Comparative Analysis of Temporal Convolutional Networks for Object Classification

Valli Chidambaram, Justin Davis, Paris Dinh, Amari Hoogland, Sofie Lange, Kamen Shah, Lei Teng, & Zhengwu Yuan

Overview

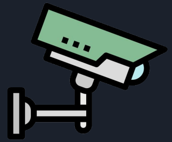
Problem

Traditional object classification models exhibit intrinsic problems when inferring on video sequences

Goal

Design a model that can account for previous classifications in the hope to reduce temporal instability in classifying objects within videos

Applications



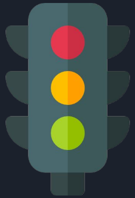
Surveillance Systems

Face Detection

Self-driving Vehicles

Sports Analysis

Autonomous Systems



Traffic analysis / optimization

Approach

1. **Dataset**
   Obtain a dataset with annotated bounding boxes for video sequences

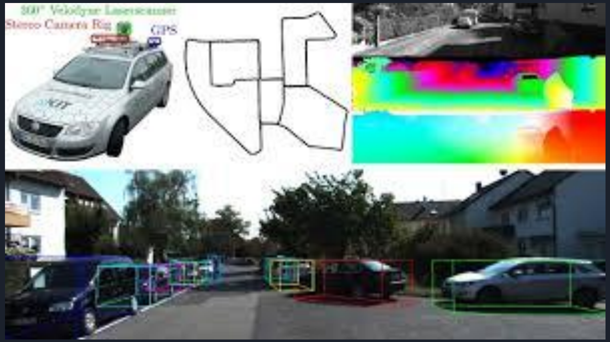
2. **Implement Base Network**
   Run YOLO V1 to establish baseline

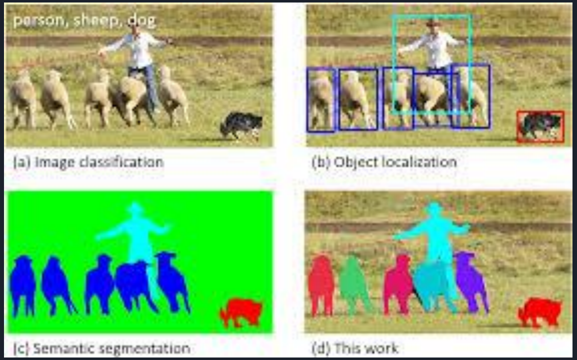
3. **Integrate Time-Dependent layers**
   - RNN
   - LSTM
   - TCNN

4. **Hyperparameter Tuning**

Dataset Selection

- Originally looking at vehicle datasets only
- Many datasets contain classified vehicles in static, stand-alone images, e.g. KITTI, COCO, Stanford Cars
- Difficult to find high-quality, publicly available labelled training data if we limit the scope to only vehicles
- Settled on **YouTube Bounding Boxes**





Dataset

YouTube Bounding Boxes - augmented with classical object tracking from OpenCV

Class labels include - bicycle, bus, dog, cat, truck, car, train, potted plant, person, etc.

- Bounding boxes and class labels for ~240,000 publicly available YouTube videos
- 15-20s video segments with b.b. at 1 fps
- 23 different classes
- features objects in natural settings without editing or post-processing
- recording quality similar to a hand-held cell phone camera

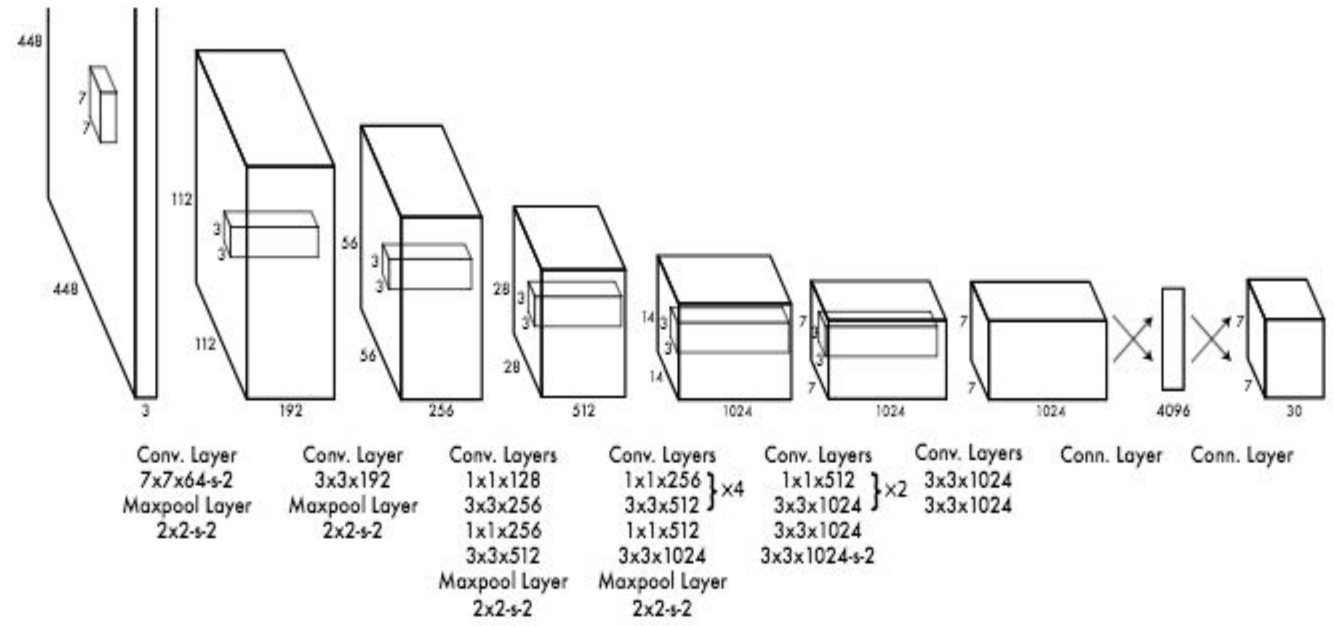
Problems with Dataset

- Unbalanced classes, far more people than anything else
- Bounding boxes from tracking are inaccurate, sometimes disappear
- Most videos only have one labeled object, but several unlabeled objects
  - Loss increases when predicting unlabeled boxes during training
  - Precision decreases when predicting unlabeled boxes

