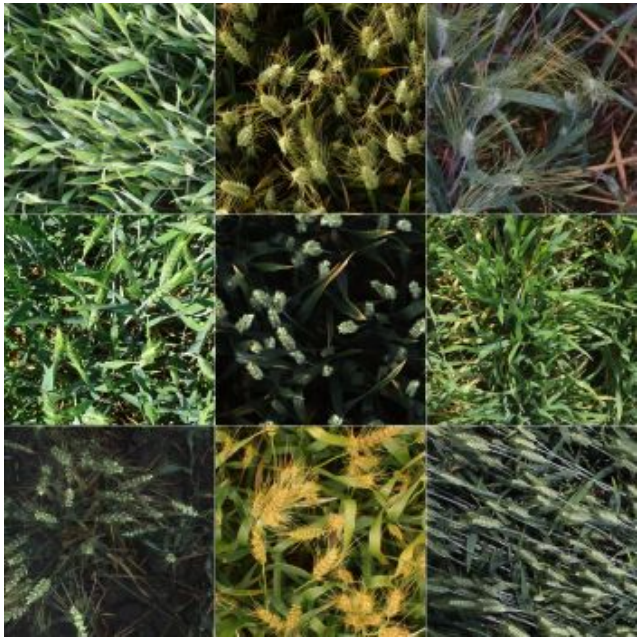
Global Wheat Detection

Albert Alonso
Alina Sode
Dina Rapp
Michael Haahr

All members contributed the same to the project.

Project Introduction

- Kaggle competition Global Wheat Detection
- Motivation: crop optimization
- High variance - global variance
- Max 6 h GPU run time



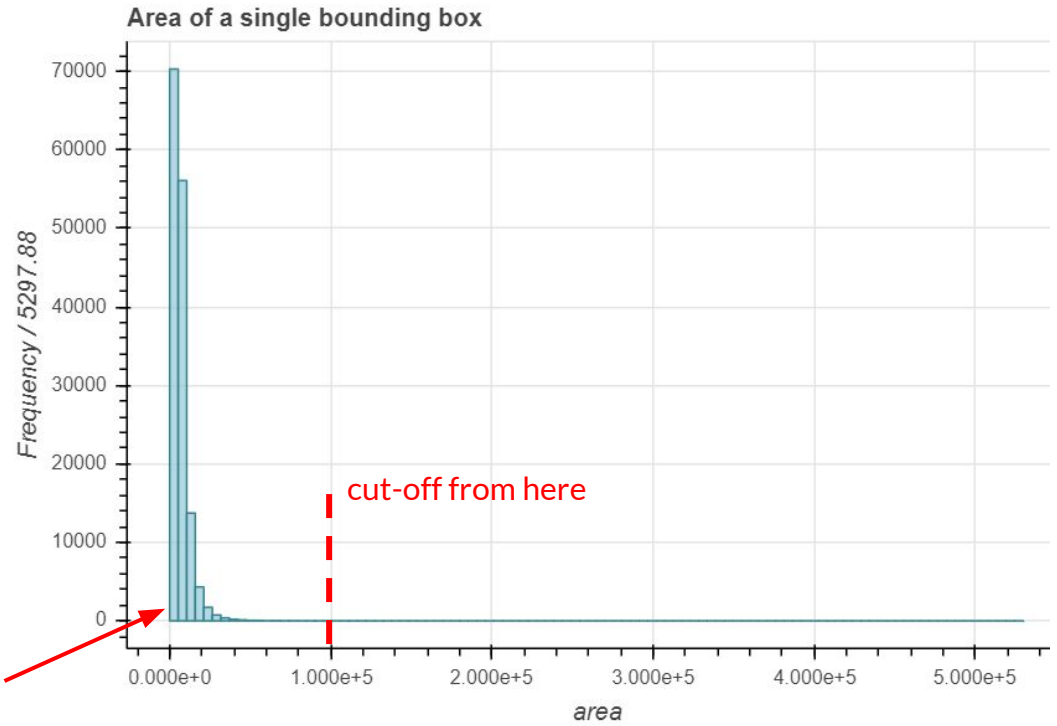
Goal: ML model capable of locating heads on a wide variety of data, without bias

Bounding Boxes

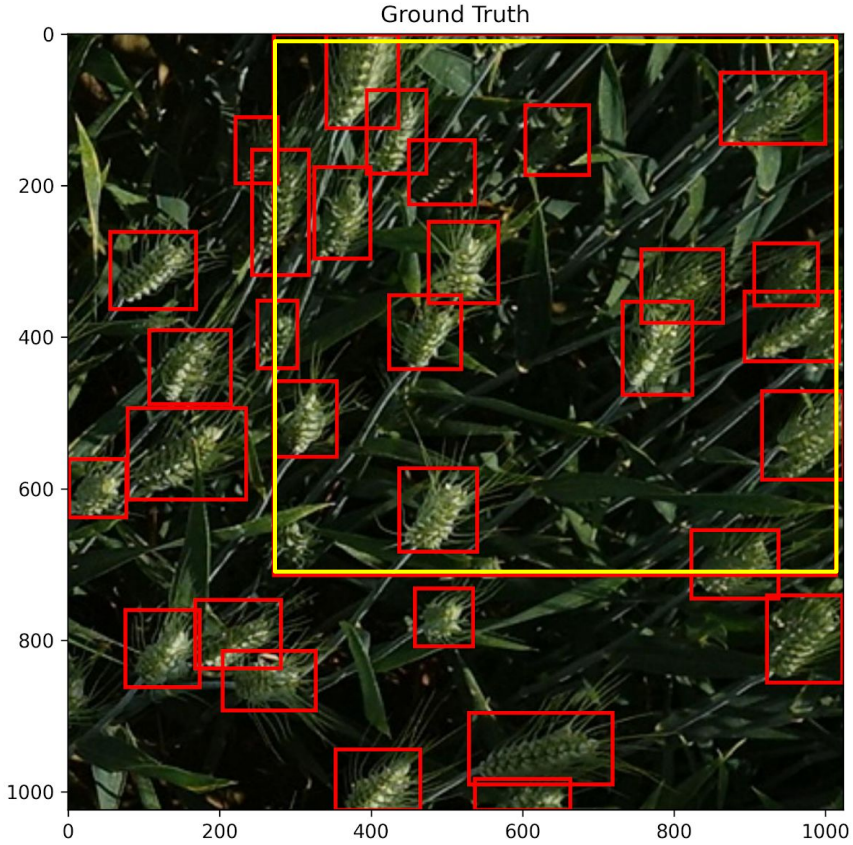
- Imaginary box around objects
- Coordinates of borders

Preprocessing: “Outlier” Removal

- Found mistakes in the ground truths
- Need for removal of “worst” / most obvious mistakes
- Skewed distribution with clear outliers of enormous bboxes



