Detection of Ball and Player Location from Frame-by-Frame Soccer Match Images

Tim Majidzadeh, Etienne Ndedi

MIDS ML 207 – Machine Learning – Final Presentation

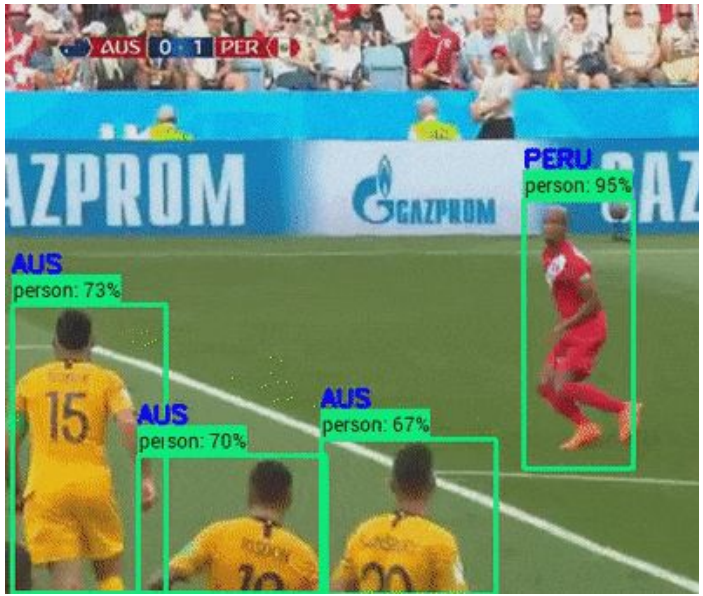
Berkeley
UNIVERSITY OF CALIFORNIA

Objective

Our **objective** is to develop a machine learning model which can identify ball location and player location with high accuracy.

Event data is of high strategic value to sports analysts; this model could help streamline the labor-intensive process of collecting data that could be used for analytics.
Importance: soccer analysts, coaches, enthusiasts.

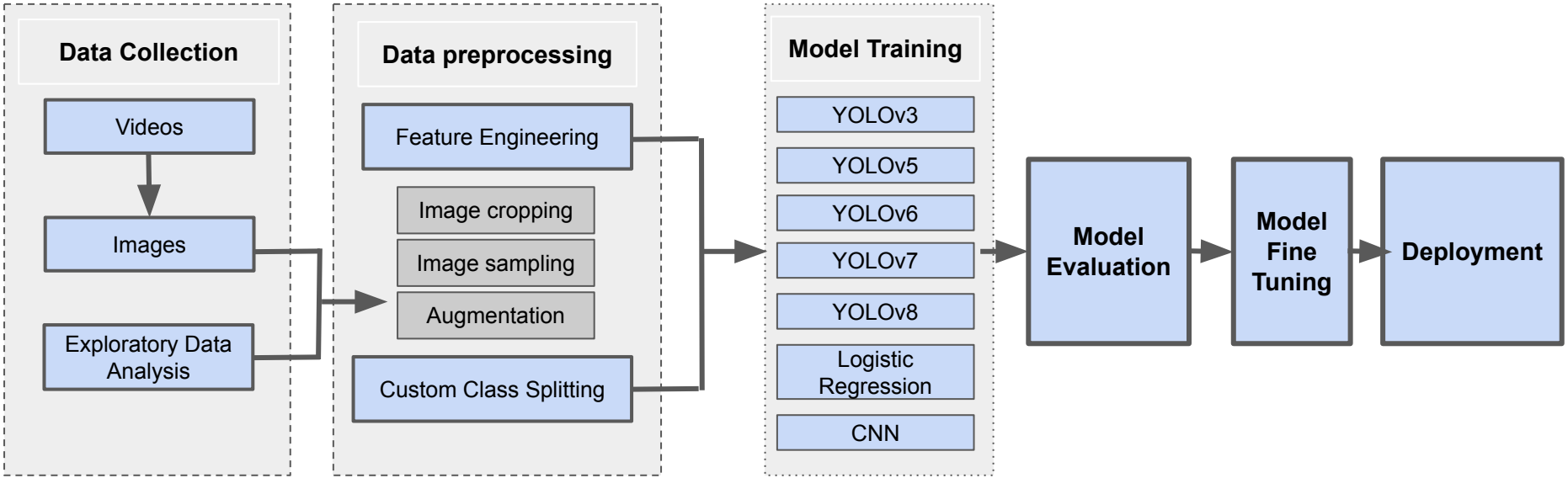
Object detection in images is the analytical framework addressed.



Approach/Methodology

For our approach, we converted videos to images that we augmented to improve the likelihood of accurate object detection, and we subsequently implemented a series of models that we evaluated based on several performance metrics.