Models

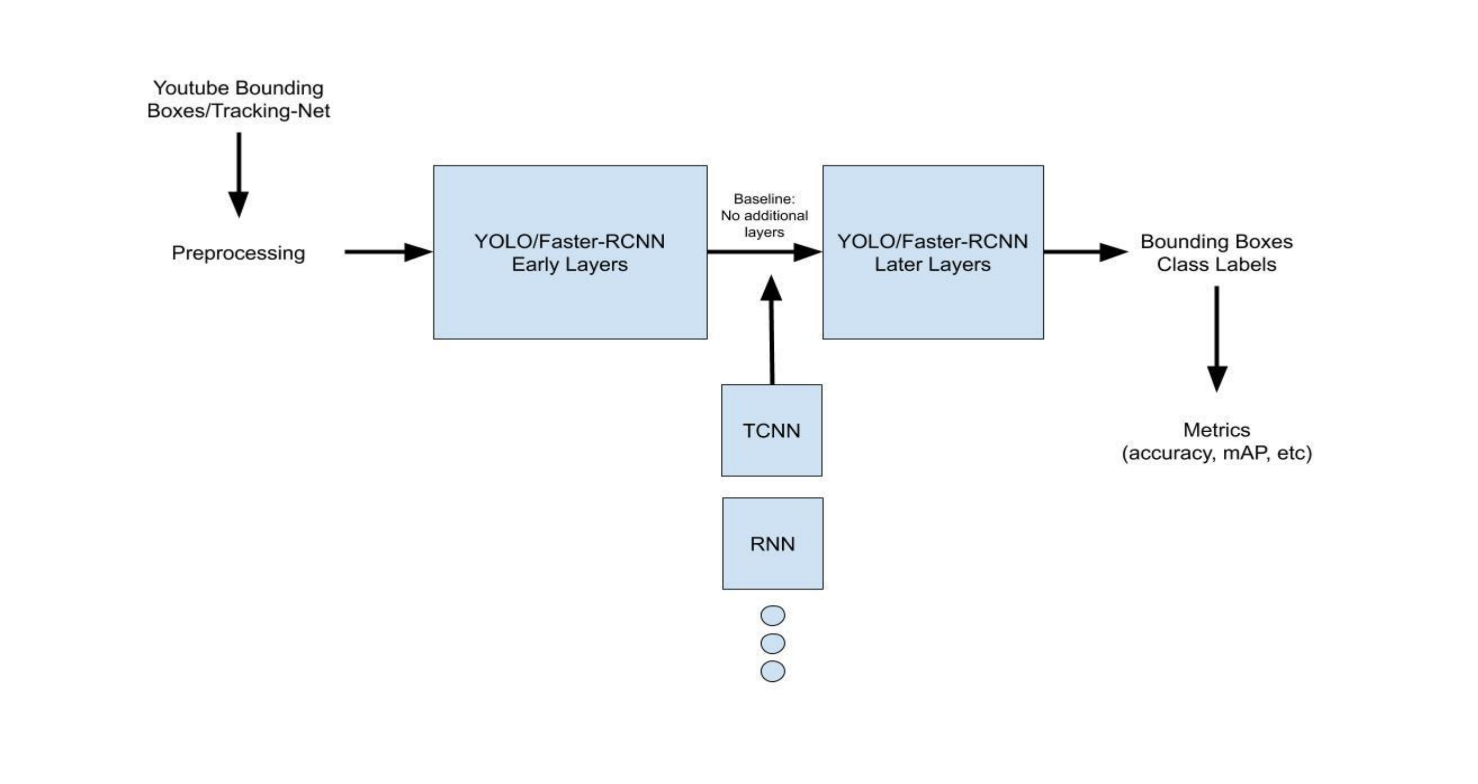
YOLO v1 - Baseline	RNN	LSTM	TCNN	Attention
• Convolutional neural network • Predicts bounding boxes and class probabilities • YOLO is extremely fast (45 frames per second)	• Recurrent Neural network • Exhibits temporal dynamic behavior.	• Long Short Term Memory • Deals with the vanishing gradient problem	• Temporal Convolutional Neural Network • Convolutional architecture that combines simplicity, autoregressive prediction, and very long memory for video sequences	• Mechanism based off the concept of directing focus to certain factors with data • Usually used bidirectionally on text data

YOLO V1



It consists of 24 convolutional layers where the first 20 convolutional layers followed by a maxpool layer and the last 4 convolutional layers followed by 2 fully connected layer.

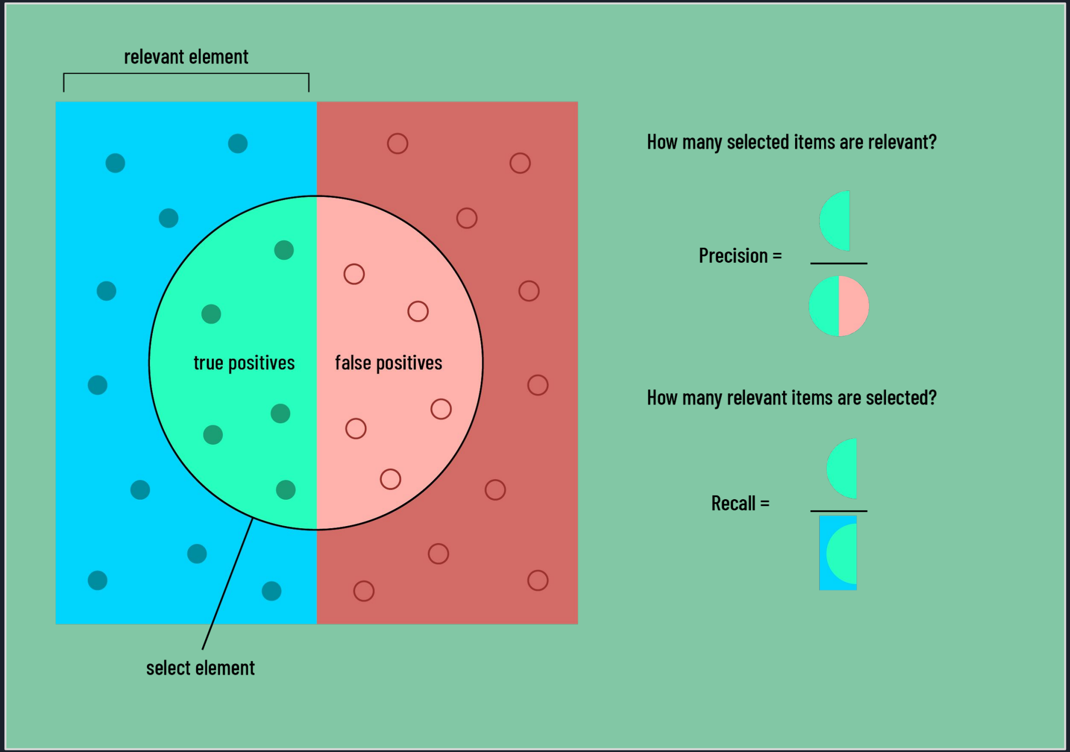
Architecture Diagram

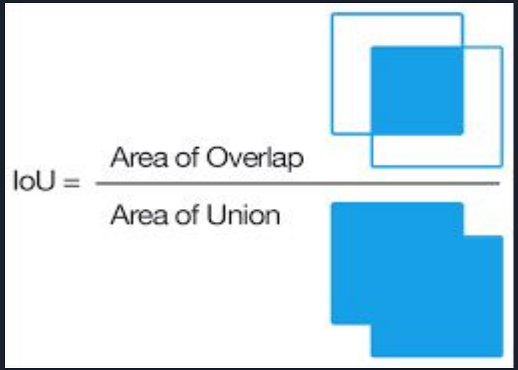


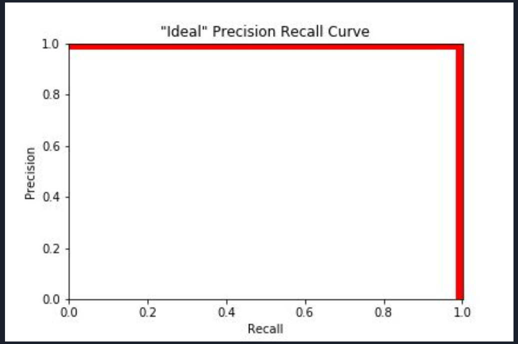
Results

- Individual Model Metrics
  - Recall v. Precision
  - Mean Average Precision Score (mAP)
- Comparative Results
- Visualizations