Examples of mistakes in the ground truth



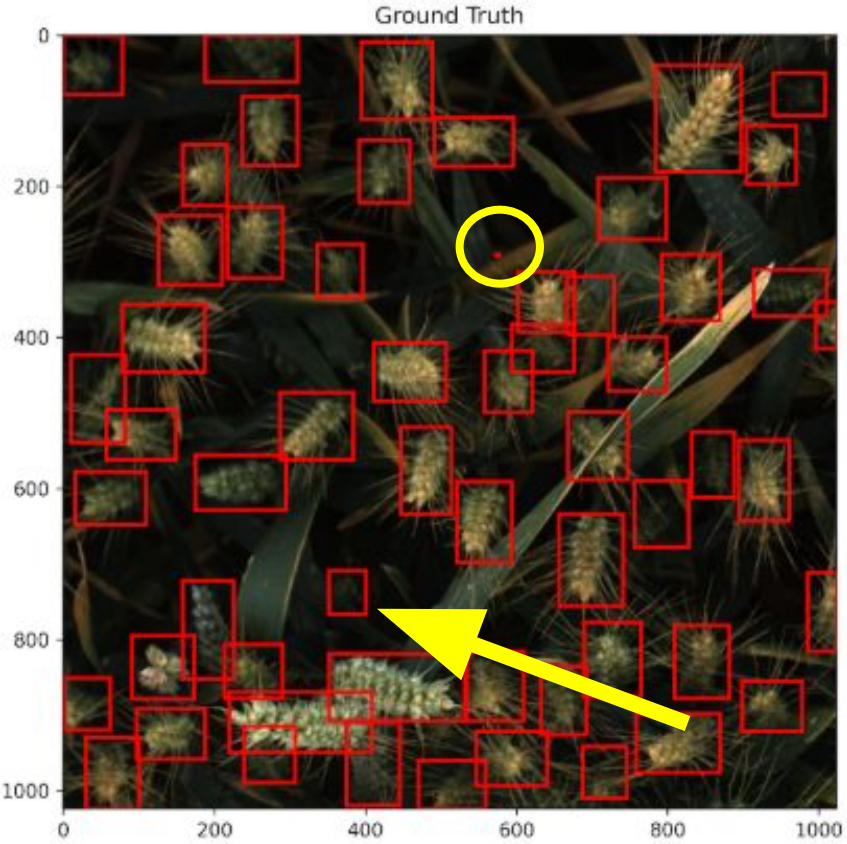
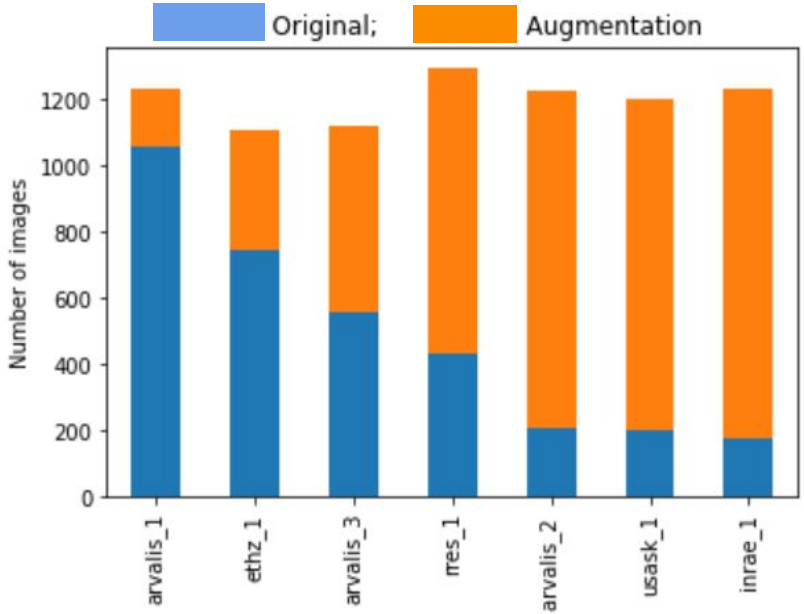
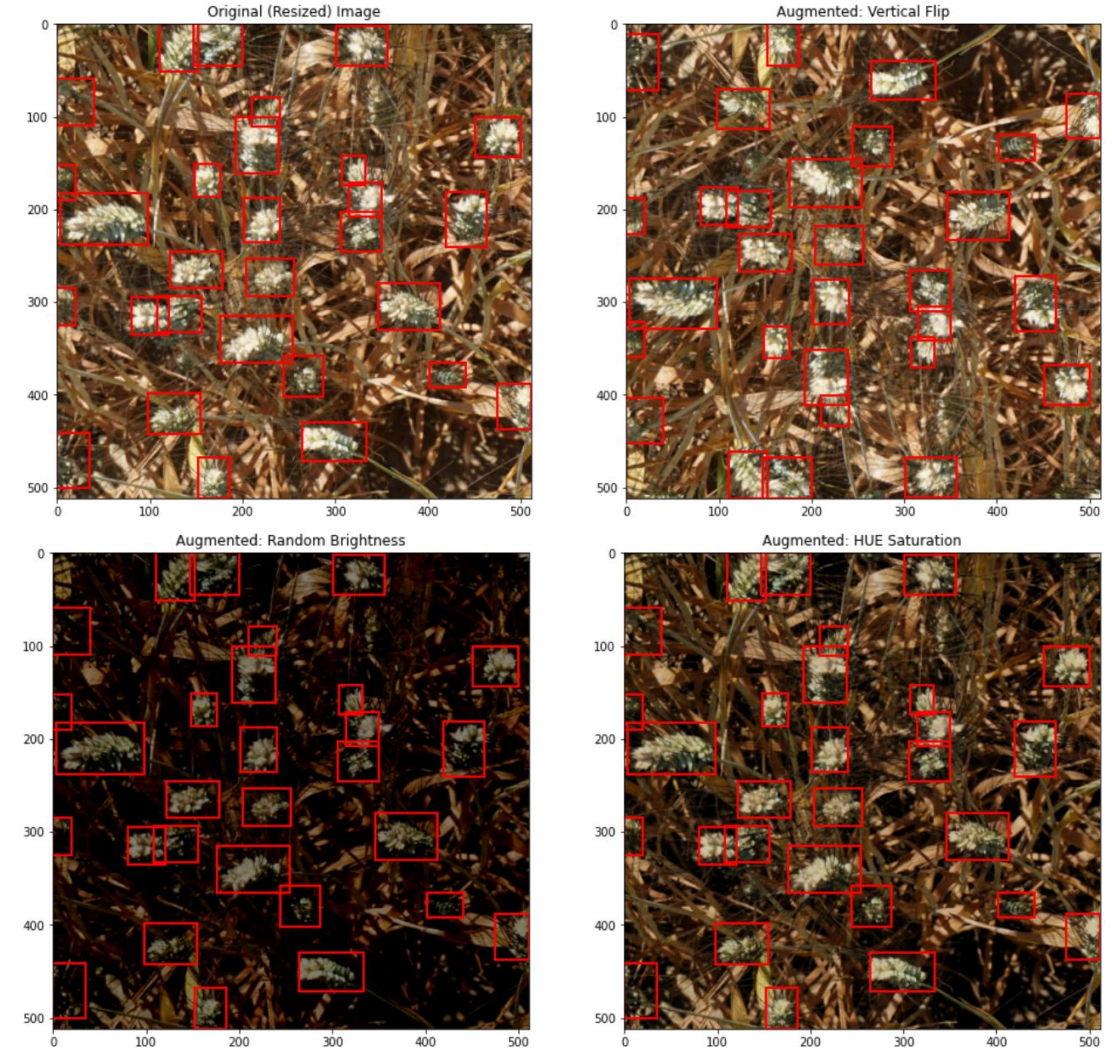


Image Augmentations

- Balance training data
- Increasing the size & variety of training set without acquiring new images
- Resizing all images



