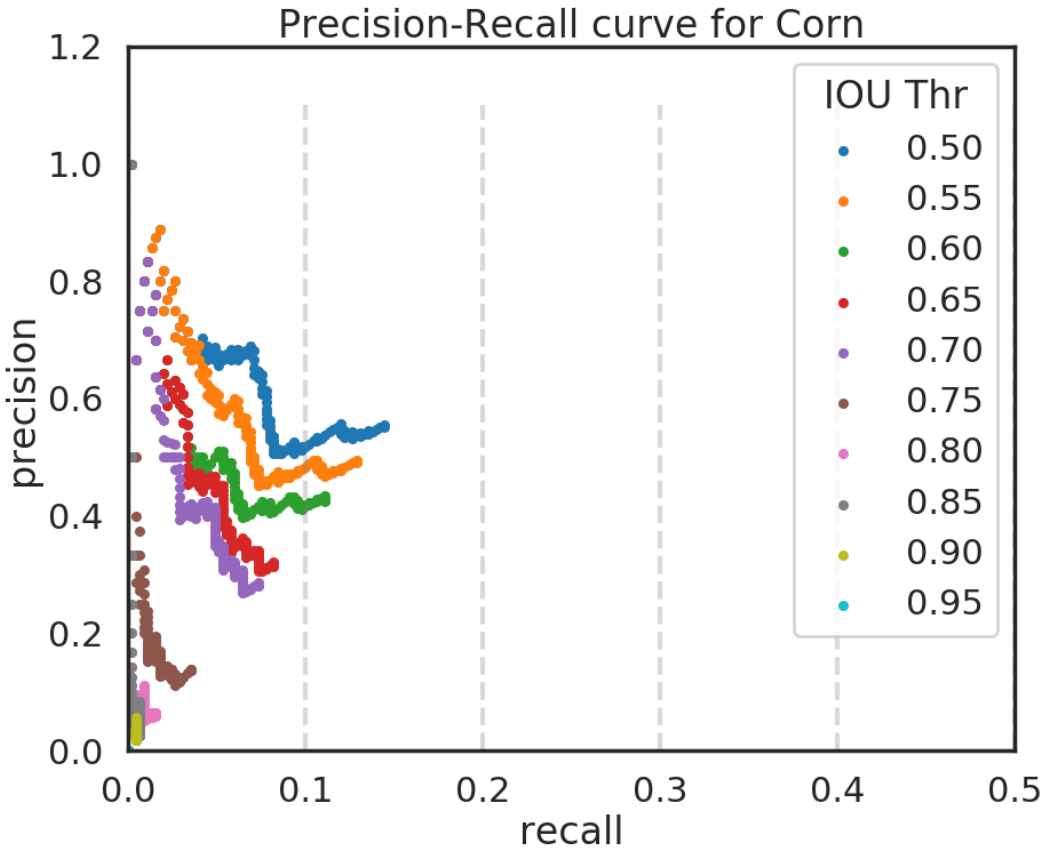
Original corn drone image IMG_0210



Detected corn in drone image IMG_0210

10 bounding Boxes found

Evaluation Results for Corn images in YOLO v2