Precision, Recall, and IOU

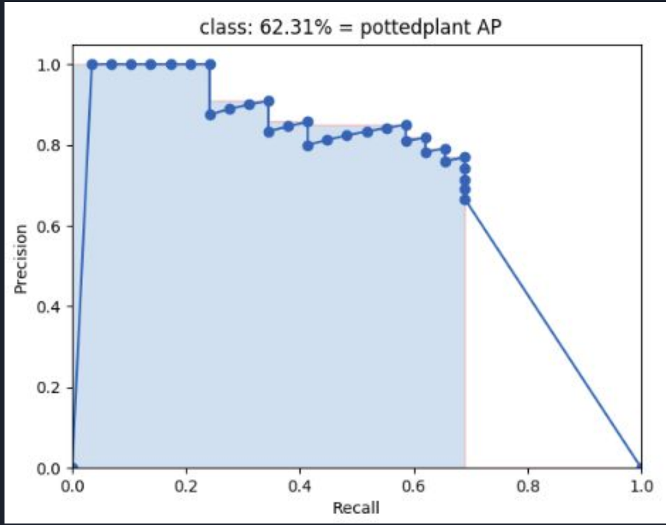






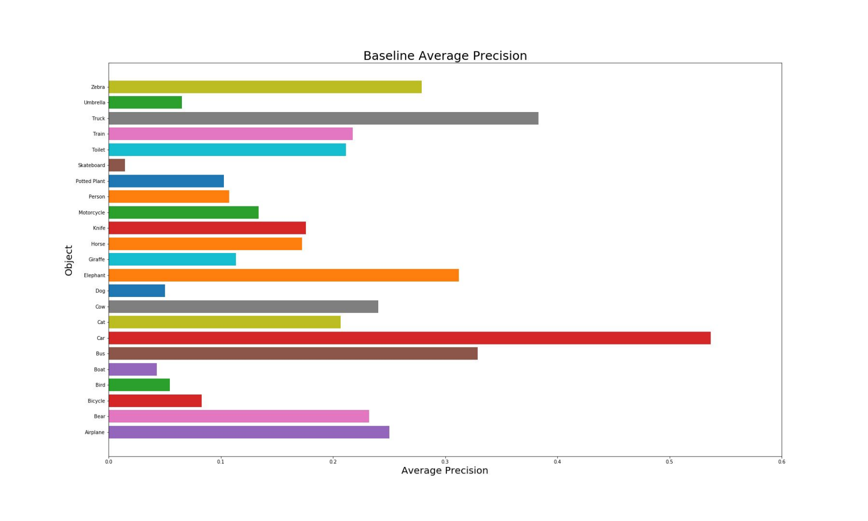
mAP (Mean Avg. Precision) Score

- Commonly used metric for object detection algorithm comparison
- Used to evaluate performance in popular object detection challenges such as [MS COCO]() and [PASCAL VOC]()



Example graph from https://github.com/Cartucho/mAP

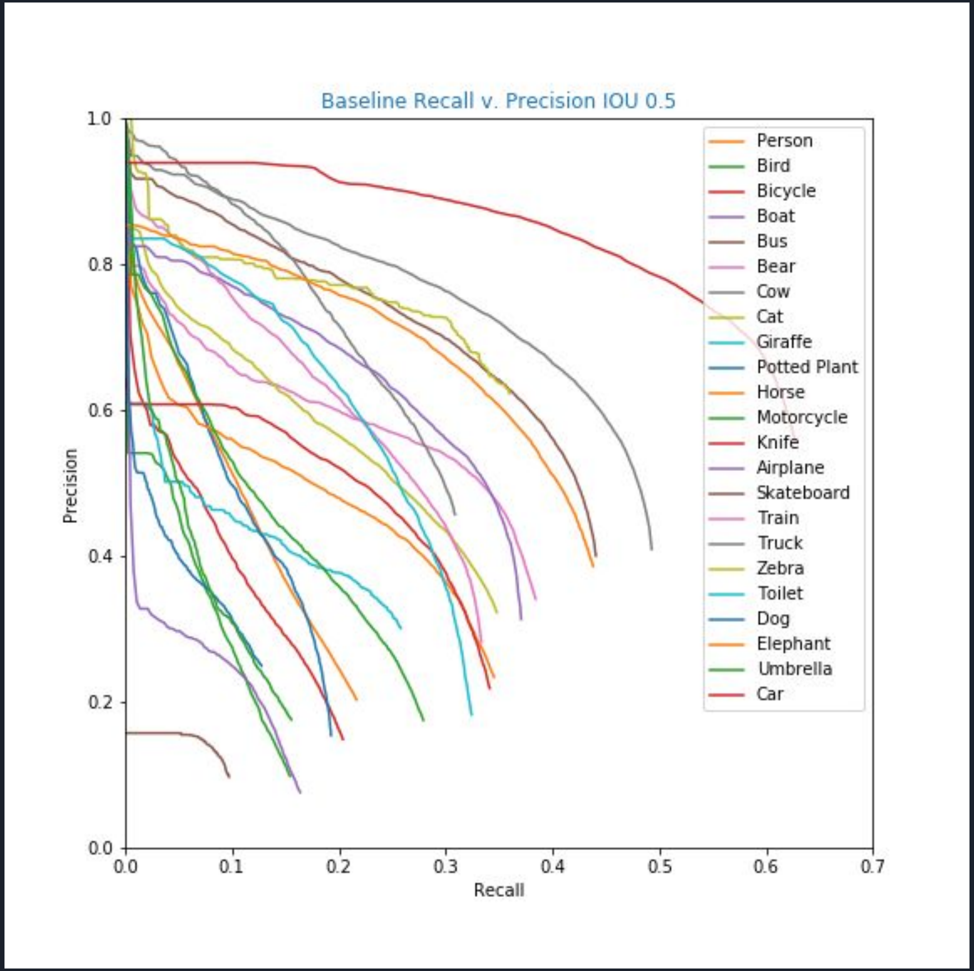
AP is the area under this curve **per class**



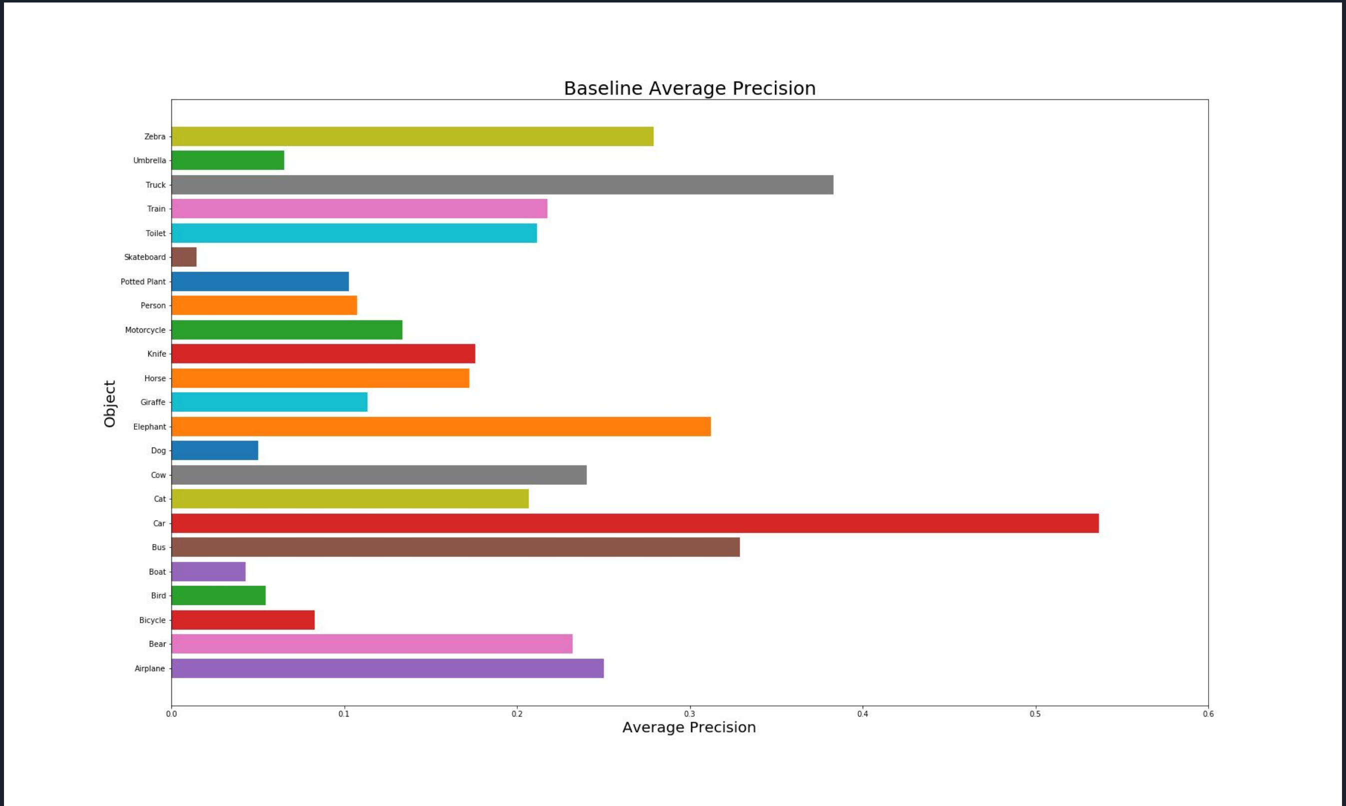
mAP is the average of AP scores for all classes