- By duplicating images with some variations;
  - Vertical Flip (p=60%)
  - Brightness (p=60%)
  - Hue Saturation Value (p=60%)



Methods

Goal: ML model capable of locating heads on a wide variety of data, without bias

	Time	Accuracy
Faster RCNN	Low	Higher
SSD	Fast	High
YOLO	Faster	Low

Kaggle

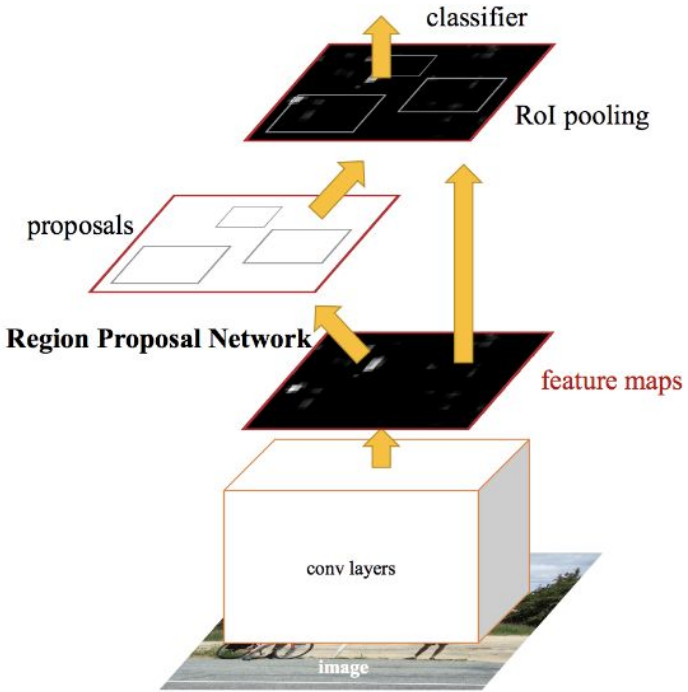
Real time detection

Information from https://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/

What is Faster than RCNN?

Faster RCNN

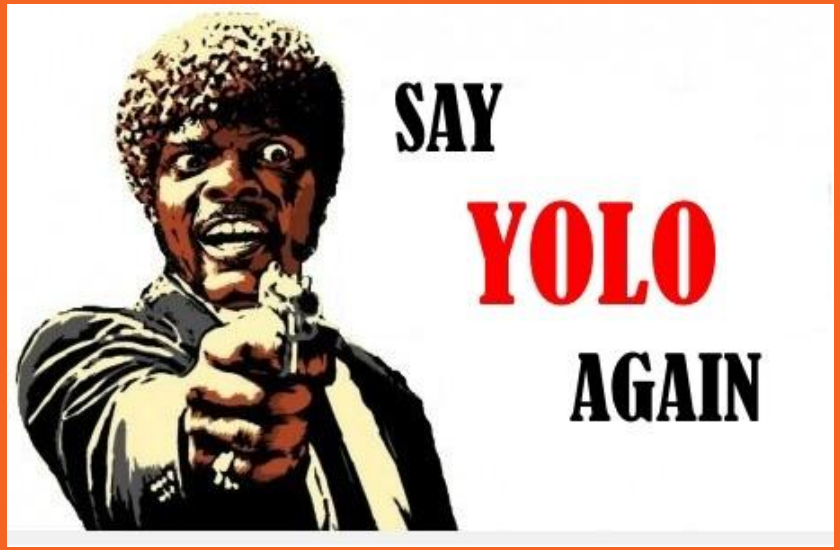
- Pre-trained CNN
- RPN - anchors (Get object proposals)
- RoI Pooling (Get feature maps)
- RCNN (Classification, adjust bboxes)