YOLO v2 model can detect corns, but problem: mAP is low

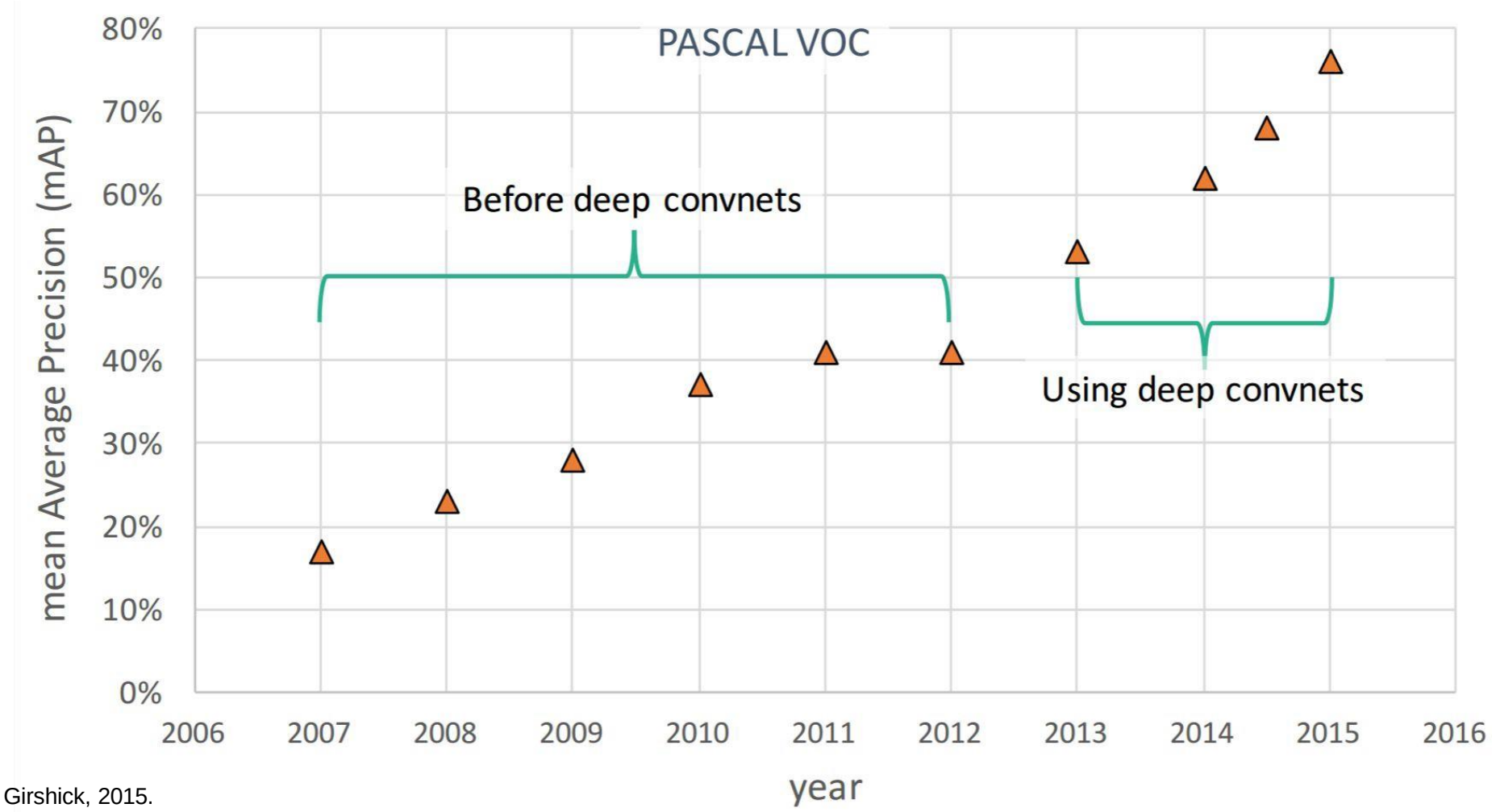
Current work

- Apply RetinaNet to object detect corns in smart farms
- Source code is from Open Source:

