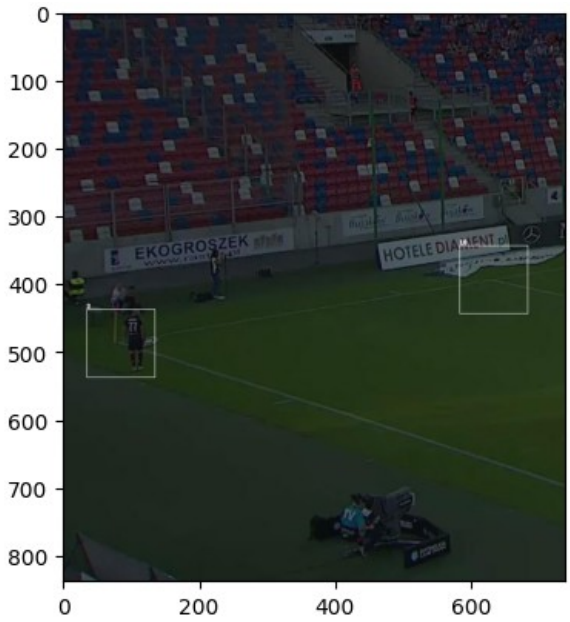
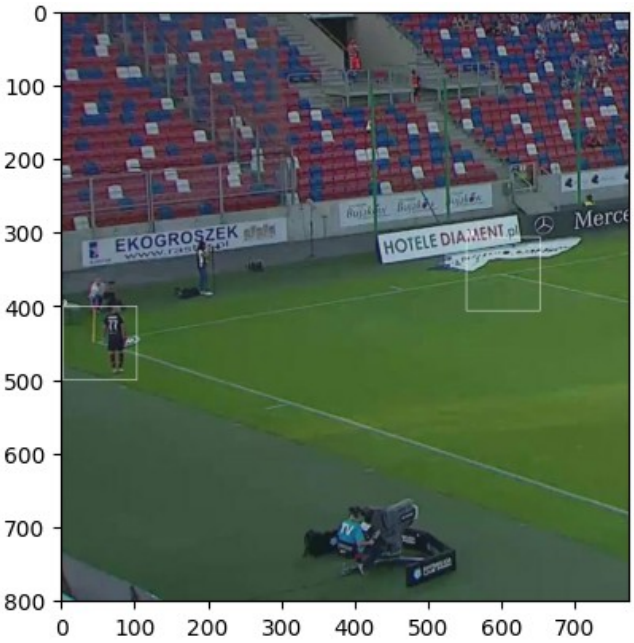
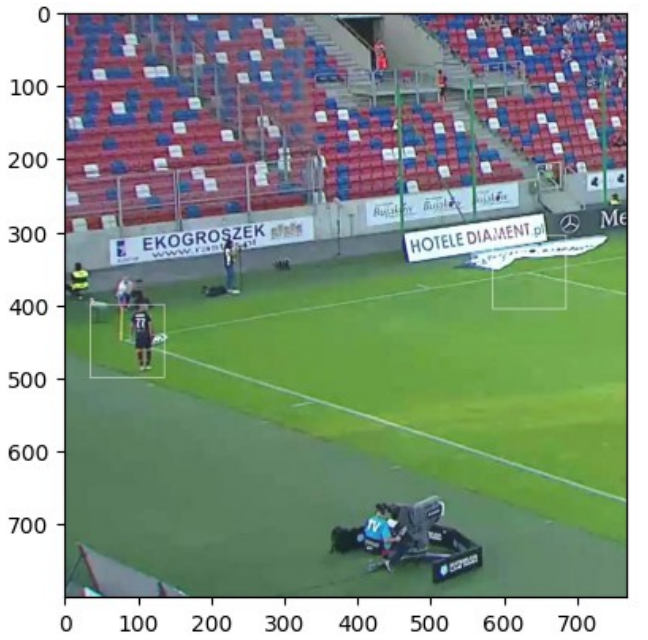
## *Experiment 1 – Data Augmentation*

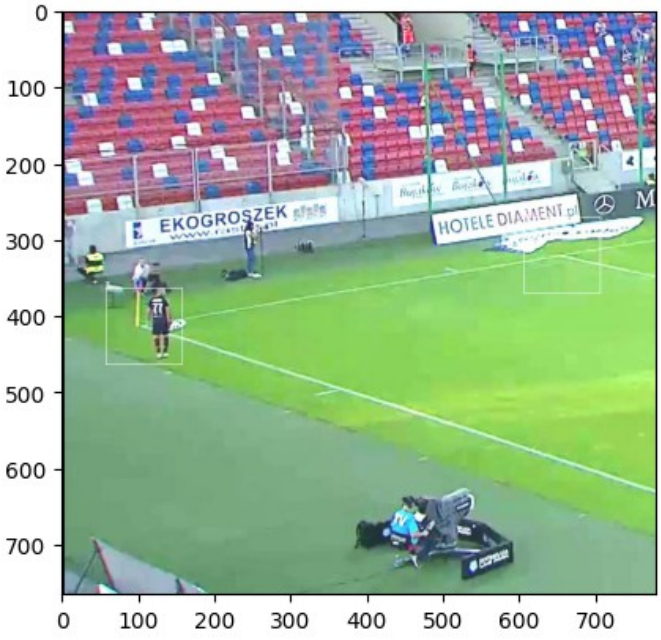
Data augmentation pipeline is included in Jupyter Notebook titled '4_Data_Augmentation.ipynb'. To increase samples of classes 3 and 23 both during training and validation (and testing in this case), clipping and brightening of images was applied.