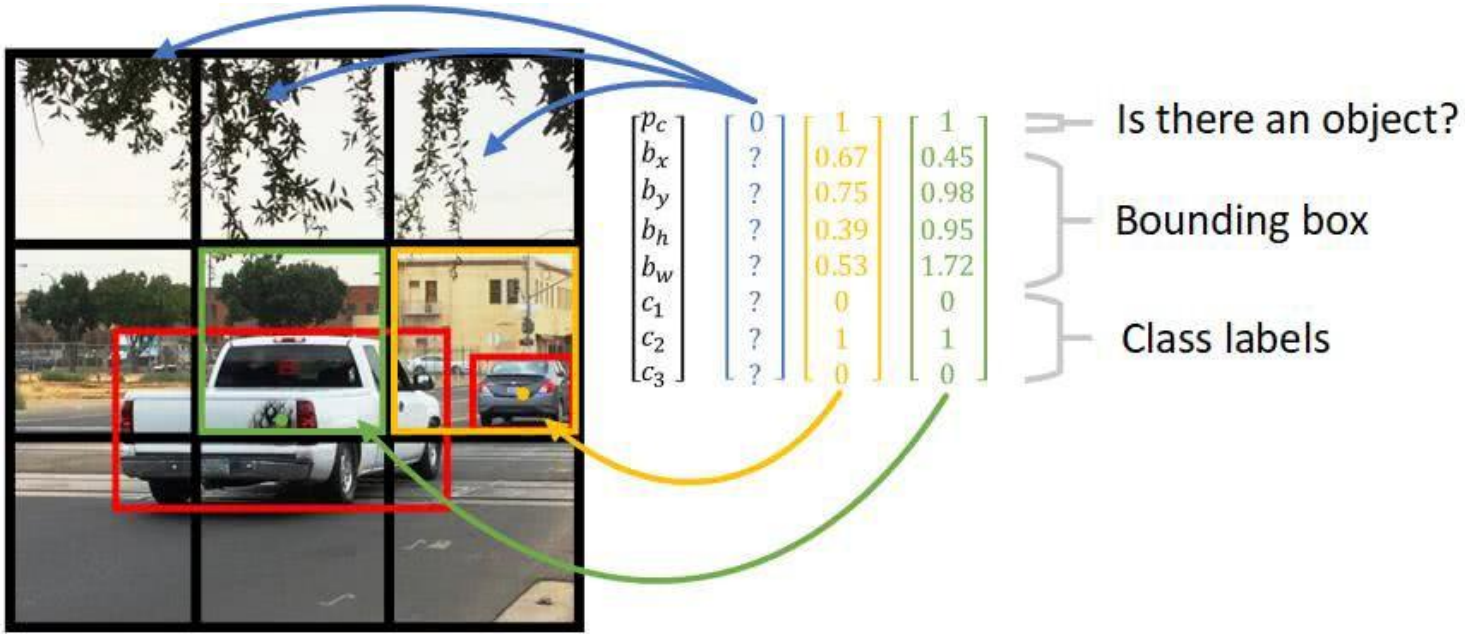
Faster RCNN implementation

- Pytorch Implementation
- Pretrained on COCO dataset
- Backbone ResNet

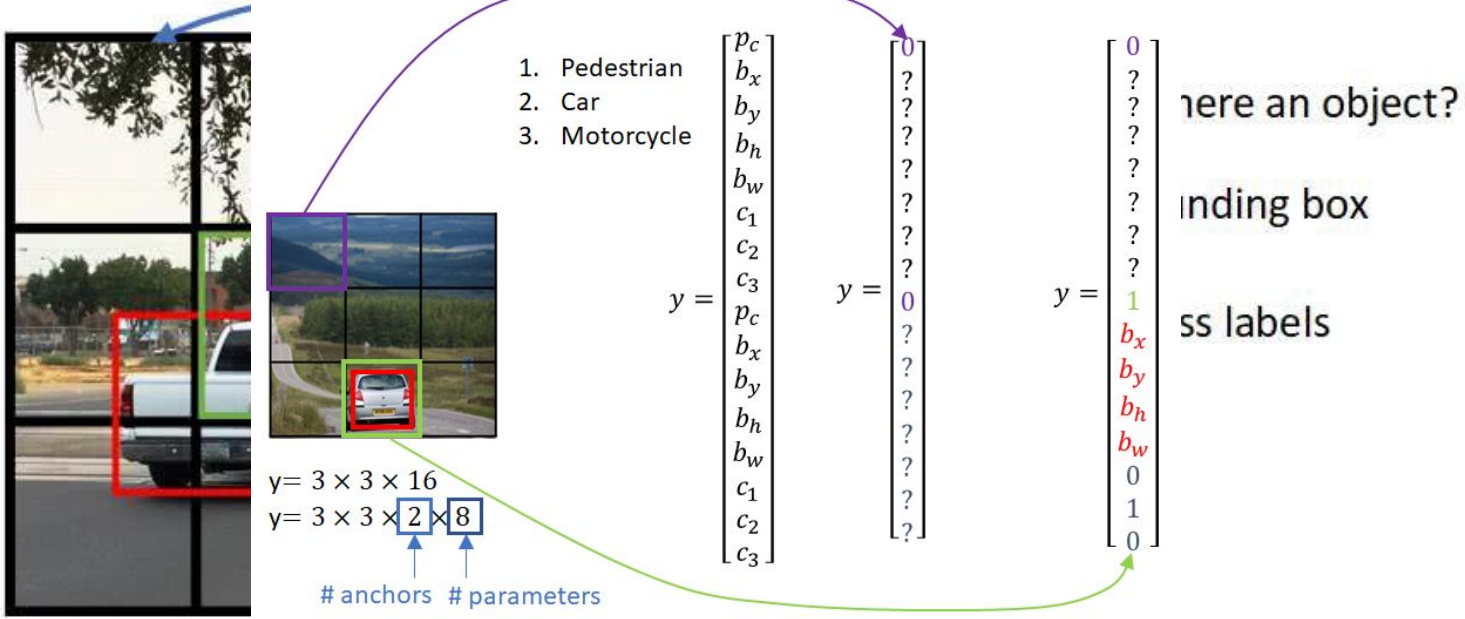
Hyperparameter	Value
Anchors	Default CoCo
Learning rate	0.01
Weight decay	0.001
Optimizer	SDG


SAY
YOLO
AGAIN

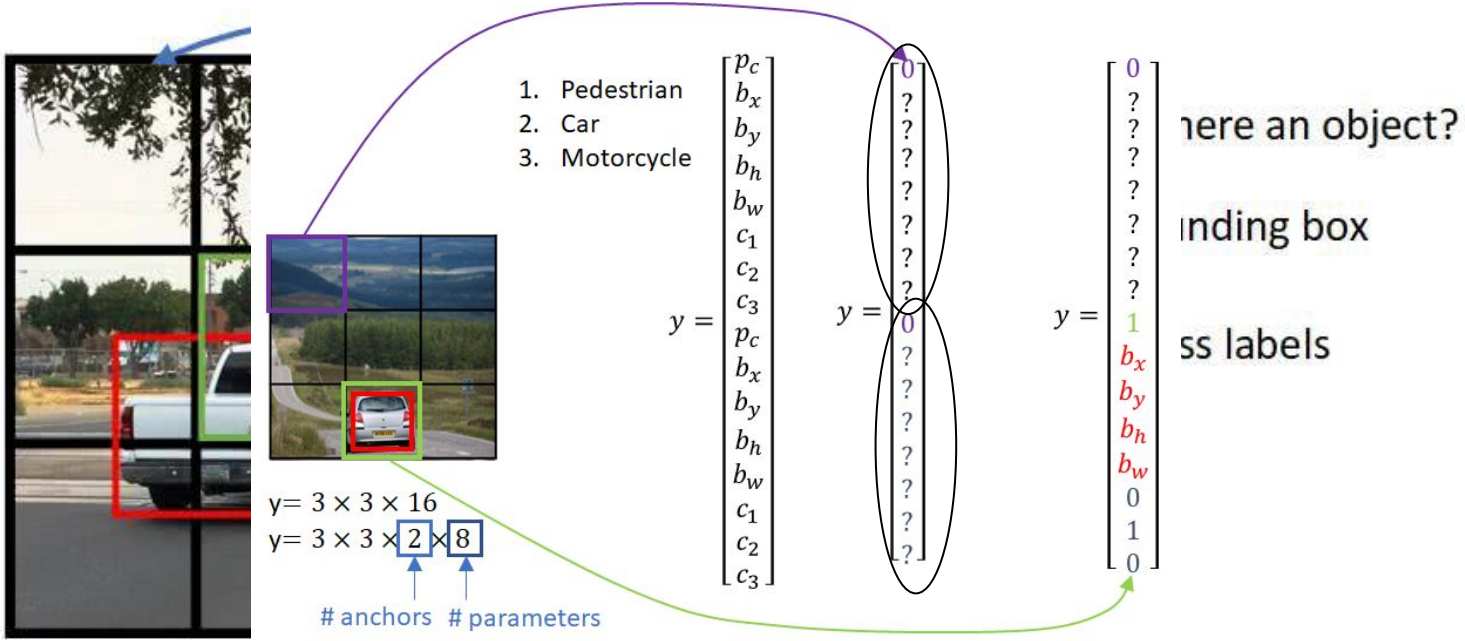
YOLO: Real-Time Object Detection



YOLO: Real-Time Object Detection



YOLO: Real-Time Object Detection



YOLO implementation

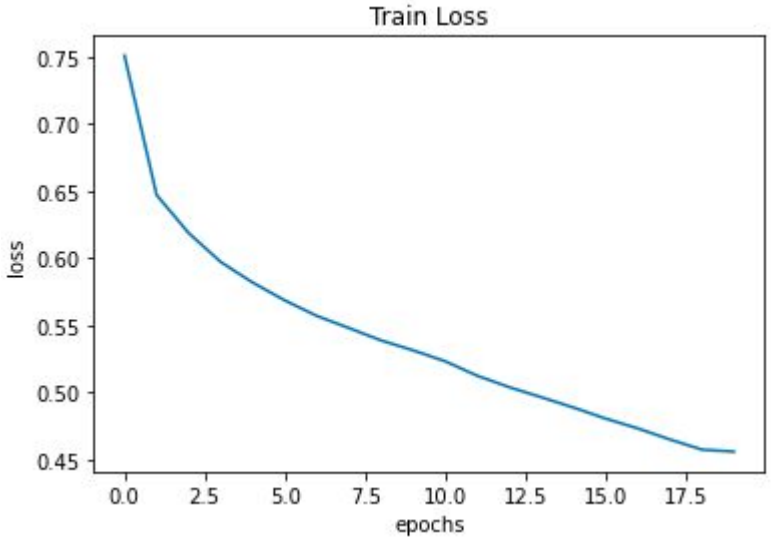
	Tiny Yolo	Full Yolo
Anchors	6	9
Convolutions	13 (+2 Yolo + 13 Extra)	75 (+2 Yolo + 31 Extra)
Model Size	33.8 MB	237 MB
FPS	200	35