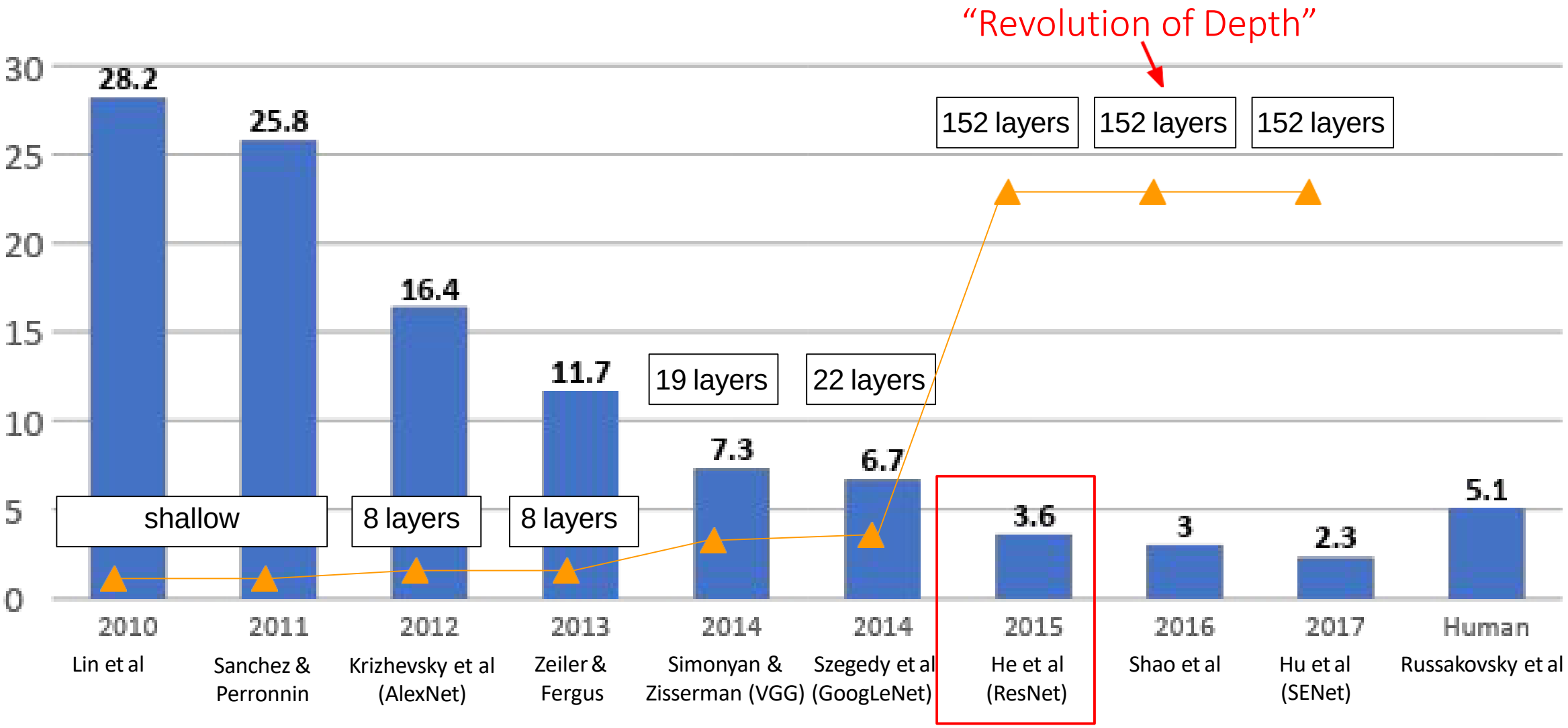
https://github.com/fizyr/keras-retinanet/

https://towardsdatascience.com/object-detection-on-aerial-imagery-using-retinanet-626130ba2203

Object Detection: Impact of Deep Learning



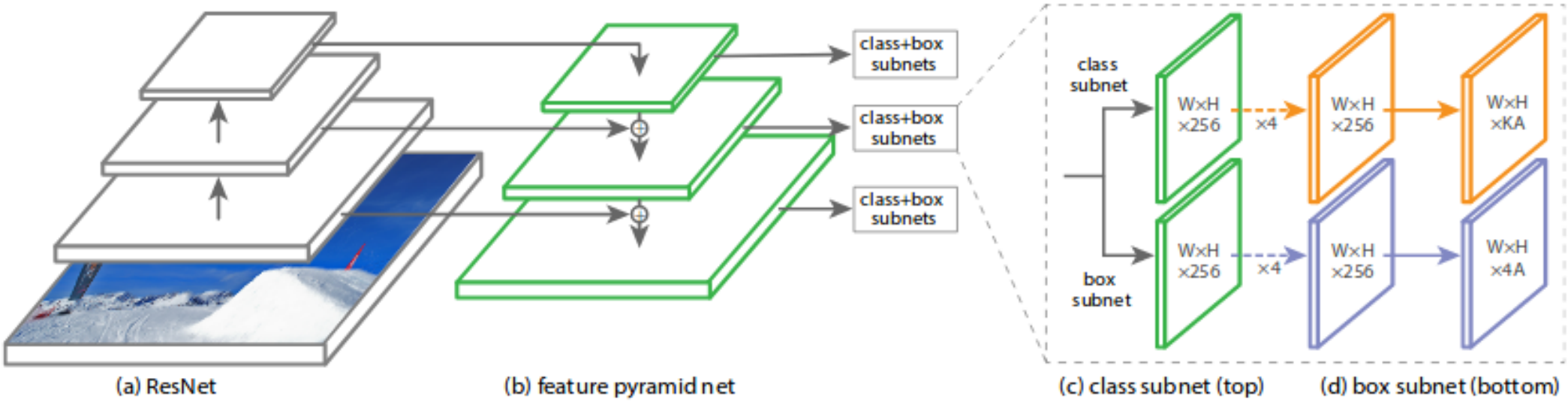
ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