YOLO v1

- Baseline



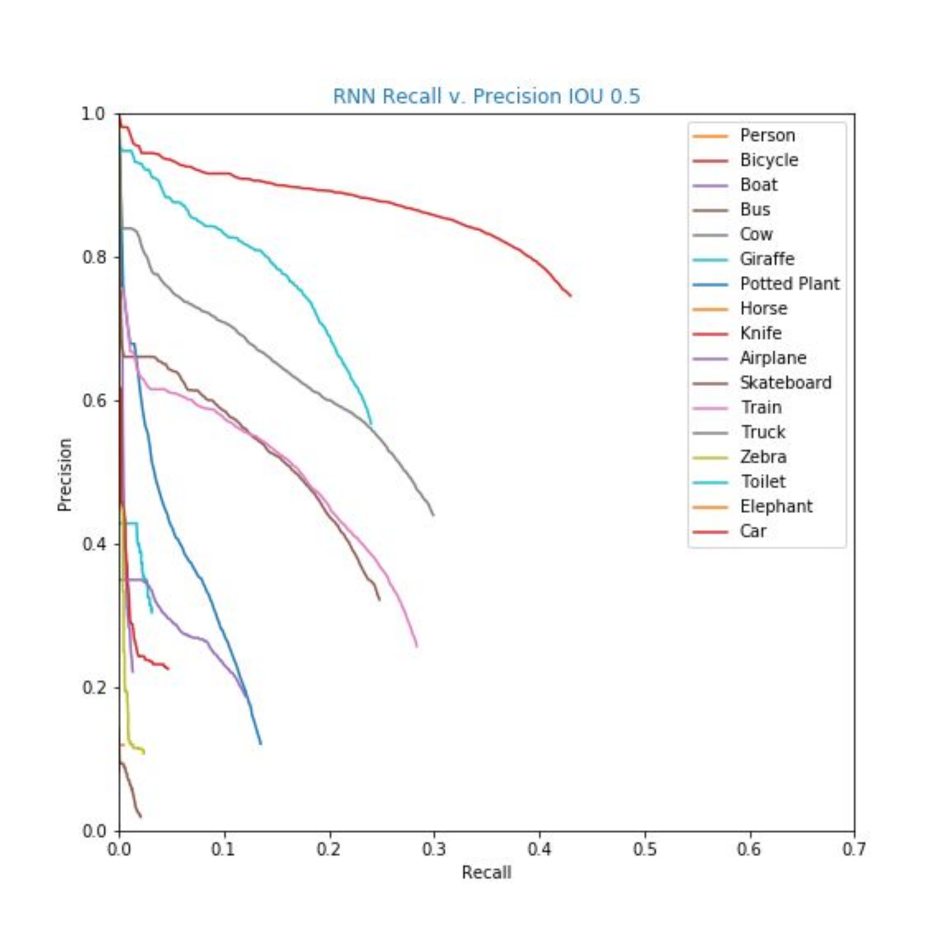
YOLO v1



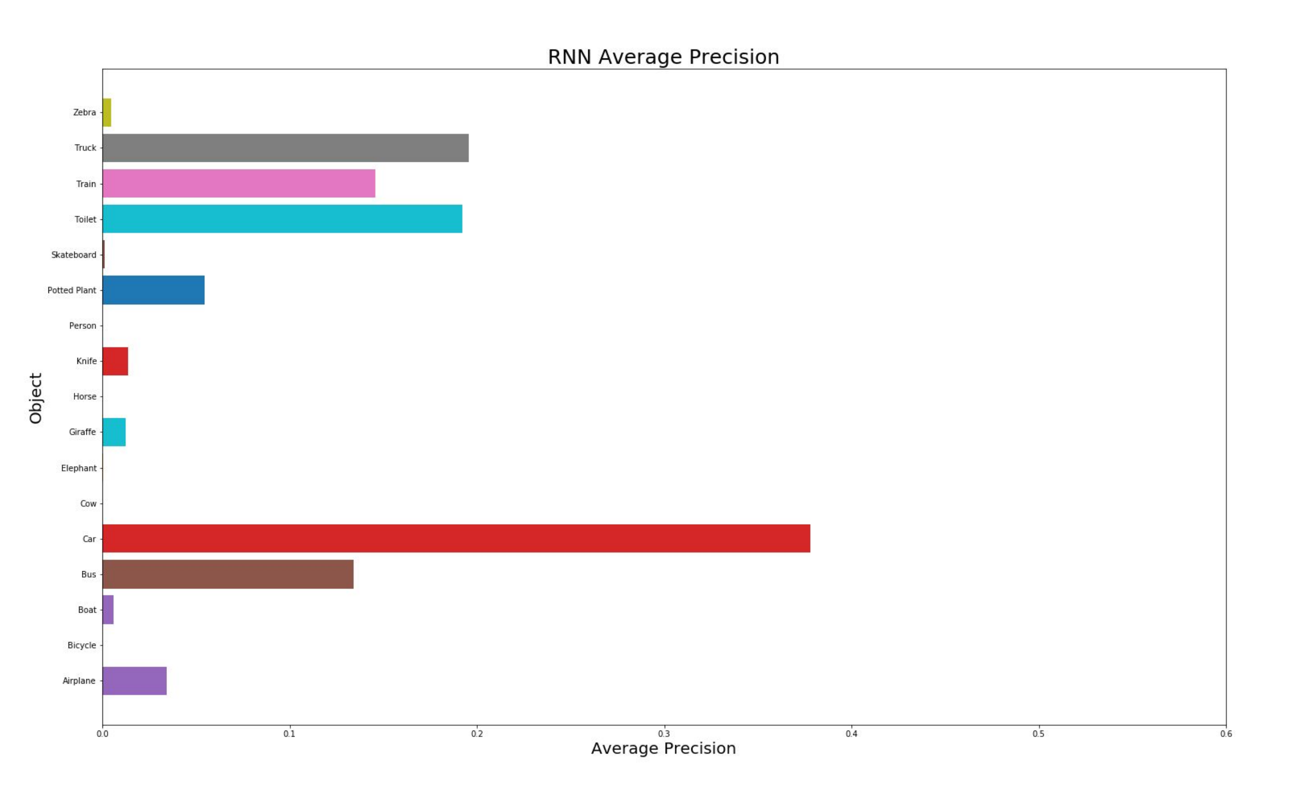
Baseline Average Precision
Zebra
Umbrella
Truck
Train
Toilet
Skateboard
Potted Plant
Person
Motorcycle
Knife
Horse
Giraffe
Elephant
Dog
Cow
Cat
Car
Bus
Boat
Bird
Bicycle
Bear
Airplane
Object
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Average Precision

RNN

- Hard to train, loss wouldn't decrease
- Lots of tuning to get it to work at all