Hyperparameter	Value
Anchors	Default (CoCo)
Learning rate	0.001
Decay	0.0005
Optimizer	Adam

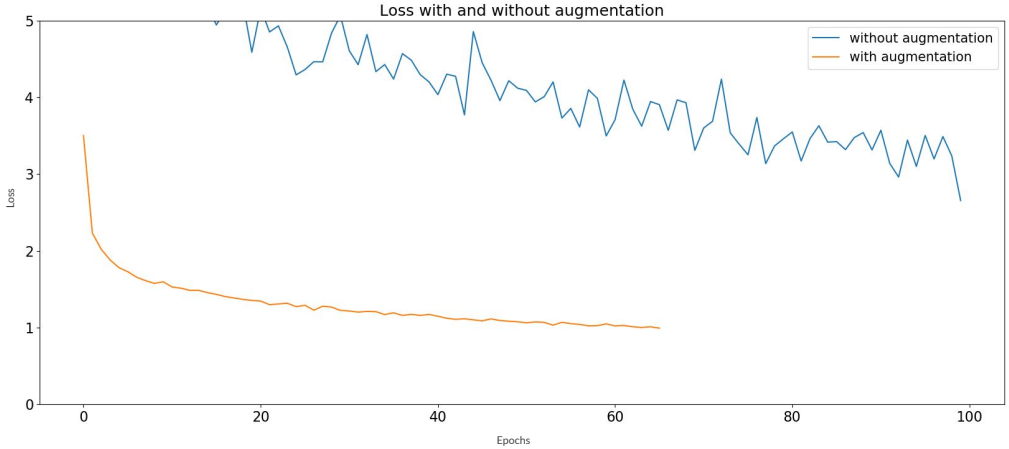
Performance

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*).$$

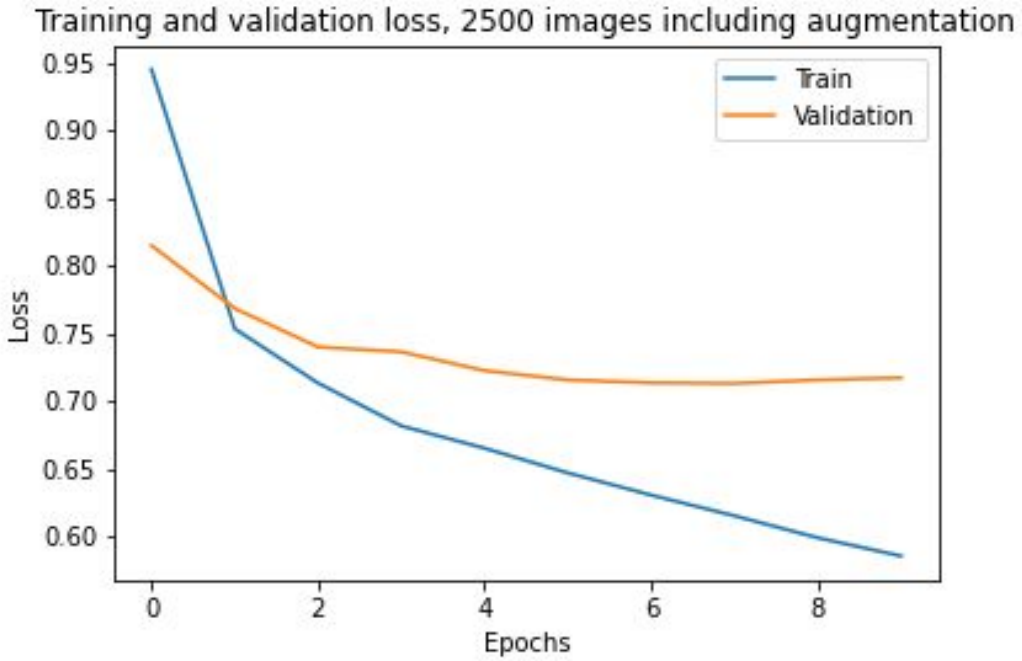
Faster RCNN



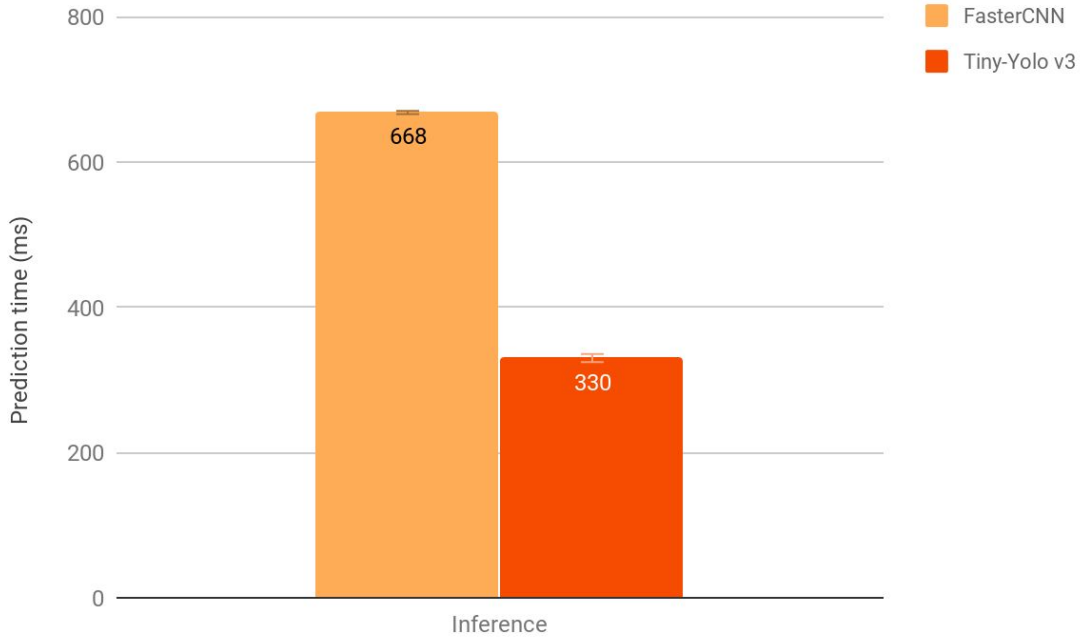
Tiny Yolo



Overfitting

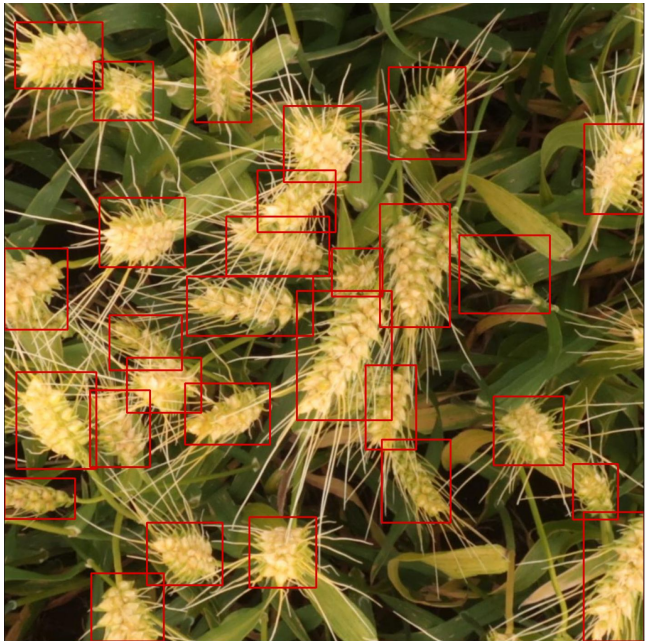


Time Comparison



Evaluation

Faster RCNN

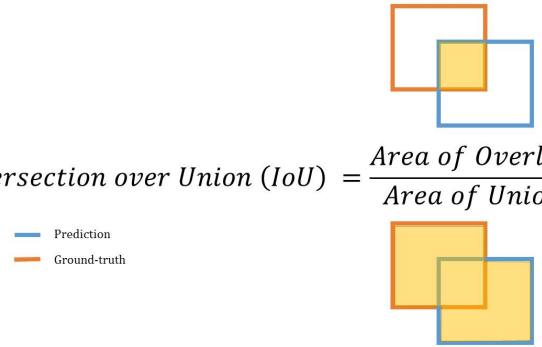


YOLO

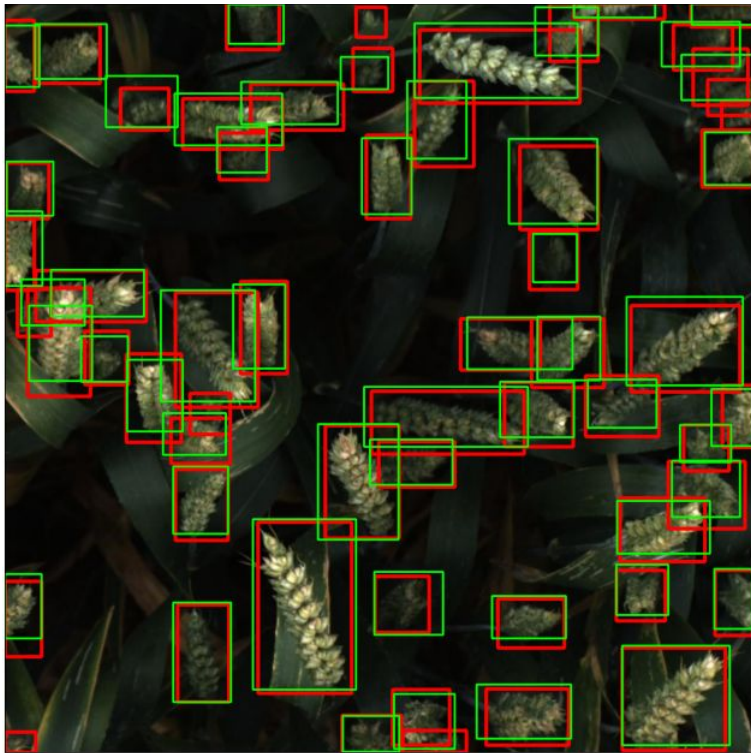
Kaggle Score	0.66	0.47

Intersection over Union (IoU)

$$Intersection\ over\ Union\ (IoU) = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$



#	Team Name	Notebook	Team Members	Score	Entries	Last
1	tereka			0.7579	94	4h
2	Anton Sakharov			0.7486	80	17h
3	Soonhwan Kwon			0.7477	25	4d
4	s_shohei			0.7463	50	1d
5	Day & Night			0.7462	55	3d



Summary

- Faster is better, but Yolo is faster