Object Detection with RetinaNet for Corns

- One-stage object detection method. Based on ResNet50 or ResNet101.
- Surpassing the accuracy of all existing state-of art two-stage detectors.
- Apply pyramid network to detect objects in scale.

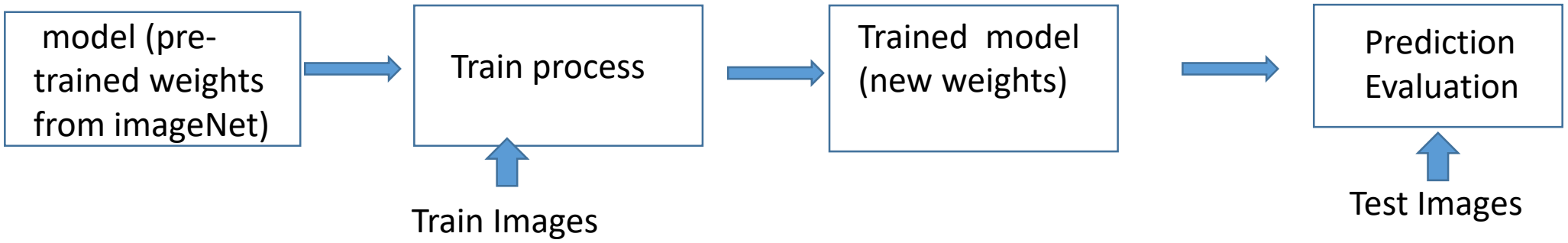
RetinaNet structure



Lin et al. “Focal Loss for Dense Object Detection”

python train.py -c config.json

Pipeline of Training/Test for RetinaNet model



- 39 training images
- 18 testing images
- Run in ARC huckleberry Server

Source: https://github.com/fizyr/keras-retinanet

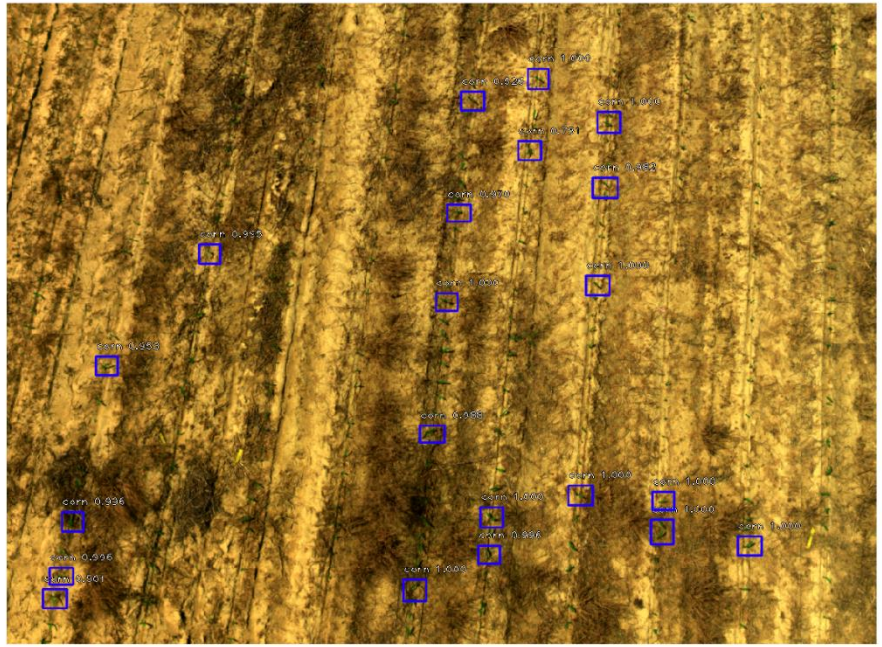
Train RetinaNet model

- $ python train.py csv path_to_train_image path_to_train_classes

  Runtime: 27070 s (1.5 days, default hyperparameter used)