Block Diagram



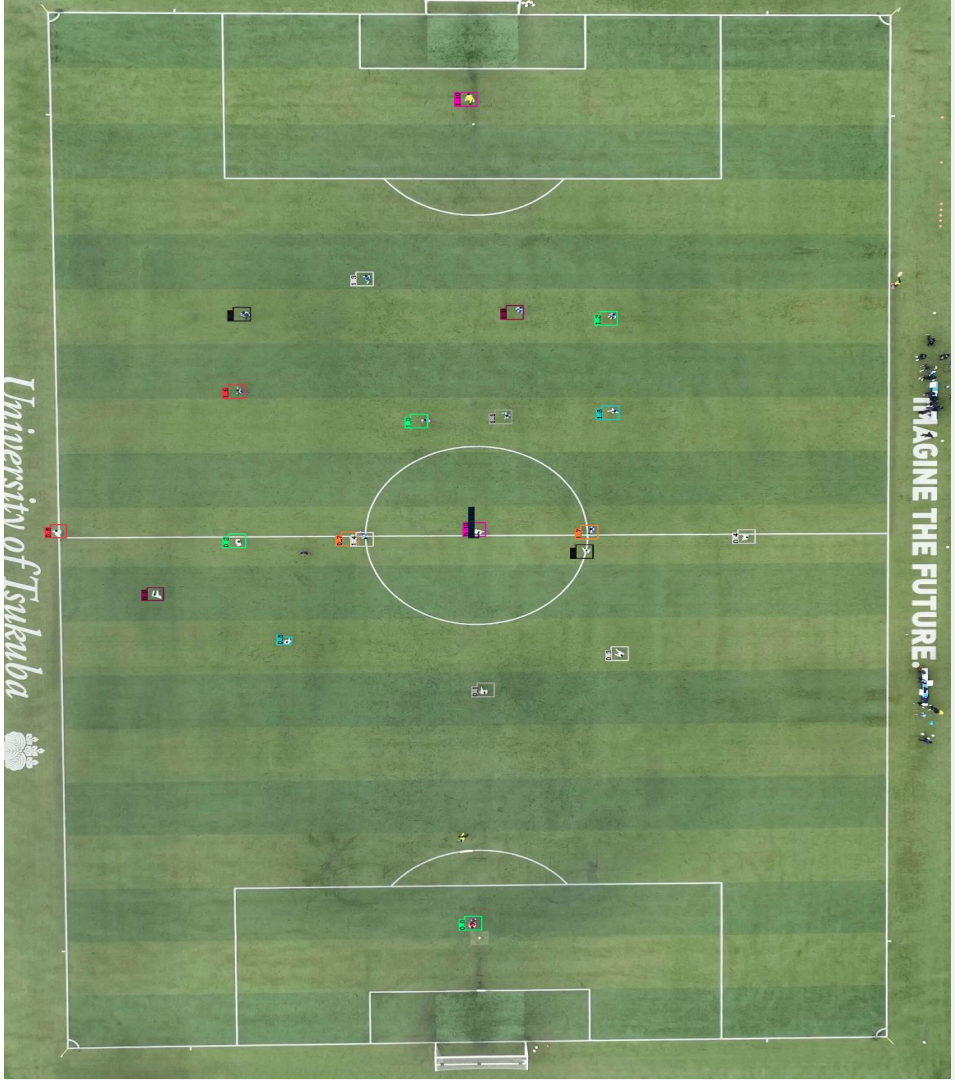
Evaluation Metrics

We used the metrics below to evaluate the performance of the trained models.

- Loss Values
  - Box Loss (cIOU)
  - Objectness Loss
  - Class Loss (Binary Cross-Entropy)
- Precision
- Recall
- Confusion Matrices
  - True Positives, True Negatives, False Positives, False Negatives
- Mean Average Precision (mAP)
  - mAP 0.5 - Uses confidence threshold at 0.5
  - mAP 0.5-0.95 - Uses range across 0.5-0.95

Data Source and Collection

- **The dataset is sourced from Kaggle.com***
- The dataset covers an hour of game footage:
  - 30 Mins top-down view
  - 30 Mins wide angle view
  - Cut to 30-second clips
  - 30 Frames per Second
  - CSV Files with frame-by-frame bounding boxes for 22 players & the ball.
- We translated the videos into images and the bounding boxes represented the labels.



*** Source:** Uchida, Ikuma, et al, "SoccerTrack", available at https://www.kaggle.com/datasets/atomscott/soccertrack.

Data Exploration - Top View

What do the Bounding Boxes of the Ball look like?

Image Size: 3840 x 2160 (w x h)	Count (Frames)	Null Values (Count)	Mean	Standard Deviation	Minimum	Maximum
X-Coordinate (Box Top Left Corner)	53,220	1,256	1,724.53	776.32	274.0	3,482
Y-Coordinate (Box Top Left Corner)	53,220	1,256	1,010.91	571.69	-3.0	2,157
Box Width (Pixels)	53,220	1,256	**14.99**	2.81	8.0	24
Box Height (Pixels)	53,220	1,256	**14.10**	2.70	3.0	24

Null values are OK (it means the ball bounced out of frame), **but negative values make no sense** and **those bounding boxes are not included**.

Bounding box **widths & heights are small** relative to the image.

Data Exploration - Wide View

What do the Bounding Boxes of the Ball look like?

Image Size: 6500 x 1000 (w x h)	Count (Frames)	Null Values (Count)	Mean	Standard Deviation	Minimum	Maximum
X-Coordinate (Box Top Left Corner)	47,322	580	3584.95	760.35	0.00	5334.00
Y-Coordinate (Box Top Left Corner)	47,322	580	586.17	118.18	136.00	992.00
Box Width (Pixels)	47,322	580	**13.77**	10.19	4.00	93.00
Box Height (Pixels)	47,322	580	**12.46**	7.25	3.00	56.00

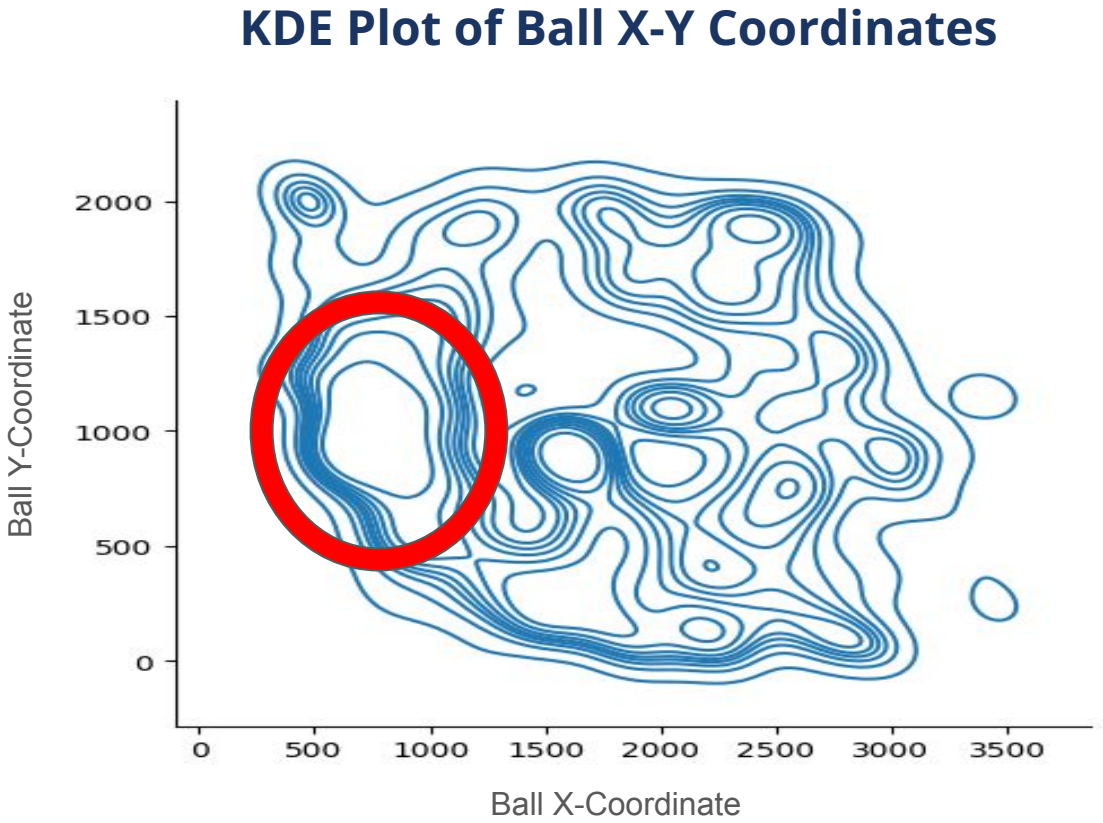
Bounding box **widths & heights are small** relative to the image.

Data Exploration

Where are the most common ball locations in the image?

A KDE Plot of ball locations shows a **bias towards the left-hand side.**

We apply a **randomized left-right flip** during augmentation to control for this.