- Augmentation really helps Yolo
- Getting a high score is almost impossible

The Competition continues.... until 4th August.

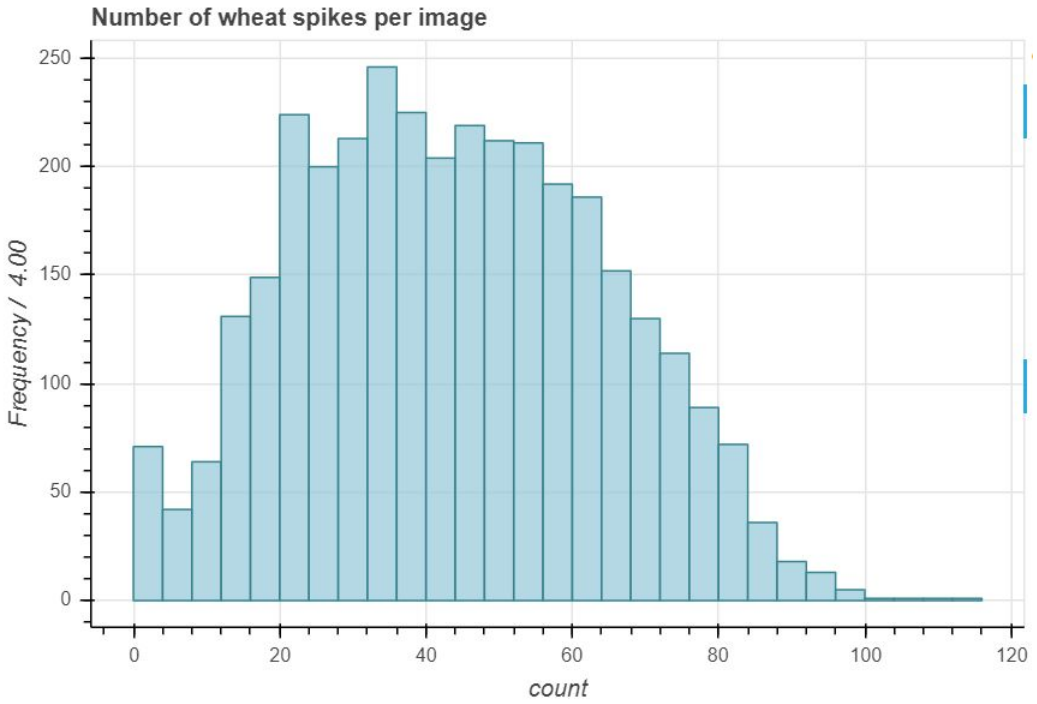
Thank you for your attention!

References

- J. Redmon, S. K. Divala, R. B Girshick, Ali Farhadi, You Only Look Once: Unified, Real-Time Object Detection
  https://arxiv.org/pdf/1506.02640.pdf
- J. Redmon, A. Farhadi, YOVOv3: An incremental Improvement
  https://arxiv.org/pdf/1804.02767.pdf
- Official Kaggle Homepage for Global Wheat Detection
  http://www.global-wheat.com/2020-challenge/
- Pytorch starter - FasterRCNN Train
  https://www.kaggle.com/pestipeti/pytorch-starter-fasterrcnn-train
- Repository with FasterCNN implementation
  https://github.com/AlbertAlonso/BDA-Global-Wheat-Detection-
- Repository with Yolo implementation
  https://github.com/Ximtecs/PyTorch-YOLOv3_WheatDetection