- Evaluation of training error
- $ python evaluate.py csv
  ../../../Corn_Images_For_retinanet/train.csv
  ../../../Corn_Images_For_retinanet/classes.csv model.h5
  mAP: 0.7315

Predict corns in images with trained model



Sample detected Corns Image

- $ python predict.py
- It works. Draw bounding box
- Effect better than YOLO v2.

Model transfer, Evaluation

- Convert Training model to Inference Model

  $ python convert_model.py path/to/train/model.h5
  path/to/save/inference/model.h5

- Evaluate model with testing data.

  $ python evaluate.py csv ../../../Corn_Images_For_retinanet/test.csv ../../../Corn_Images_For_retinanet/classes.csv model.h5

  mAP: 0.2176, better than YOLO v2 results, But….

Problem: Overfitting.

Scale Training Process in ARC Server

- Available scale resources in VT ARC:
- IBM PowerAI distributed deep learning (DDL) is a MPI-based communication library, can be used to scale train process.
- DDL support ddlrun and LMS(Large Model Support).

- Problems:
- Data parallel or model parallel is hard for our RetinaNet model because:
- RetinaNet model is complicated: ResNet50 + FeaturePyramidNet.
- Corn image training size is small(39 images).
- Divide small training samples in ddlrun may not work.

Scale Training Process

- Solutions: Instead we run ddlrun for tf_cnn_benchmark.py to simulate training process.
- The tf_cnn_benchmark.py script is with backbone resnet50, train with synthesized imageNet data for classification task.
- Example ddlrun:

```
$ ddlrun -H host1, host2 python tf_cnn_benchmarks.py –
  variable_update=ddl --model=resnet50  --num_gpus=1 –
  batch_size=128 --num_batches=10000
```

Scale training process in ARC Machine

Node, GPU Number	Running Time (Seconds)
1 Node, 4 GPUs	5205
2 Node, 8 GPUs	5248
3 Node, 12 GPUs	5283

Table 1. Running time of ddlrun for tf_cnn_benchmark.py for different setting. Different 3 hosts were used.

- Problem: No variance, not scale at all
- Possible Reasons: Security issue and communication overhead among hosts. Even ssh each other needed.