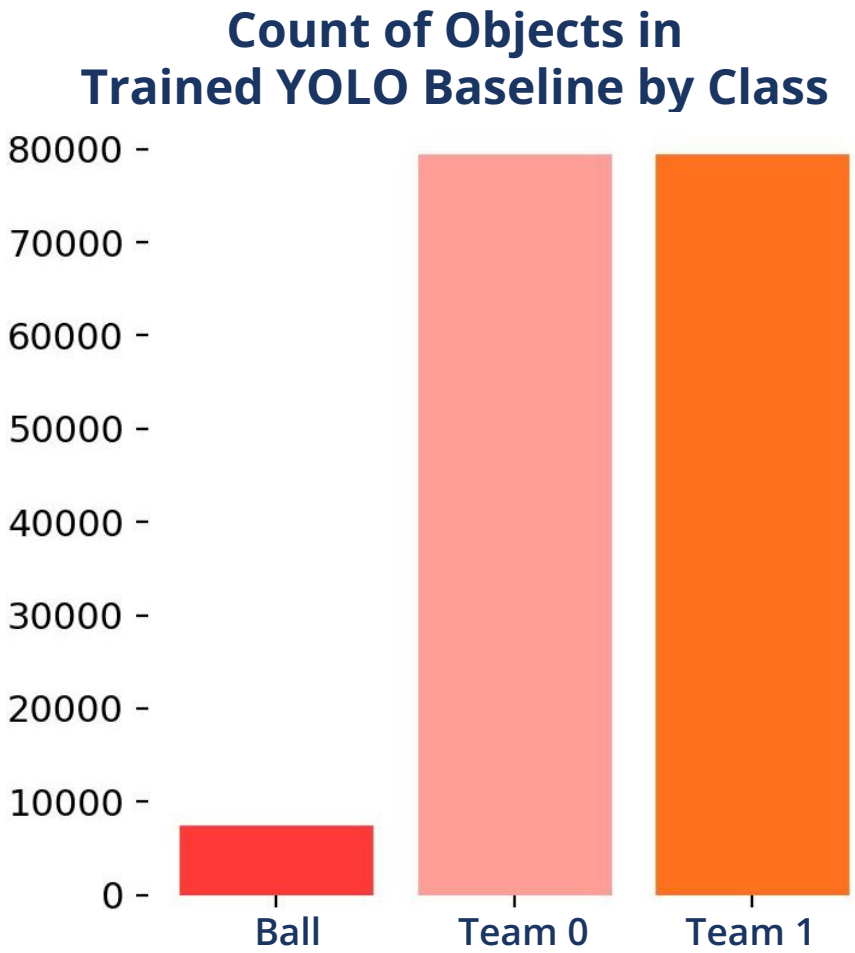
Data Exploration

Are the Object Classes Balanced?

The two teams are strongly balanced. **However, the ball is far less common** (22 players vs 1 ball)!

Baseline model **struggles to detect the ball**.

Improved models deal with this issue by cropping & over-sampling images with a ball.



Baseline Data Preparation

- ✔ **Data Collection** - Save every frame of each video as a PNG. Match & stack the labels.
- ✔ **Image Compression** - Compress images to 800p to reduce memory cost.
- ✔ **Image Augmentation** - Random flip each image horizontally. Randomly adjust hue, saturation, translation, and zoom.
- ✔ **Feature engineering** - Update the labels to match the altered images. Normalize pixel RGB inputs to (0, 1). Set to grayscale (for logistic regression only). Set bounding box coordinates to be centroids instead of top-left corner. Arrange project directory for the packages implementing YOLO.

Binary Models and Results - Simple Classifiers

Stratified Sample: Ball Somewhere in 50% of Images, Each Team in 70%.

Model	Output Type	Loss	Accuracy	Precision	Recall
Log. Regression	Binary (Ball)	1.665	.560	0.500	.179
Log. Regression	Binary (Team 0)	0.745	.719	.816	.701
Log. Regression	Binary (Team 1)	3.671	.712	.733	.989
Log. Regression	Multiclass Predictor	4.561	.197	.392	.138
CNN	Binary (Ball)	0.693	0.481	N/A	.000
CNN	Binary (Team 0)	0.586	0.700	0.716	1.000
CNN	Binary (Team 1)	0.595	0.719	0.722	1.000
CNN	Multiclass Predictor	1.549	0.468	0.202	0.450

Note: The simple classifiers are run on the images after cropping to 800p x 800p squares.

Baseline Models and Hyperparameters

- **Baseline Models:**
  - **Ultralytics, Meituan Vision AI, or Wong Kin-Yiu's** implementation of **YOLOv3, v5, v6, v7, and v8** using the preset "**nano**" or **"tiny"** backbones.
  - **~10,400 input images** with 80-10-10 train-val-test split, with **batch size 64**.
- **Hyperparameters**
  - **Adam** optimizer.
  - Initial LR = **0.01**, Final LR = **0.01 * 0.01 = 0.0001**.
  - Bounding Box Loss: **Complete Intersection over Union** ("CIoU").
  - Classification Loss: **Binary Cross Entropy**.
  - **5** Training Epochs.

Baseline Models and Results - YOLO Object Detection

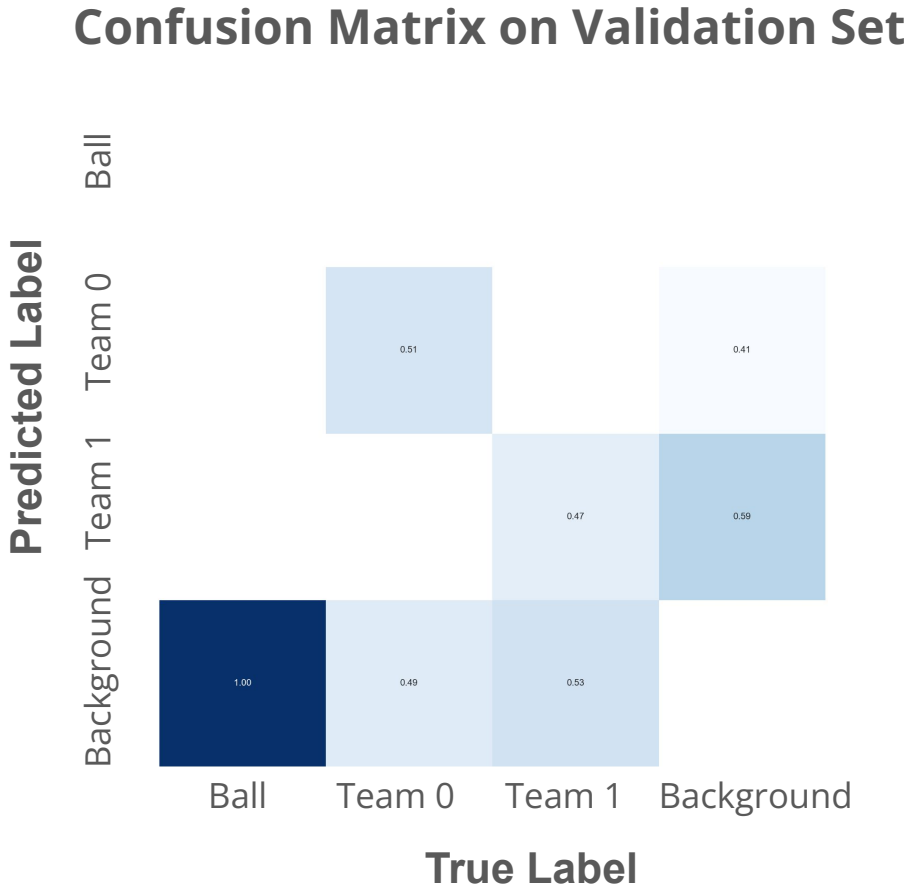
Model	Author	Box Loss	Obj. Loss	Class. Loss	Precision	Recall	mAP 0.5	mAP 0.5-0.95
YOLOv3	Ultralytics	0.116	0.247	.0063	.586	.264	.206	.059
YOLOv5	Ultralytics	0.041	0.024	.0010	.343	.202	.178	.054
YOLOv6	Meituan	3.397	2.560	.9375	.177	.044	.035	.013
YOLOv7	Wong Kin-Yiu	0.019	0.019	.0008	.447	.313	.264	.082
YOLOv8	Ultralytics	2.156	0.821	1.061	.401	.267	.298	.116