BBox Counts

Distribution over bounding boxes per single training images before preprocessing

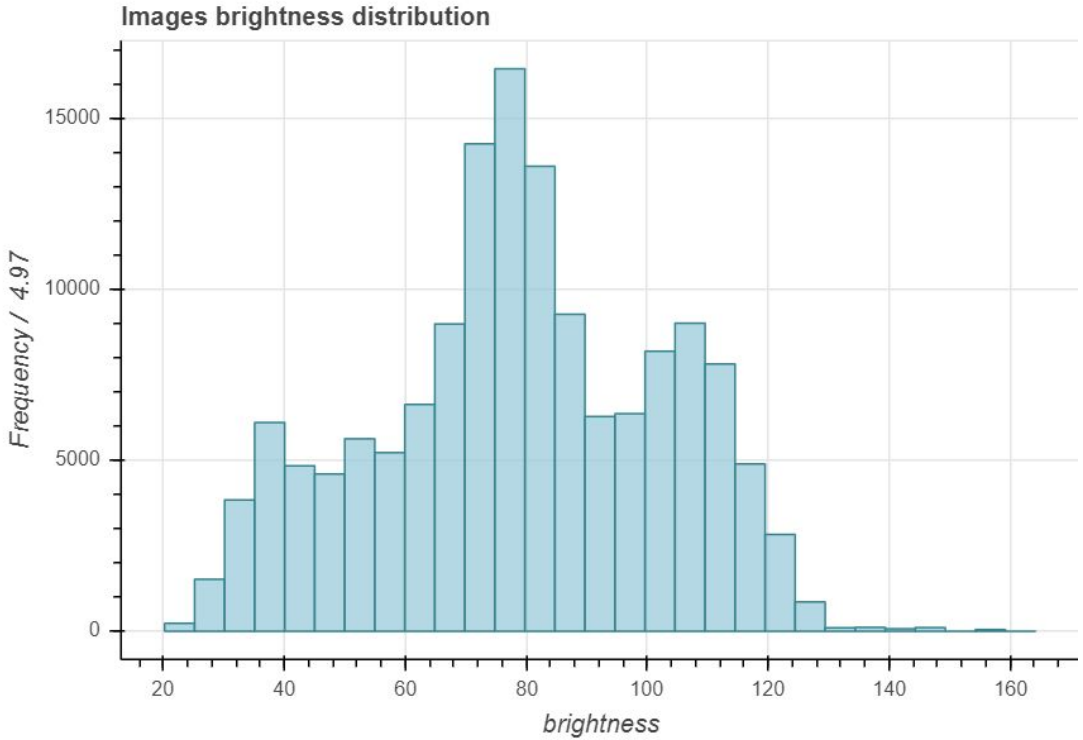


Reasons Behind Choices of Image Augmentation Techniques

- **Vertical Flip:**
  - Simply to generate more data with variety
- **Brightness:**
  - Tune bright images to darker and vice versa
- **Hue Saturation Value:**
  - Weath heads have different coloring so tune HSV

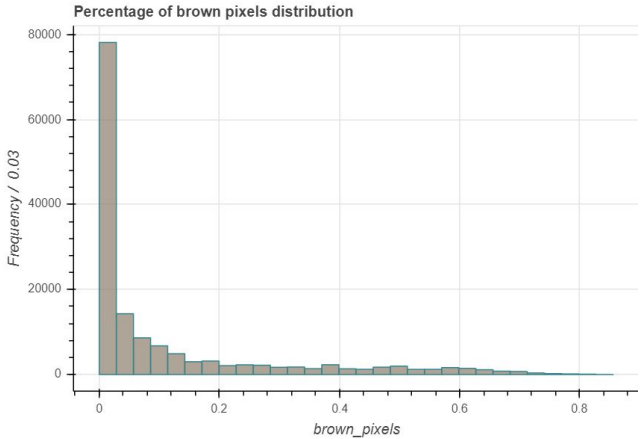
Image Brightness

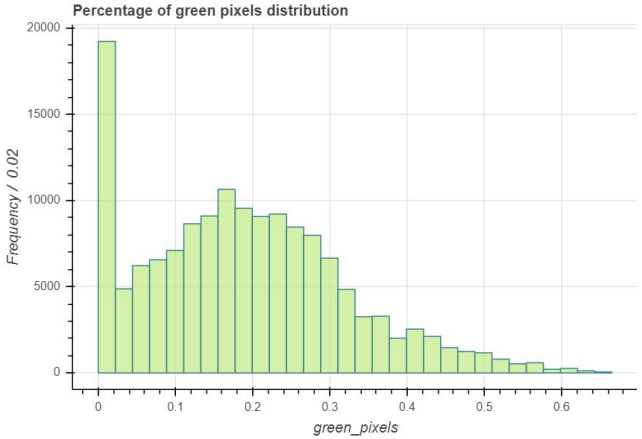
Distribution of image brightness over only the original training images