Set up Jetson Nano

- Problem: Jetson Nano version does not support anaconda

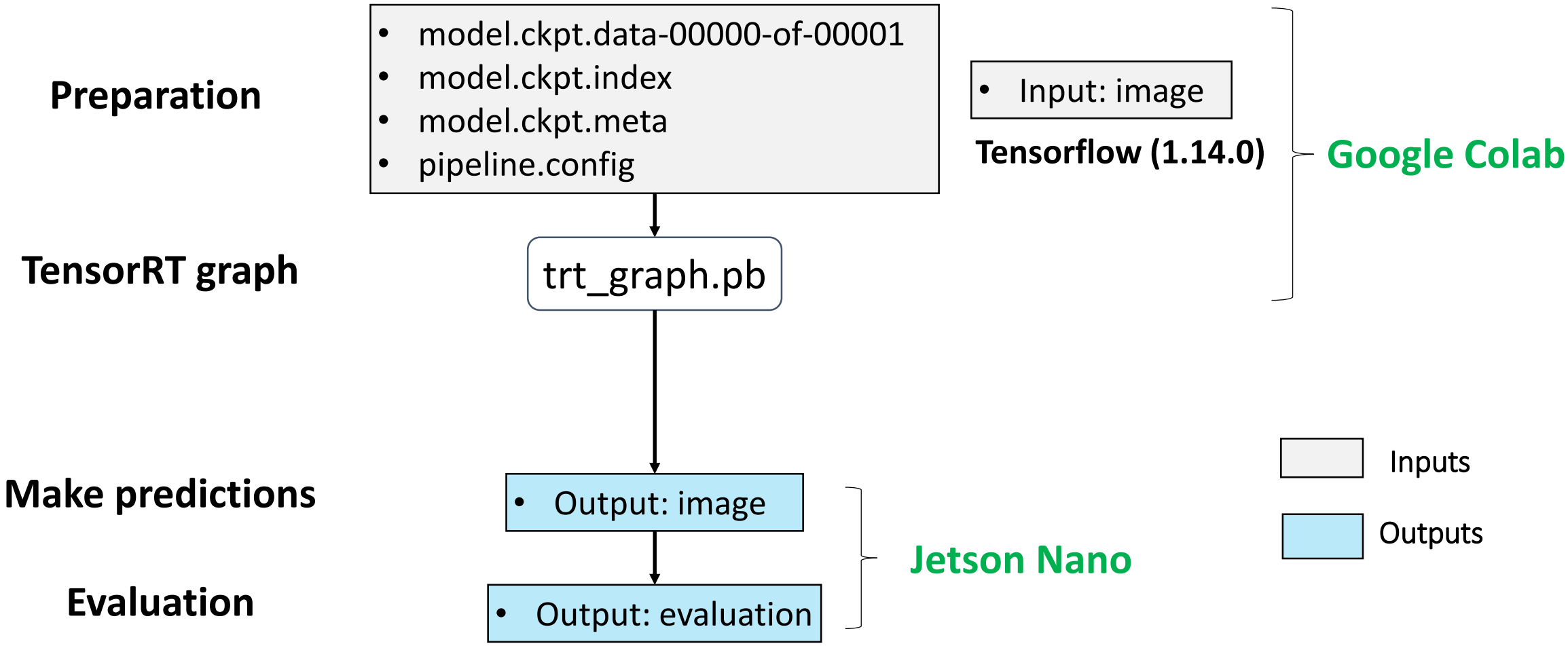
  Installed

  Solution: Install

  - ❑ python3.6
  - ❑ keras (2.3.0)
  - ❑ tensorflow (v1.14.0)

- Installed system from ground-up
- Monitor, keyboard, mouse, Even WIFI setup

Run Tensorflow object detection model on Jetson Nano

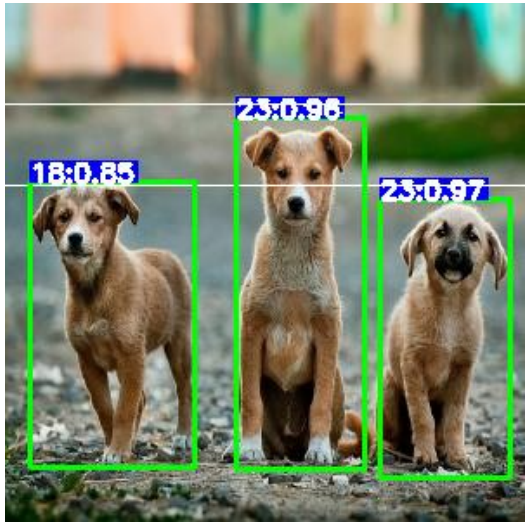


Problems we found in Jetson Nano

- For object detection, retinaNet is not found corresponding version for Jetson Nano.
- Other object detection methods: ssd_inception_v2_coco, and ssdlite_mobilenet_v2_coco available
- Checkpoints and config files in training process were required to create model for Jetson Nano.
- However, our RetinaNet model training process is default train, these two files not available.

Run Tensorflow object detection model on Jetson Nano

- Problem: We are not able to conduct object detection on our own image due to RAM(memory) limitation.
- Alternative solution: Run pre-train model called SSD MobileNet V2 (ssd_mobilenet_v2_coco) in Jetson Nano.



- Class 18 means a dog.
- Class 23 suggest a bear.
- The right two dogs were incorrectly classified as bears.

Run Keras model for Prediction on Jetson Nano

Predicted: [('n02504458', 'African_elephant', 0.6990039), ('n01871265', 'tusker', 0.16008943), ('n02504013', 'Indian_elephant', 0.03917188)]

- We successfully recognize the elephant in the picture is African elephant.