Baseline YOLOv7 Results - Validation

The model's **performance on human players is fair** - correctly predicting ~50% for both teams.

Players are usually confused with the background & not with the other team.

Biggest weakness: the baseline ***never*** detects the ball!

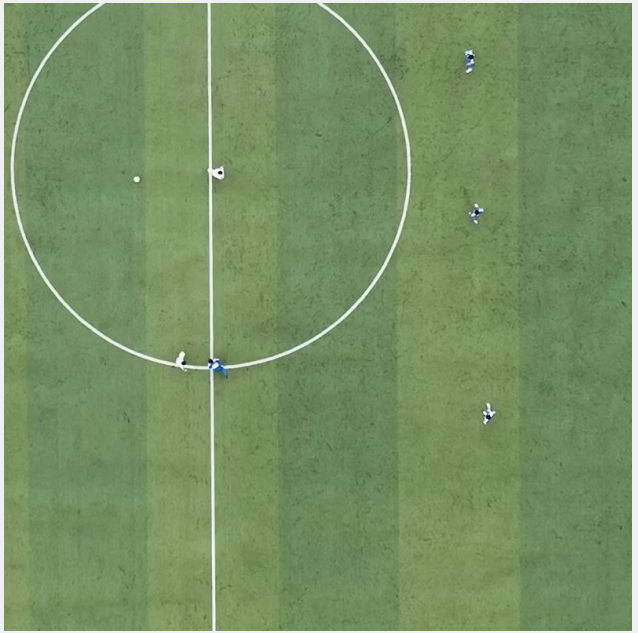


Improving The Models: Cropping & Zooming

Original Example

Can *you* see the ball?

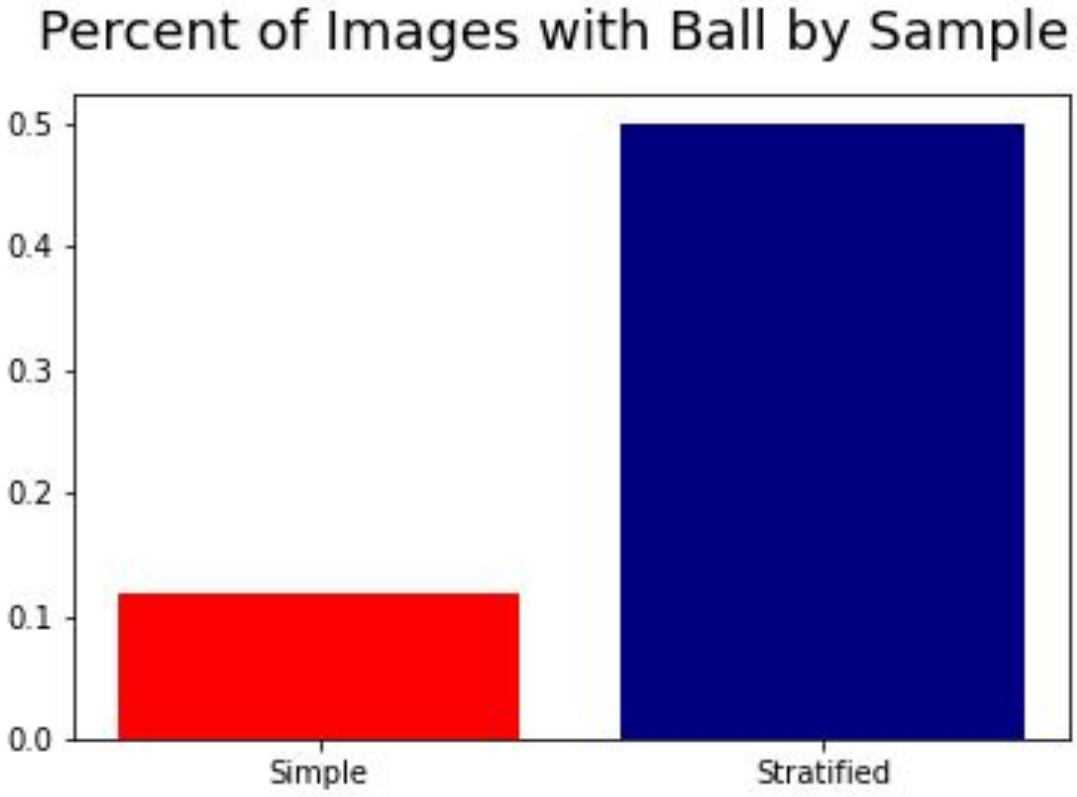
After Cropping and Zooming



How about now?