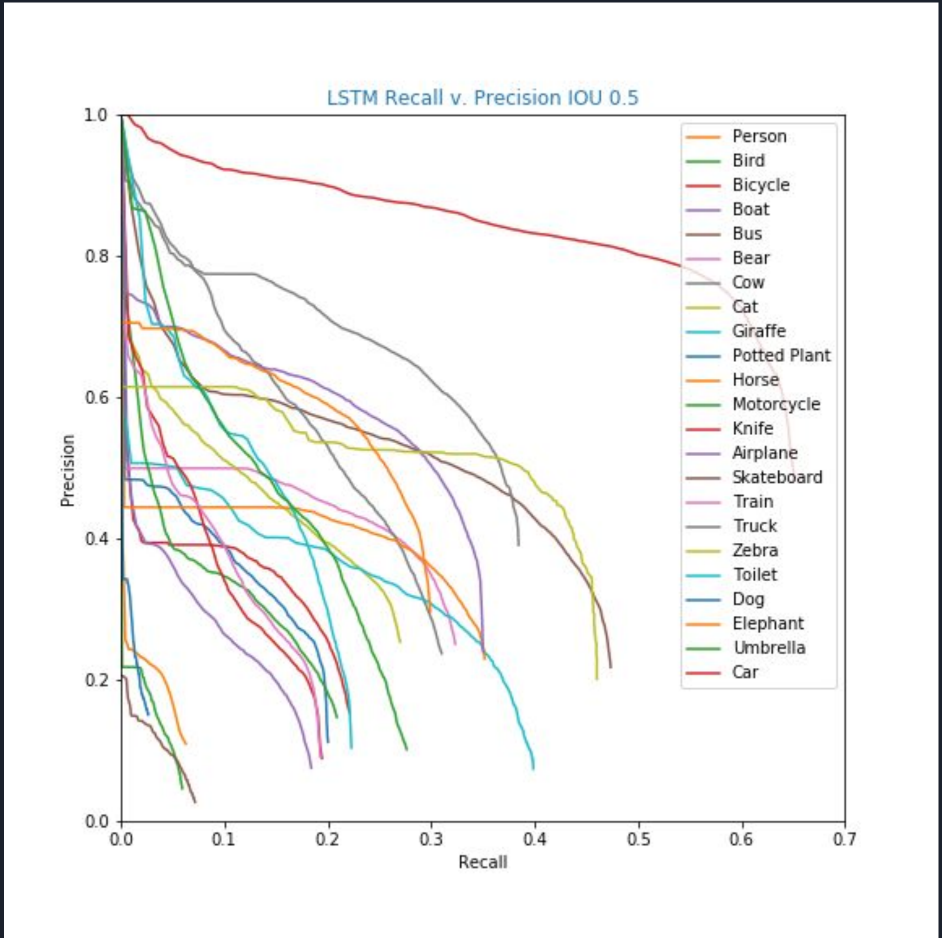
RNN



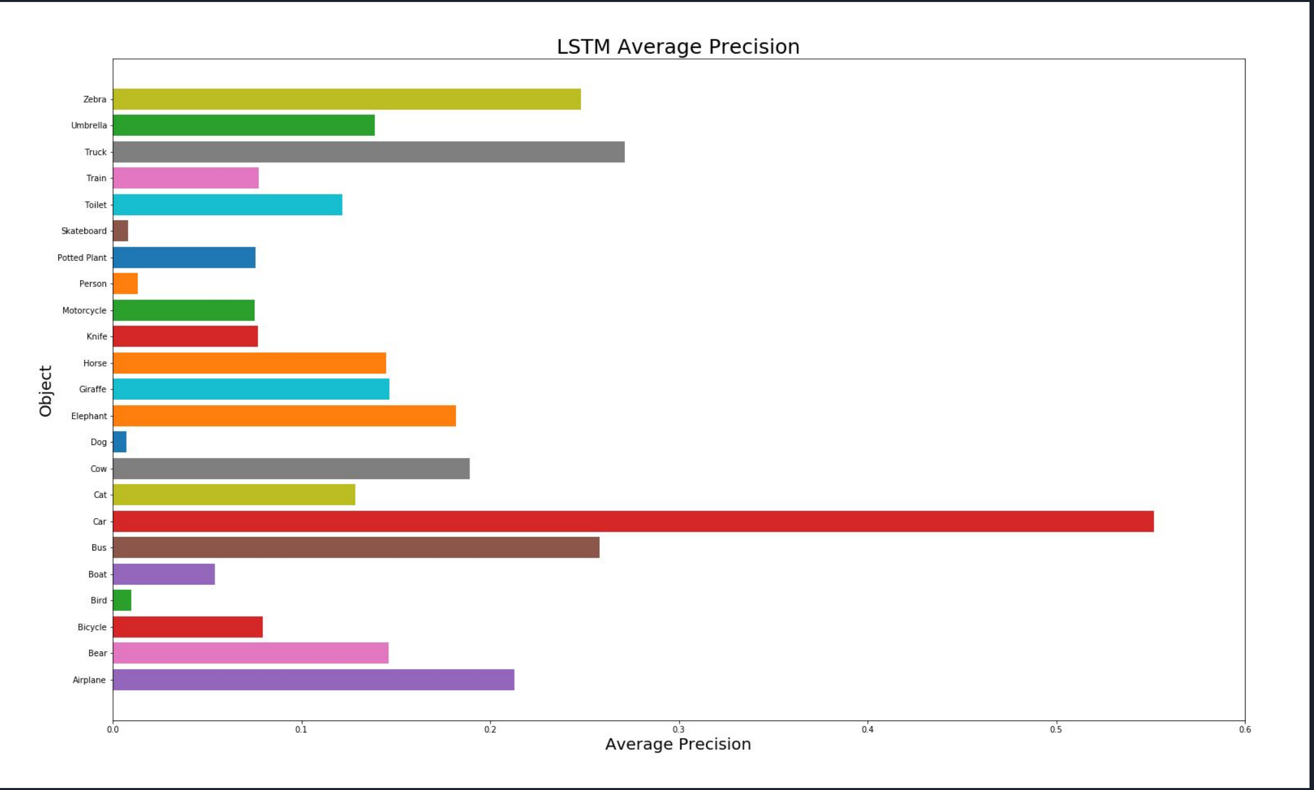
RNN Average Precision
Zebra
Truck
Train
Toilet
Skateboard
Potted Plant
Person
Knife
Horse
Giraffe
Elephant
Cow
Car
Bus
Boat
Bicycle
Airplane
Object
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Average Precision