Several lessons learned for running tensorflow on Jetson Nano

- It is recommended to use colab (google) to run codes to generate TensorRT inference graph.
  - It is easy to install some packages that are hard to install (such as keras-retinanet) in regular server.
- Colab must install tensorflow with version 1.14.0 that is compatible with version of tensorflow in Jetson Nano
- We need prepare checkpoint files and pipeline.config file training in the Object detection model.

Lessons learned from this Hackthon

- Find problems and possible solutions for research work.
  - Model overfitting problem.
  - Test ARC IBM PowerAI DDL function, Scalable problems.
  - Learned how to deploy Deep Learning Model to edge devices.
  - Zhiyi Li even earned NVIDA DLI Certificate “get start with AI on Jetson Nano”

Future work

- Improve retinaNet model:
  Weakly supervised learning,
  More image dataset,
  Data Augmentation,
  Hyperparameter tuning…

- Deploy to Jetson Nano in drones.
  Real-time detection.

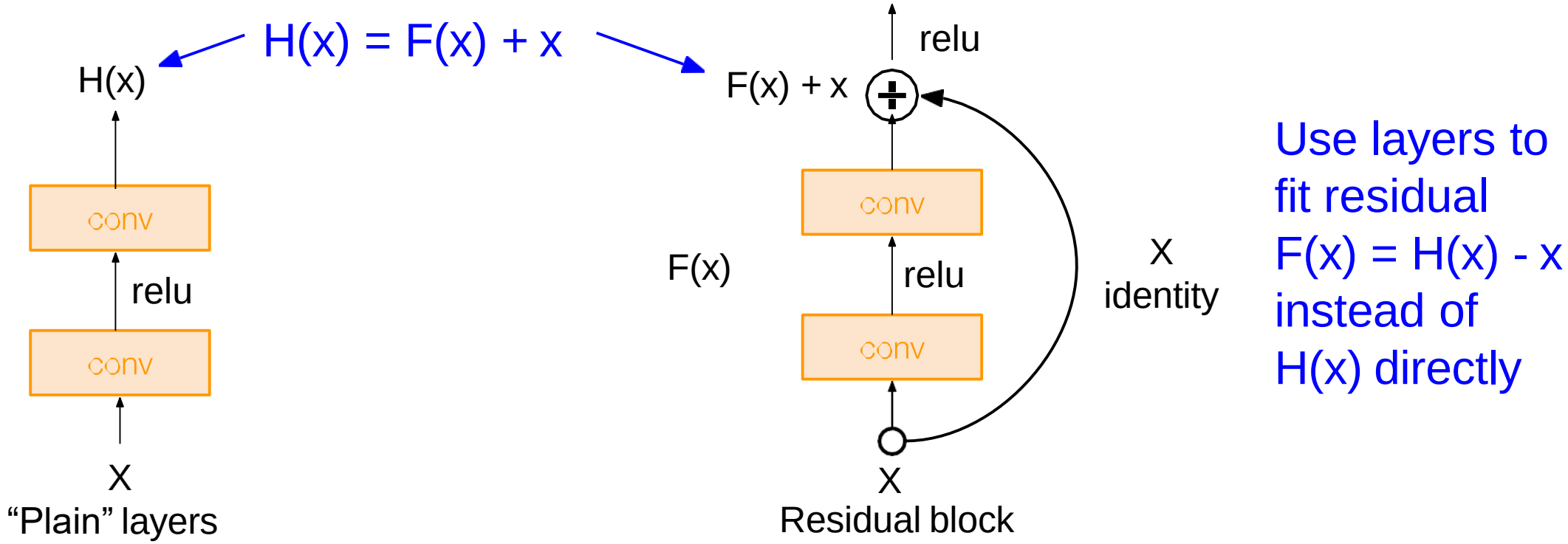
Acknowledgement

- This work is supported by Virginia corn board and usda.
- Thanks support by VT ARC.

Case Study: ResNet

[He et al., 2015]

Solution: Use network layers to fit a residual mapping instead of directly trying to fit a desired underlying mapping



Use layers to fit residual $F(x) = H(x) - x$ instead of $H(x)$ directly