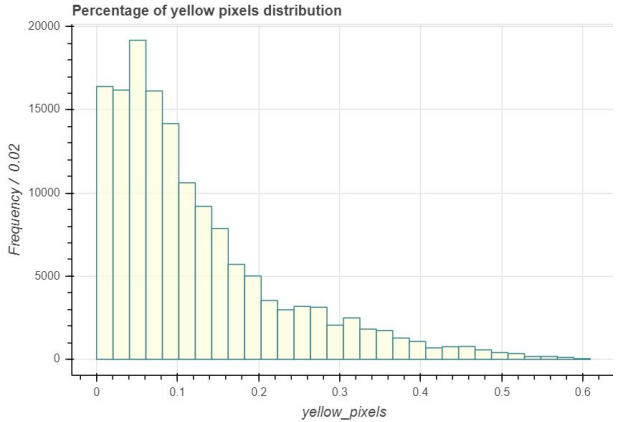


RGB Pixels

Some RGB (brown, green & yellow) distributions over original training images

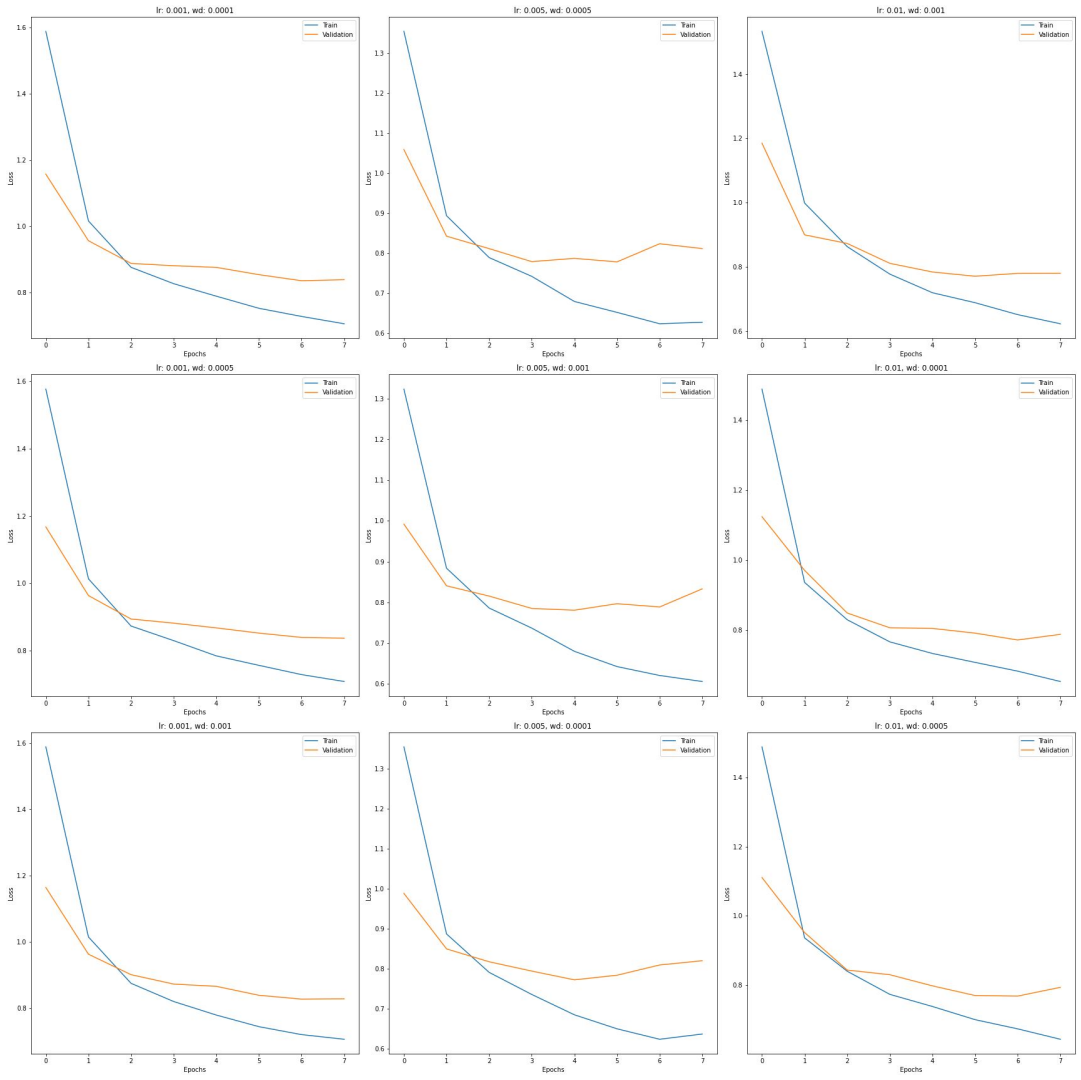






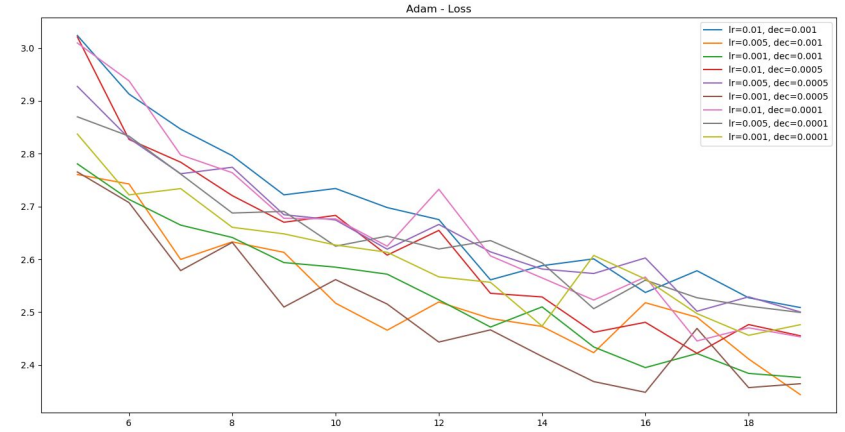
Hyperparameter optimization for Faster RCNN

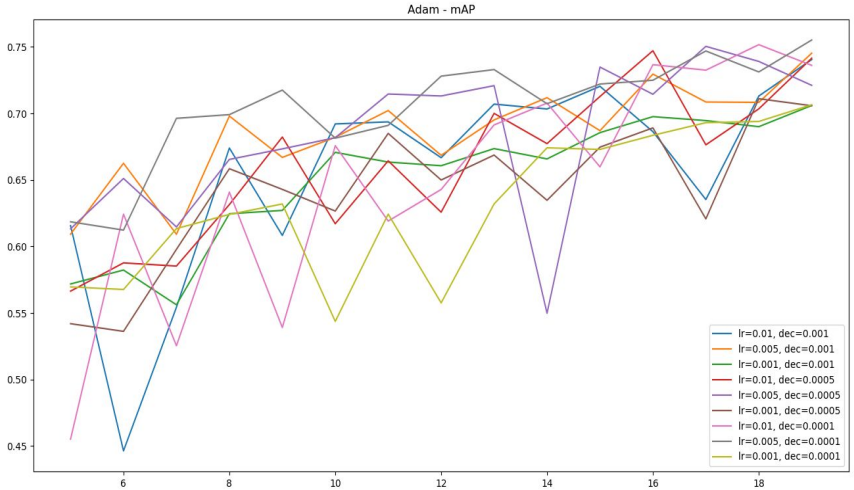
This is done with a subsample of 300 images over 8 epochs.



Hyperparameter optimization for YOLO - Adam optimizer

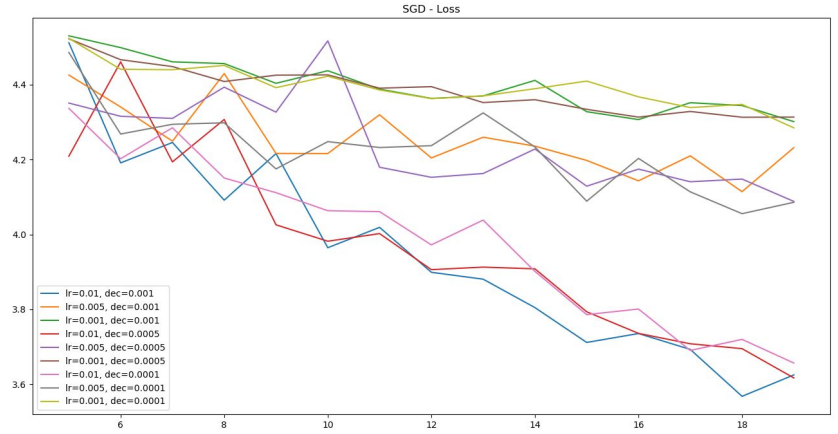
This is done with a subsample of 300 images over 20 epochs.

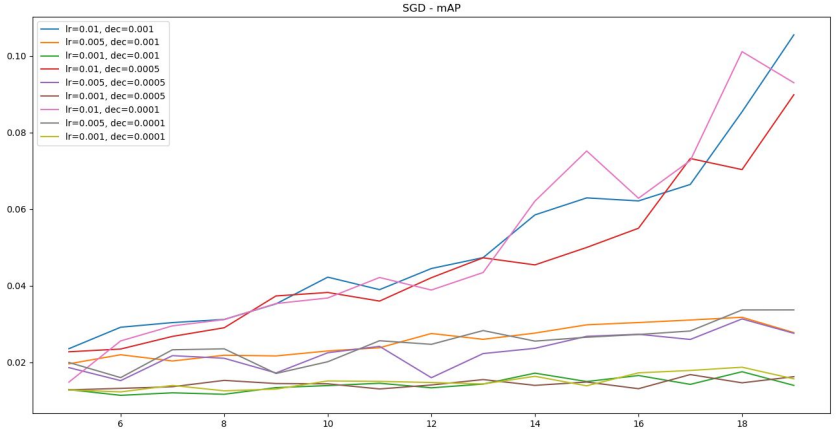




Hyperparameter optimization for YOLO - SGD optimizer

This is done with a subsample of 300 images over 20 epochs.





With and without augmentation - YOLO

Trained for 1.5h