LSTM

- Best performance of time-dependent models

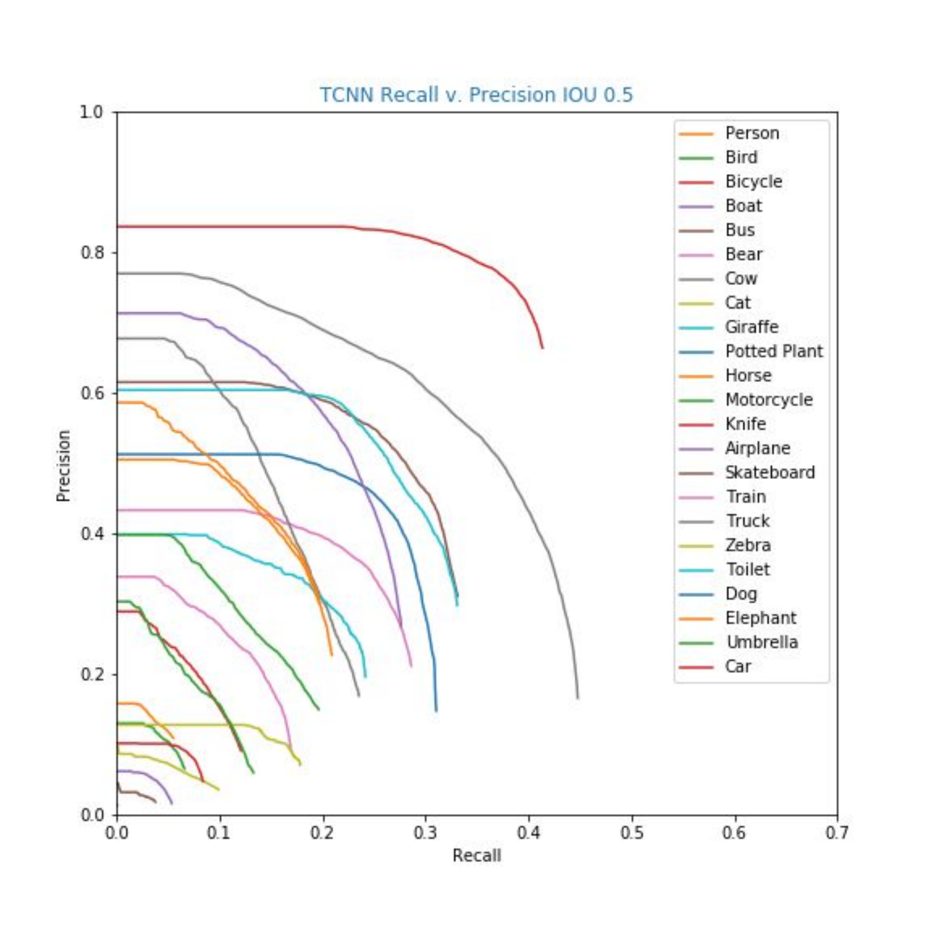


LSTM

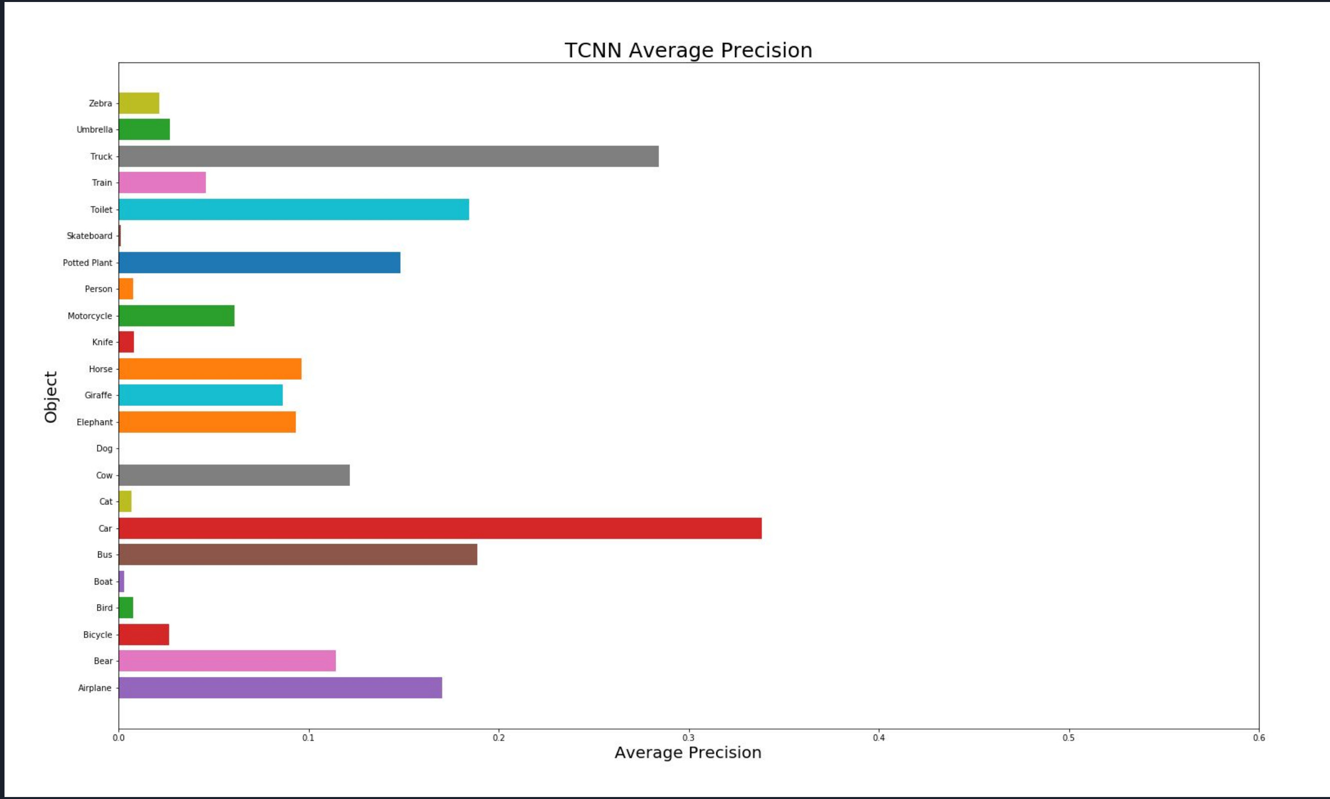

LSTM Average Precision
Zebra
Umbrella
Truck
Train
Toilet
Skateboard
Potted Plant
Person
Motorcycle
Knife
Horse
Giraffe
Elephant
Dog
Cow
Cat
Car
Bus
Boat
Bird
Bicycle
Bear
Airplane
Object
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Average Precision

TCNN

- Only trained once
- No changes to architecture
- No tuning
- Surprisingly well performing



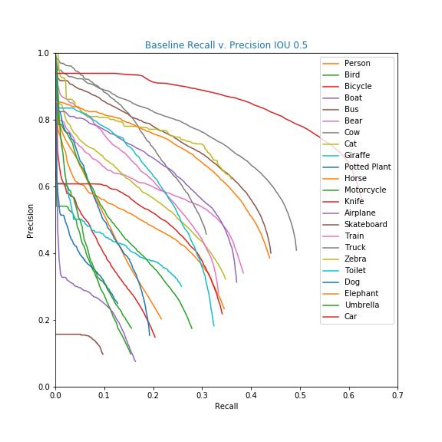
TCNN

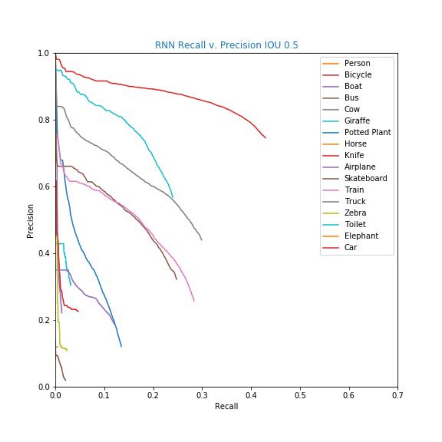


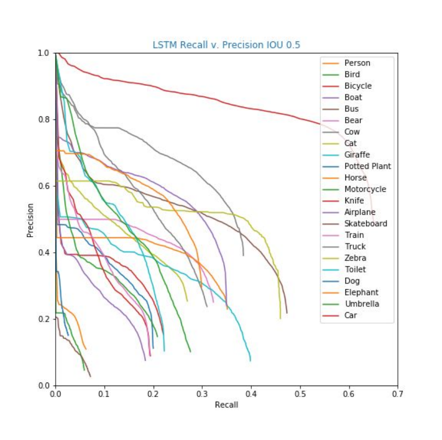
TCNN Average Precision
Object
Zebra
Umbrella
Truck
Train
Toilet
Skateboard
Potted Plant
Person
Motorcycle
Knife
Horse
Giraffe
Elephant
Dog
Cow
Cat
Car
Bus
Boat
Bird
Bicycle
Bear
Airplane
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Average Precision