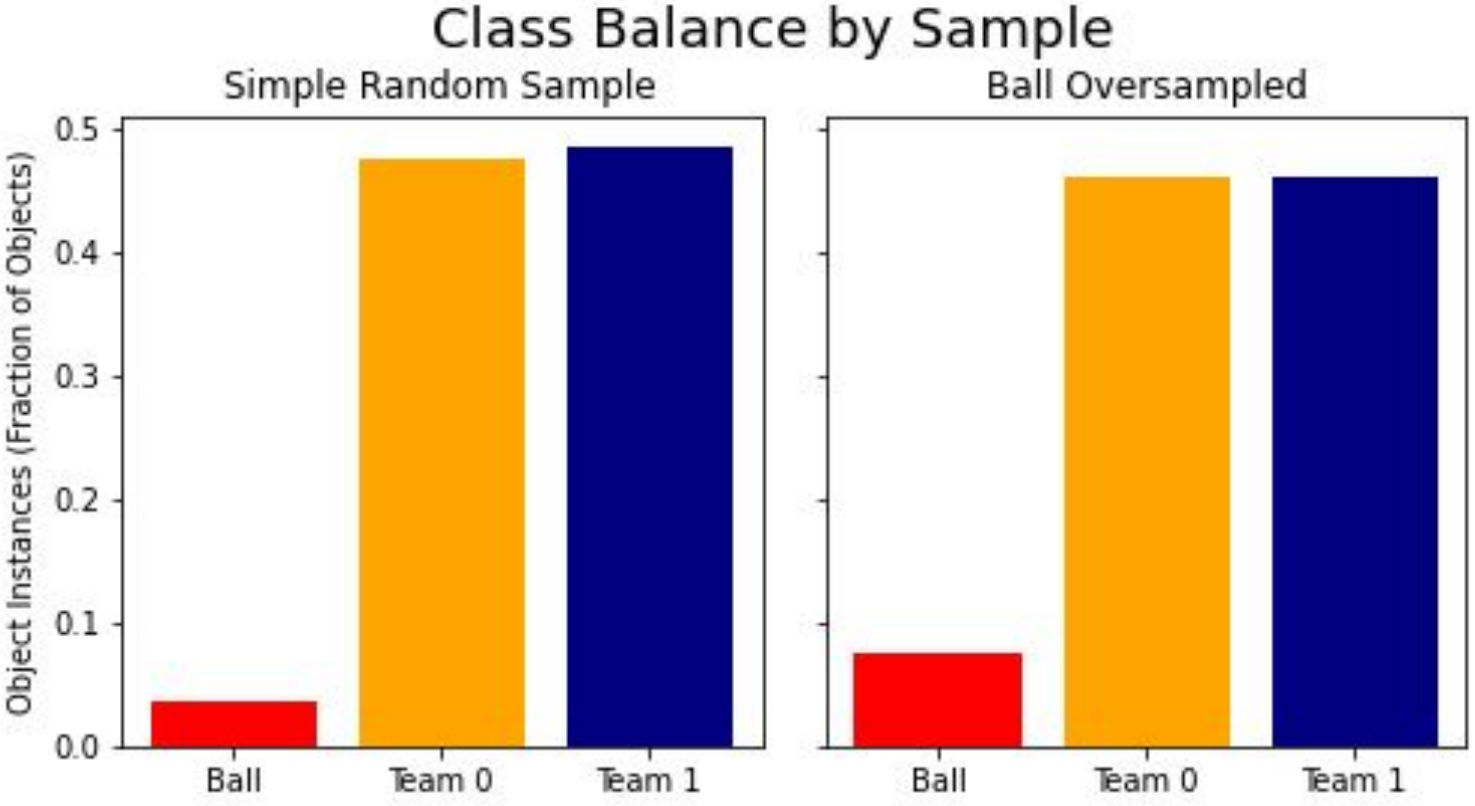
Improving The Models: Over-Sampling the Ball

50% Ball & Players; 30% Players Only; 20% No Objects



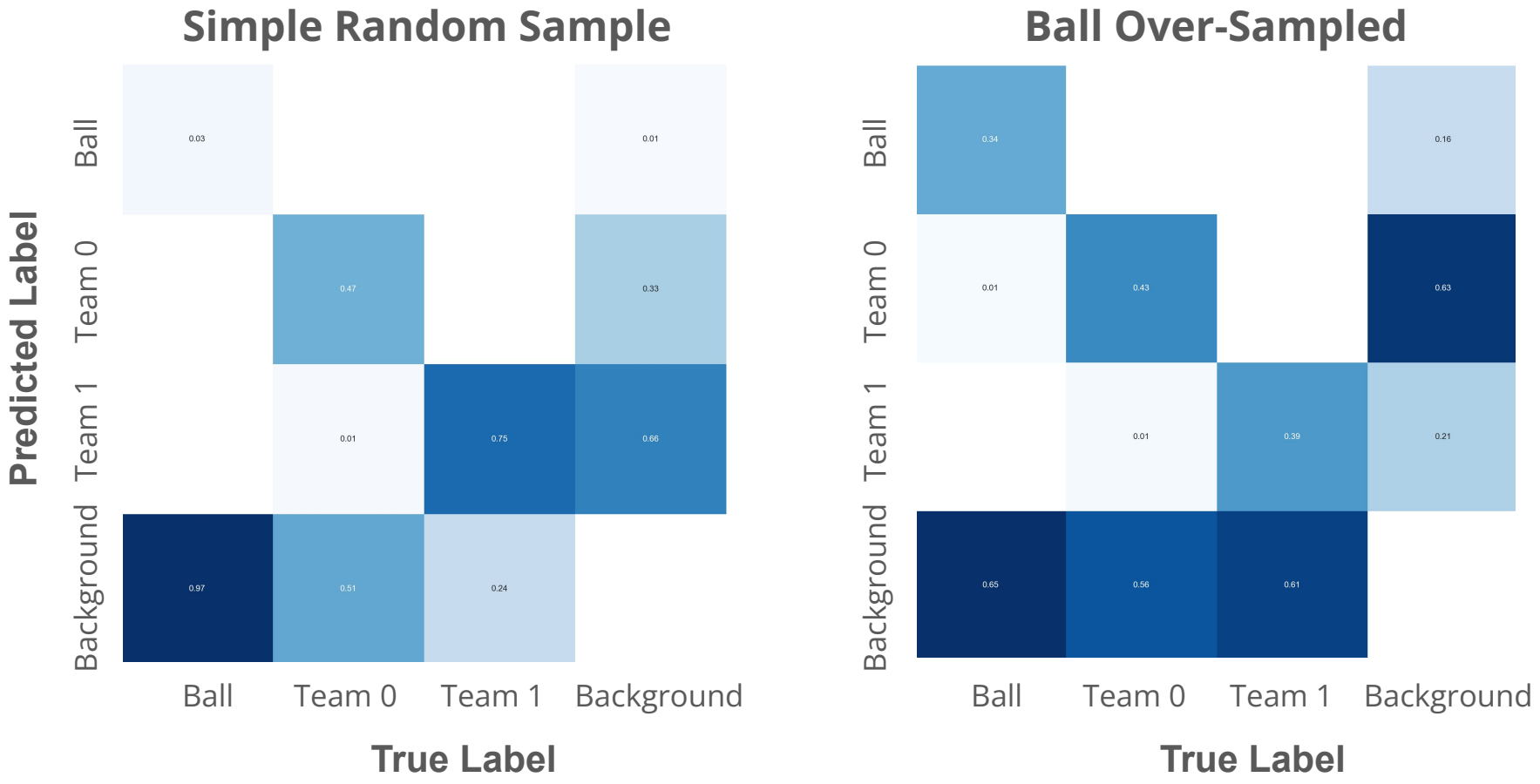
Improving The Models: Over-Sampling the Ball

Class Balance is Improved: 3.7% Ball to 7.6% Ball.



Effect of Over-Sampling on YOLOv8

Better for the Ball, Worse for the Players



Cropped & Over-Sampled Data Results

Model	Baseline or Improved	Box Loss	Obj. Loss	Class. Loss	Precision	Recall	mAP 0.5	mAP 0.5-0.95
YOLOv3	Improved	.057	.095	.0067	.772	.682	.673	.255
YOLOv3	Baseline	.116	.247	.0063	.586	.264	.206	.059
YOLOv5	Improved	.051	.032	.0054	.806	.678	.691	.264
YOLOv5	Baseline	.041	.024	.0010	.343	.202	.178	.054
YOLOv6	Improved	1.966	1.275	1.124	.760	.648	.648	.243
YOLOv6	Baseline	3.397	2.560	0.9375	.177	.044	.035	.013
YOLOv7	Improved	.0465	.0281	.0040	.841	.742	.745	.281
YOLOv7	Baseline	.019	.019	.0008	.447	.313	.264	.082
YOLOv8	Improved	1.858	1.267	1.016	0.512	0.299	0.255	0.074
YOLOv8	Baseline	2.156	.821	1.061	.401	.267	.298	.116

Tuning Parameters

Model	Tested Parameter	Value	Box Loss	Obj. Loss	Class. Loss	Precision	Recall	mAP 0.5	mAP 0.5-0.95
YOLOv8	Sampling	SRS	1.931	1.304	1.159	.721	.479	.383	.149
YOLOv8	**Sampling**	**Stratified**	**1.858**	**1.267**	**1.016**	**0.512**	**0.297**	**0.255**	**0.074**
YOLOv7	LRs: Start, End	0.1, 0.01	.050	.029	.0045	.807	.692	.706	.247
YOLOv7	**LRs**	**0.01, 0.01**	**.0465**	**.0281**	**.0040**	**.841**	**.742**	**.745**	**.281**
YOLOv7	LRs	0.001, 0.01	.0484	.0287	.0053	.676	.713	.681	.246
YOLOv7	LRs	0.1, 0.1	.0516	.0297	.0047	.724	.587	.607	.186
YOLOv7	LRs	0.01, 0.1	.0474	.0284	.0041	.831	.733	.738	.269
YOLOv7	LRs	0.001, 0.1	.0485	.0287	.0051	.778	.7105	.7042	.264

Tuning Parameters

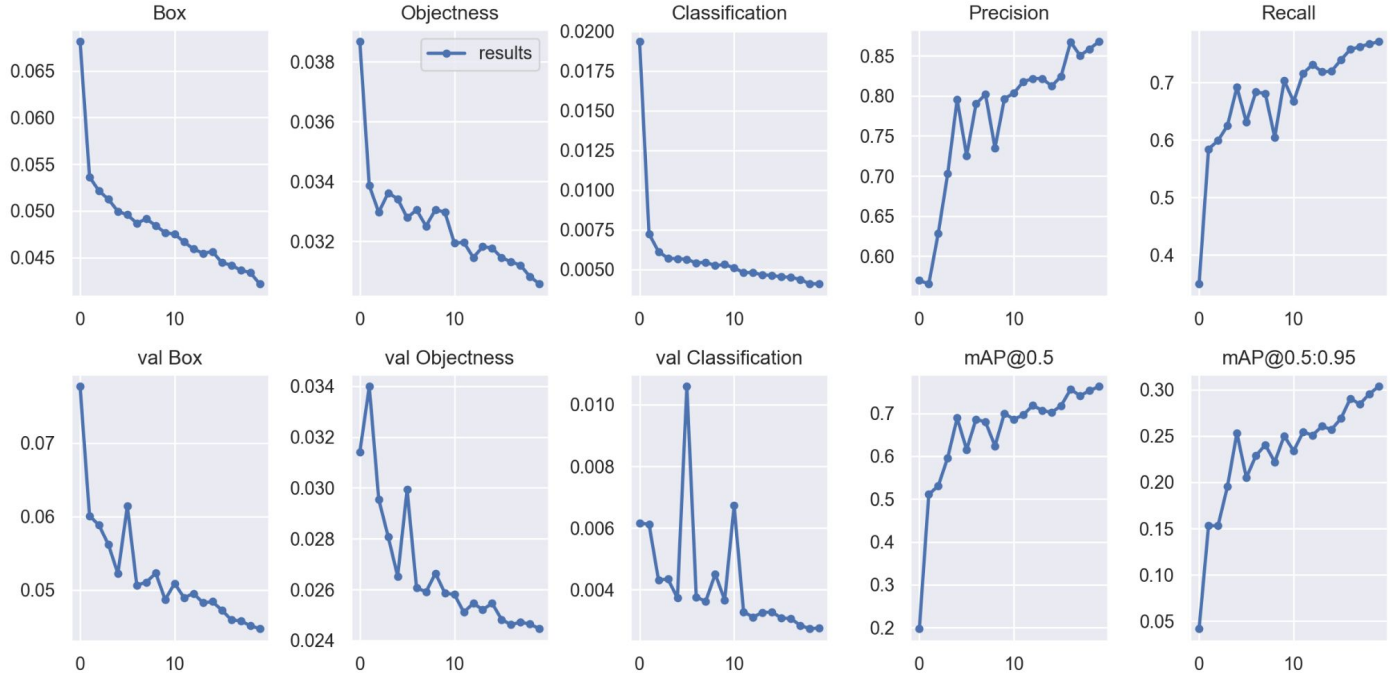
Model	Tested Parameter	Value	Box Loss	Obj. Loss	Class. Loss	Precision	Recall	mAP 0.5	mAP 0.5-0.95
YOLOv7	Batch Size	16	.0491	.0289	.0043	.816	.728	.728	.279
YOLOv7	Batch Size	32	.0495	.0288	.0043	.770	.689	.683	.236
YOLOv7	**Batch Size**	**64**	**.0465**	**.0281**	**.0040**	**.841**	**.742**	**.745**	**.281**
YOLOv7	**Optimizer**	**Adam**	**.0465**	**.0281**	**.0040**	**.841**	**.742**	**.745**	**.281**
YOLOv7	Optimizer	SGD	.0472	.0282	.0052	.808	.714	.717	.273
YOLOv7	Image Aug.	Basic	.0465	.0281	.004	.841	.742	.745	.281
YOLOv7	**Image Aug.**	**Extra**	**.0452**	**.0321**	**.005**	**.832**	**.737**	**.734**	**.282**

Choices for tuned model: YOLOv7, Stratified Sample, LRs of 0.01x0.01, Batch Size 64, Adam optimizer, with extra augmentations shear, perspective, & rotation included.