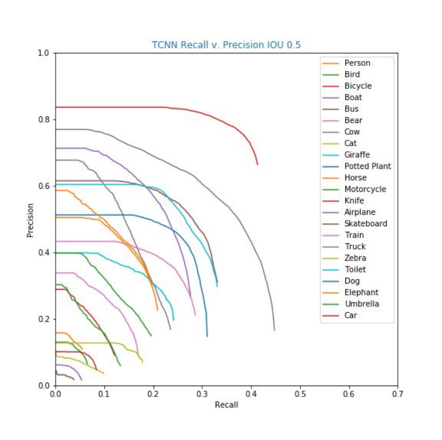
Comparative Results

- All recurrent models did worse than the baseline
- Recurrent models were harder to train
- LSTM was the best recurrent model
- TCNN had the least hyperparameter tuning and testing









Results

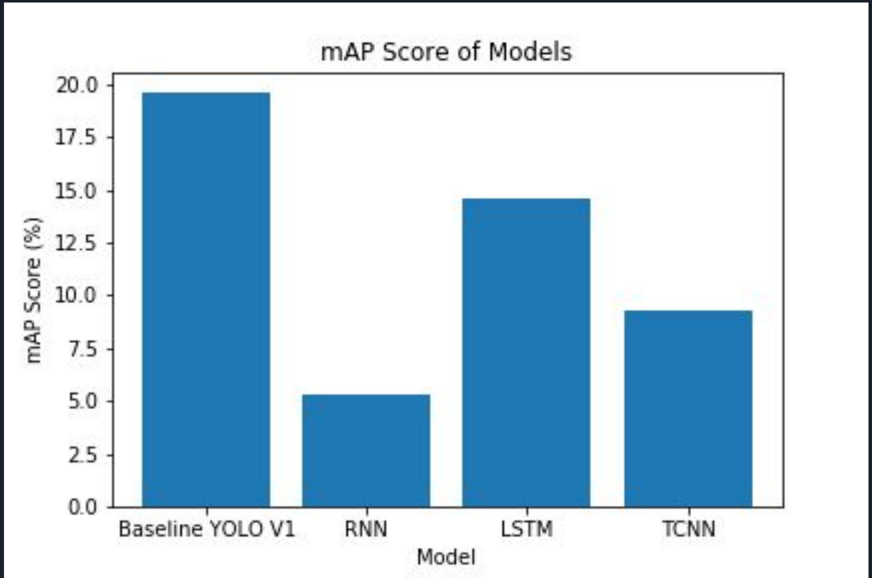
IOU 0.5 mAP Scores:

Baseline YOLO: 19.62%

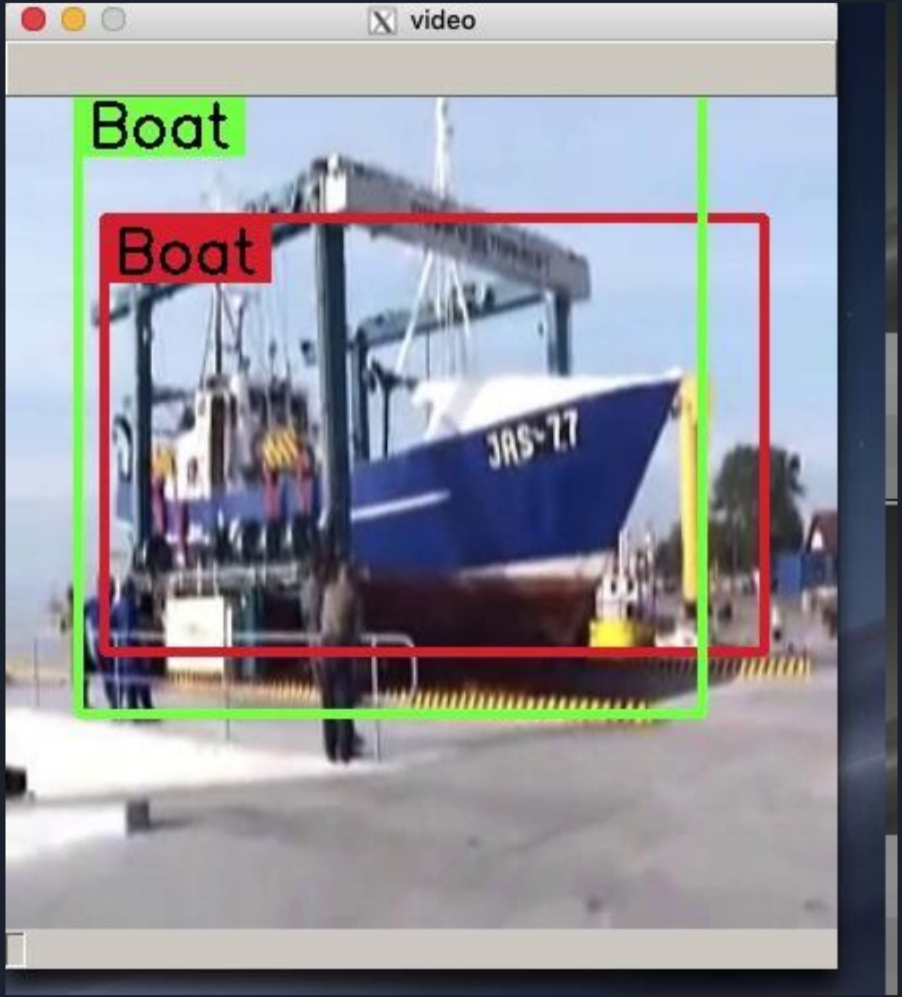
RNN: 5.34%

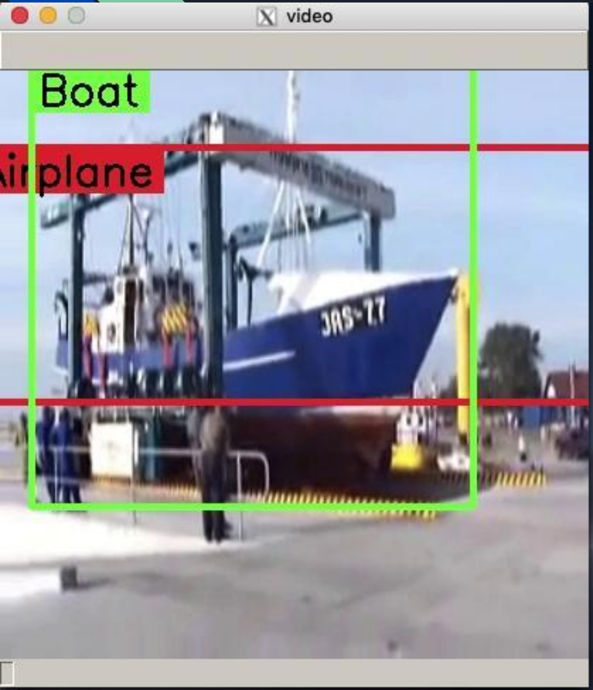
LSTM: 14.63%

TCNN: 9.29%



YOLO V1 Baseline





LSTM

TCNN

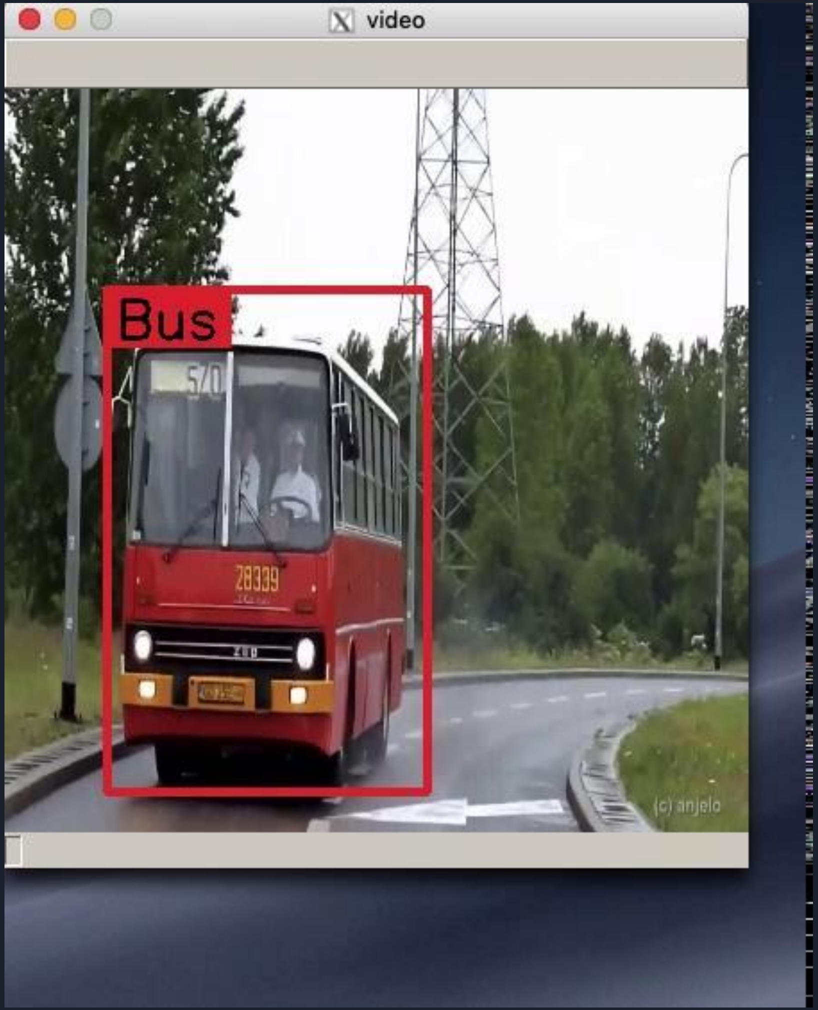
RNN

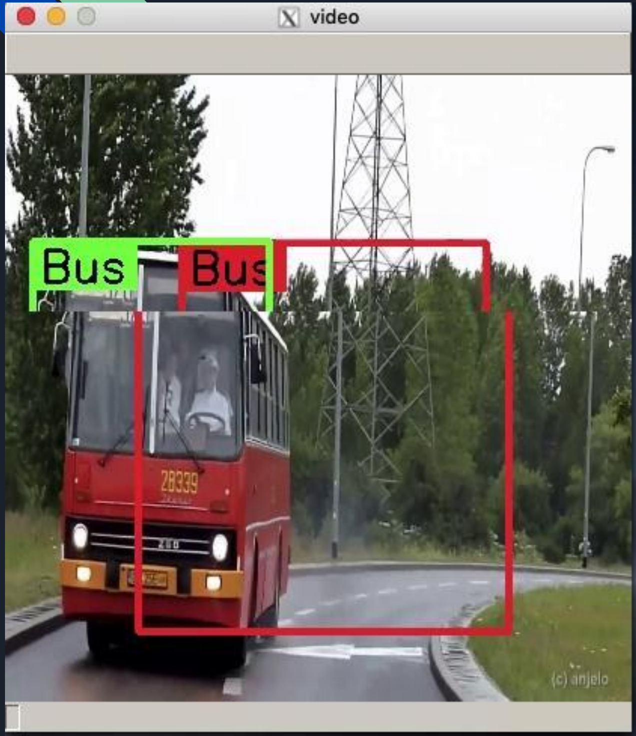
LSTM

TCNN

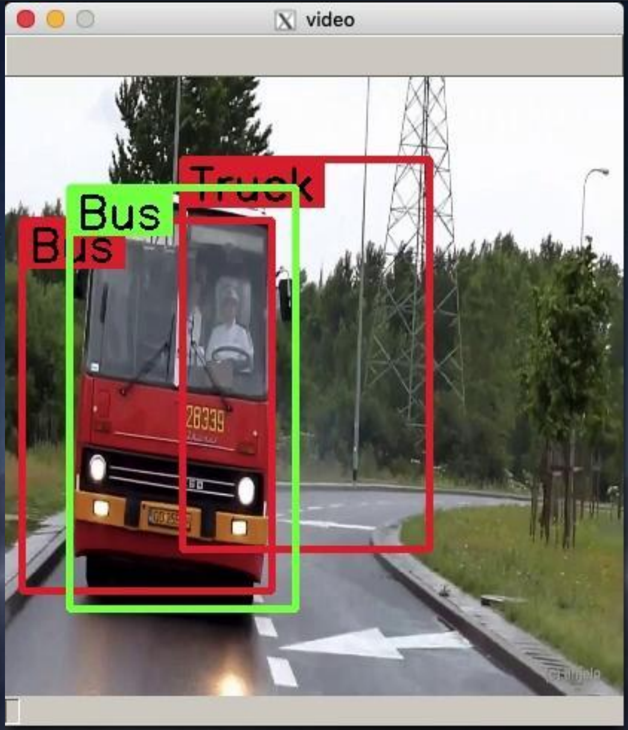
RNN

YOLO V1 Baseline

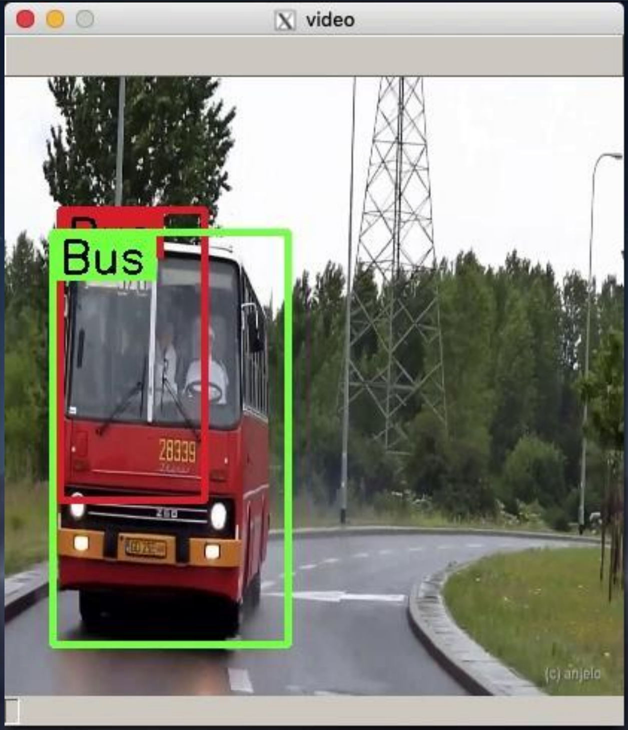




LSTM



TCNN



RNN

Conclusion

- **Recurrent models can significantly improve temporal stability for classification**
  - They do not directly improve overall mAP scores
  - LSTM models achieve the best balance between mAP scores and temporal stability
  - TCNN works well with little to no tuning
- **Models can be further improved**
  - Better convolutional network
  - Further hyperparameter tuning
  - Different combinations of recurrent layers

Challenges

- Temporal models require immense amount of memory leading to out-of-memory errors on GPU
- Unbalanced dataset, almost half the videos are people
- Inaccurate dataset, bounding boxes propagated by classical tracking algorithms aren't always accurate
  - Sometimes bounding boxes disappear during video
  - Almost every video only has one object labeled
- Training time
- Difficulty debugging neural networks
- Overfitting
- Hyperparameter tuning is difficult

Future Work

- Run RNN and LSTM over all features in a sequence
- Use random sequence length inputs while training recurrent models
- Grid Search over all parameters in validation to find optimal inferencing parameters
- Implement recurrent layers within a larger convolutional neural network like Yolo v3
- Research further the effects of messy data (heterogeneous values, missing entries, and large errors)

Questions?

Thank You

David Motta

Christian Butterfield

Northrop Grumman