Additional Tuning Summary

- **Hyperparameters**
  - Best Model: **YOLOv7** over v3, v5, v6 and v8.
  - Best Optimizer: **Adam** over SGD
  - Best Learning Rate: **0.01** base, **0.01** final over 6 combos.
  - Best Batch Size: **64** over 32 & 16
  - **Adding** Additional Augmentations:
    - Random Image Shear: +- up to **10** Degrees
    - Random Image Perspective: +- up to **0.0003**
    - Random Image Rotation: +- up to **20** Degrees
    - This is on top of existing augmentations, such as left-right flip, hue & saturation changes, zoom & rescale, etc.
  - Additional Training Time: Up to **20** Epochs

Training Tuned YOLOv7 Model



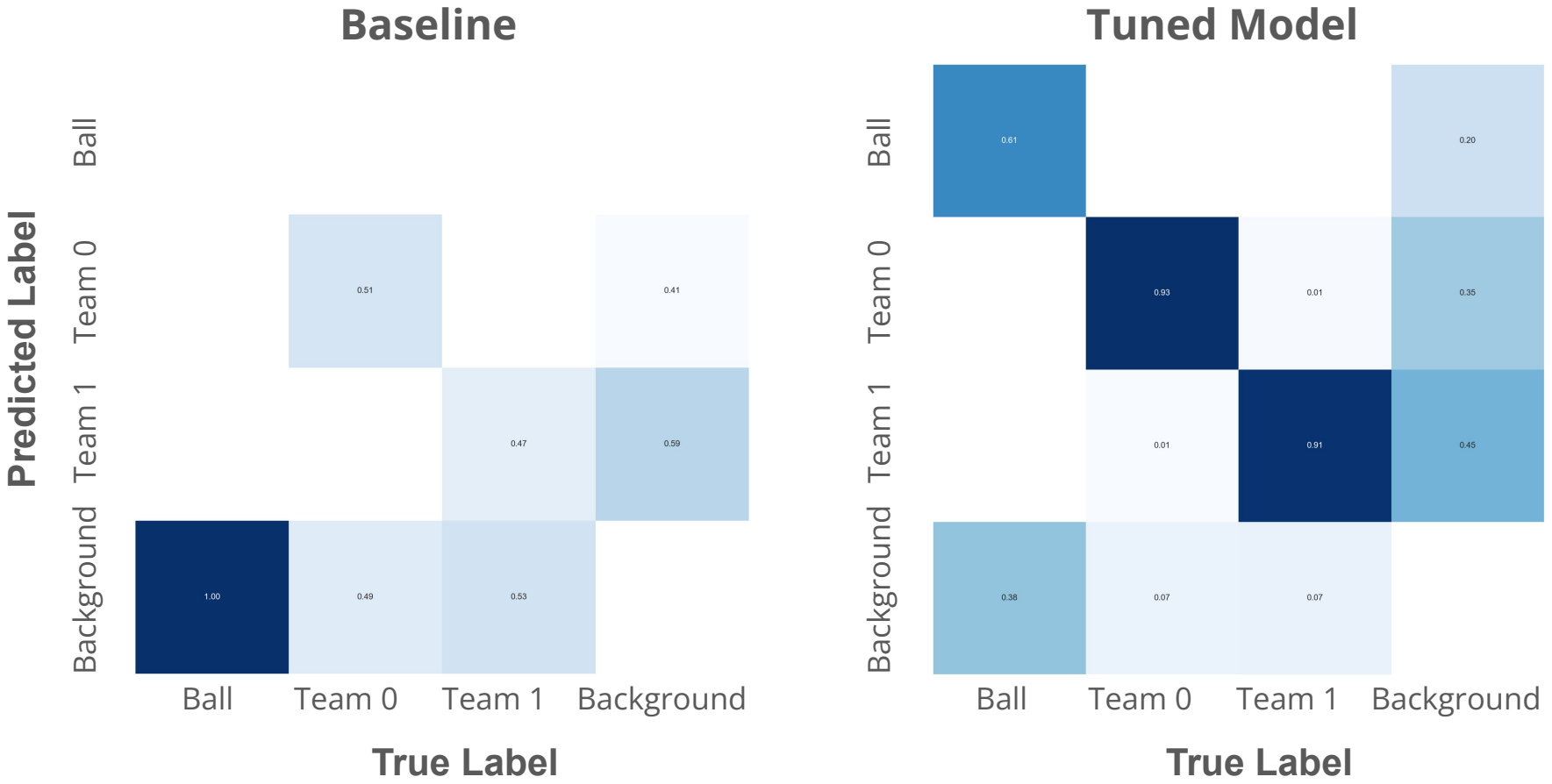
Precision, Recall, mAP, and loss are **near a plateau but can still improve** after 20 epochs.

Not much evidence of over-training: can still benefit from **adding more training epochs, sample size,** or **increasing network complexity**.

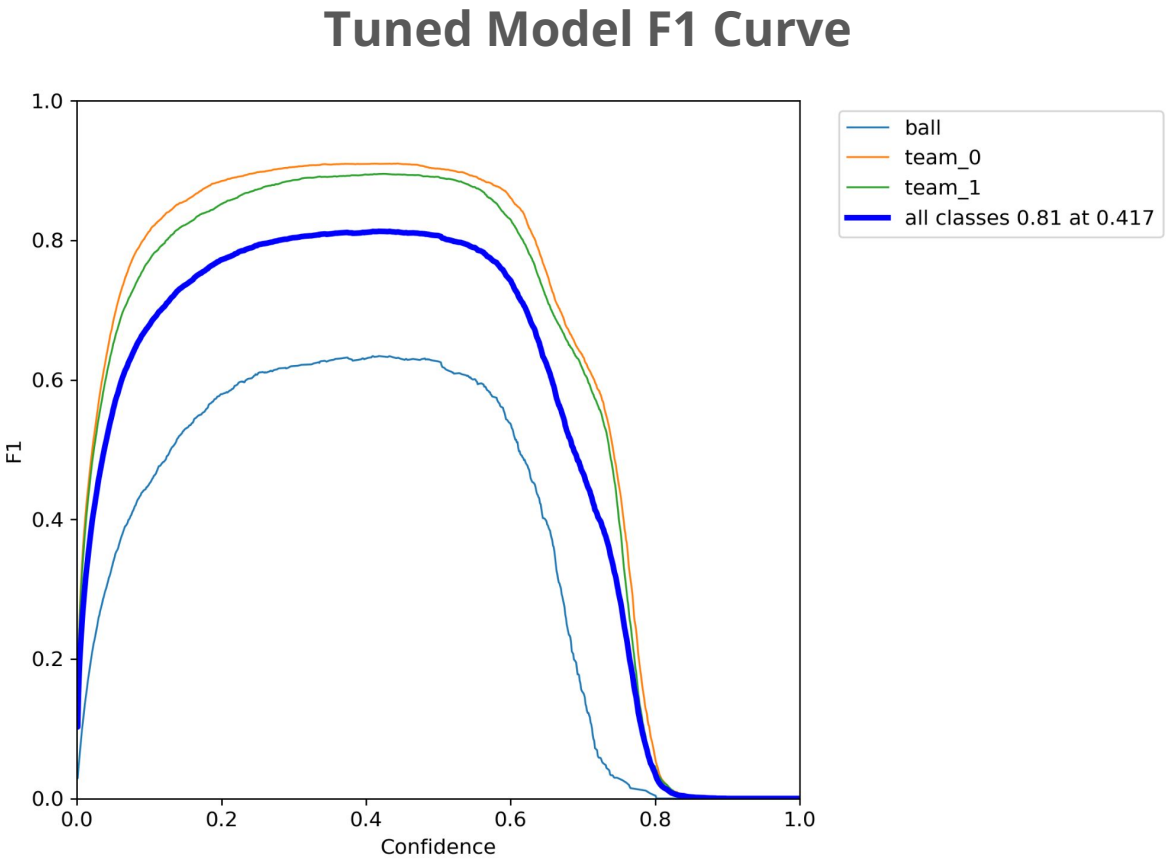
Tuned YOLOv7 Model - Results

Model	Epochs	Box Loss	Obj. Loss	Class. Loss	Precision	Recall	mAP 0.5	mAP 0.5-0.95
YOLOv7	5	0.050	0.033	0.0057	0.796	0.692	0.689	0.253
YOLOv7	10	0.048	0.033	0.0053	0.796	0.703	0.700	0.250
YOLOv7	15	0.045	0.032	0.0046	0.812	0.720	0.702	0.257
YOLOv7	**20**	**0.042**	**0.031**	**0.0041**	**0.868**	**0.771**	**0.763**	**0.303**

YOLOv7 Confusions Improvement



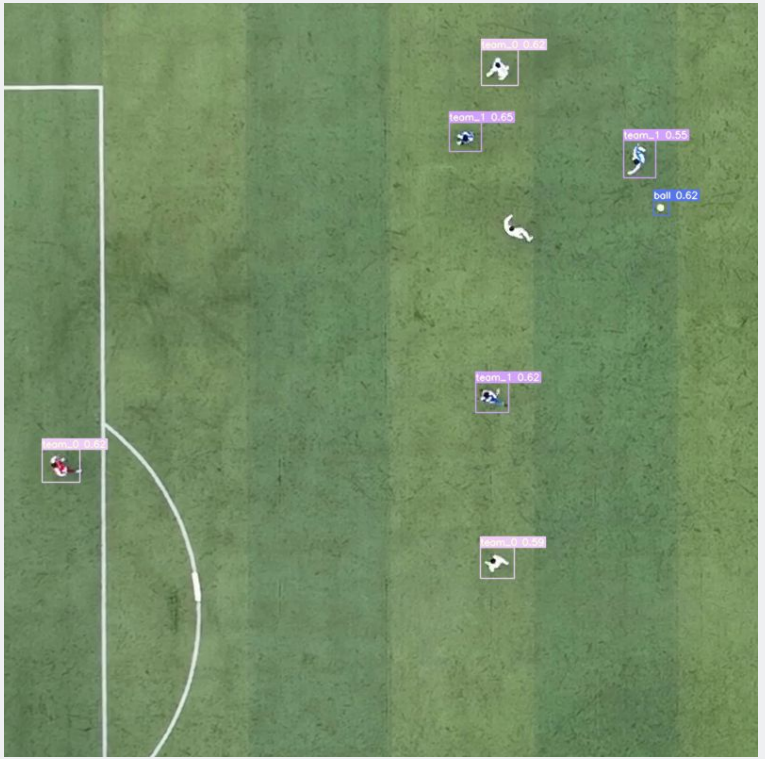
YOLOv7 Accuracy Maximized Around 0.5 IoU



Example Images

Penalty Spot - Not a False Positive!

Ball Can Be Detected!



Example Images

Can Handle Overlapping Boxes!

Handles Top View & Wide View!

Limitations & Future Work

1. Compute Time & Hyperparameter Tuning

We tested several hyperparameters, but not all, and could not test all pairwise combinations. **Improve with additional testing.**

2. Generalization & Bias

All the images are from the *same match*. This could easily create bias & overtuning. **Improve by combining with separate image data.**

3. Low Complexity & Training Time

Model training has not fully plateaued yet. We can still improve results by **adding epochs**, **increasing sample size**, **tuning hyperparameters**, and **increasing the size** of the neural network.

Conclusions

- Our tuned model achieves **87%** overall precision, **80%** overall recall, and **78%** mAP.5 / **32%** mAP.5-.95 on the test set.
  - Per-class:
    - Ball: **78%** P, **60%** R, **53%** mAP.5, **17%** mAP .5-.95
    - Team 0: **94%** P, **90%** R, **91%** mAP.5, **40%** mAP .5-.95
    - Team 1: **93%** P, **90%** R, **91%** mAP.5, **39%** mAP .5-.95
  - **Not quite at our goal** of 90% precision & recall per class
  - **Significantly improved from baseline** values below 50%, and near 0% recall for the ball.

- Work can be improved by **overcoming limitations:**
  - More experiments, hyperparameter tuning, and augmentation tests.
  - Sampling data from multiple matches instead of just one.
  - Training time, model size, and model complexity might still increase accuracy.


Q
&
A

Works Cited

Uchida, I., et al. (2022) “SoccerTrack: A Dataset and Tracking Algorithm for Soccer with Fish-eye and Drone Videos,” (“SoccerTrack”), kaggle.com, available at https://www.kaggle.com/datasets/atomscott/soccertrack/data.

Redmon, J., et al. (2016). “You Only Look Once: Unified, Real-Time Object Detection,” (“YOLO Object Detection”), *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Redmon, J., and Farhadi, A., (2018). “YOLOv3: An incremental improvement,” arXiv 1804.02767.

Jocher, G. (2020). YOLOv5 by Ultralytics (Version 7.0). https://doi.org/10.5281/zenodo.3908559.

Li, C., et al. (2023). YOLOv6 by Meituan Vision AI Department (Version 3.0), https://doi.org/10.48550/arXiv.2301.05586.

Wang, C., et al. (2022). “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.” arXiv 2207.02696.

Jocher, G., Chaurasia, A., & Qiu, J. (2023). Ultralytics YOLO (Version 8.0.0). https://github.com/ultralytics/ultralytics.

Works Cited

T. Majidzadeh and E. Ndedi, Project Proposal, “Detection of Ball and Player Location from Frame-by-Frame Soccer Match Videos,” MIDS W207 Spring 2024.

Homework Submissions of Etienne Ndedi, MIDS W207 Spring 2024.

Homework Submissions of Timothy Majidzadeh, MIDS W207 Spring 2024.

“How to Detect Small Objects: A Guide,” Jacob Solawetz, robloflow blog, available at https://blog.roboflow.com/detect-small-objects/.

“Imbalanced Data,” Machine Learning Foundational Courses, Google for Developers, available at developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data.

“Model Training with Ultralytics YOLO,” Ultralytics, available at https://docs.ultralytics.com/modes/train/#augmentation-settings-and-hyperparameters.

“Tips for Best Training Results,” Glenn Jocher, docs.ultralytics.com, available at https://docs.ultralytics.com/yolov5/tutorials/tips_for_best_training_results/#training-settings.

Contributions

Timothy Majidzadeh - I wrote the code to extract images from videos and process the labels. I wrote the code to crop the images, rescale the labels, and over-sample the ball for the 'square' dataset. I created the bar graphs showing the class imbalance. I implemented the YOLO, Logistic Regression, and CNN baselines, experiments, and final model, except for the YOLO Optimizer experiment. I wrote the initial draft of the final paper, and revised and edited. I revised and edited the final presentation slides. I helped Etienne debug his programs.

Etienne Ndedi - Wrote alternative program to translate videos to images with additional purpose to validate number of images on a per-video basis. Wrote alternative/support program to provide EDA support for ball's bounding box statistics and baseline model. Validated Yolo programs using different hyperparameters in a different computing environment. Created presentation slides. Reviewed and edited the final paper. Scheduled the daily meetings.