- For each image containing classes 3 and 10 or 23 and 30 respectively, 4 images with increased or decreased brightness and different x,y of labelled class is generated – like below:



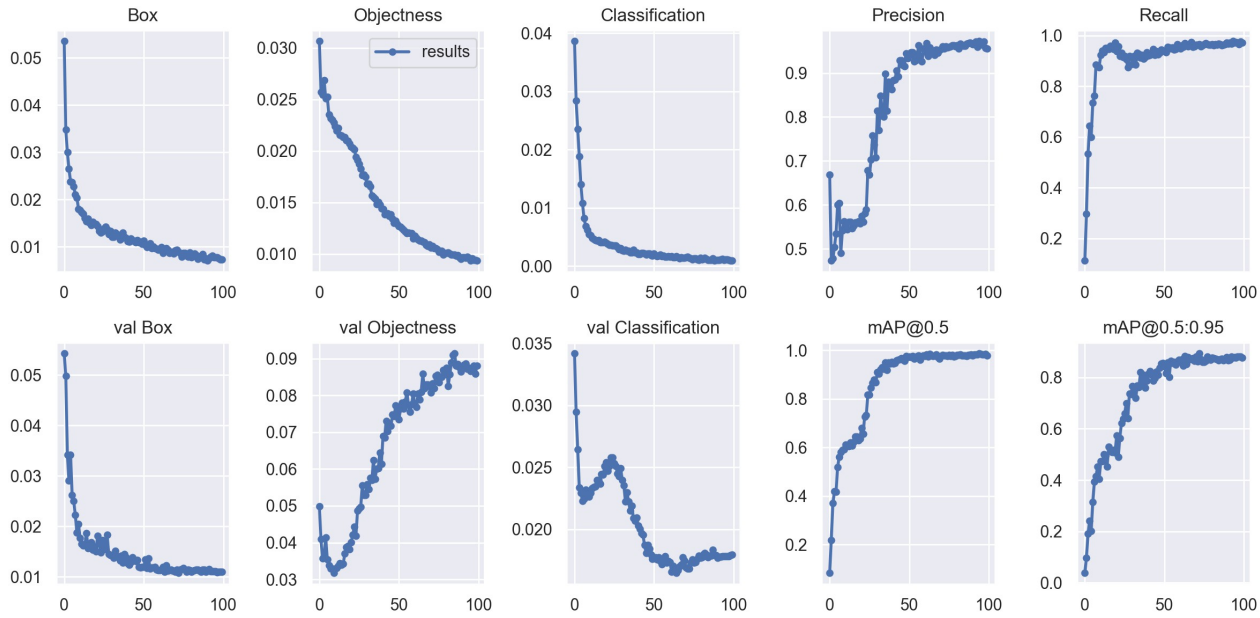





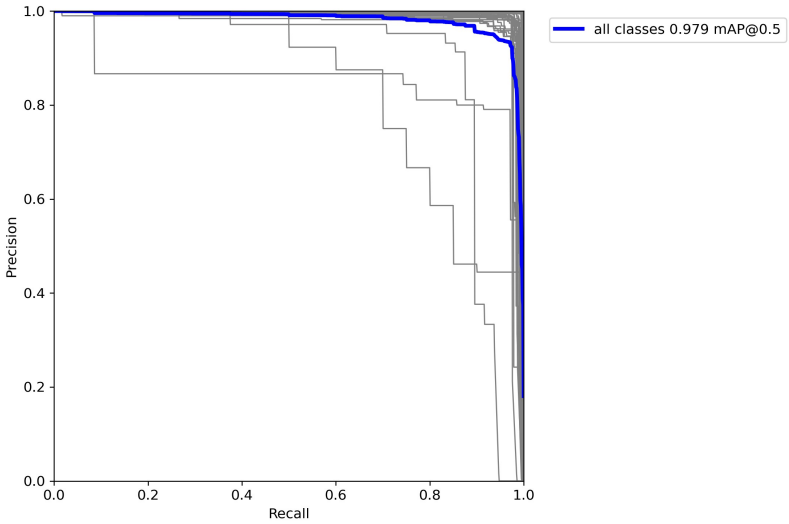

- 72 and 16 additonal images containing classes 3 and 10 were respectively added to training set and validation set.
- 108 and 28 additonal images containing classes 23 and 30 were respectively added to training set and validation set.

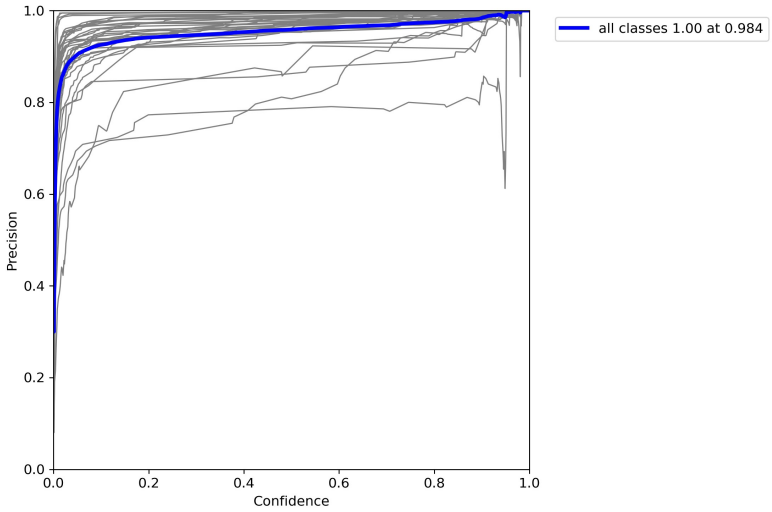
Unfortunately it did not improve performance of the model, although considering shapes of training charts (val Objectness – it started to fall near epoch 100 and classification – it is possible it starts to fall in next few epochs), new model might be underfitted. I think that this experiment needs another 100 epochs.

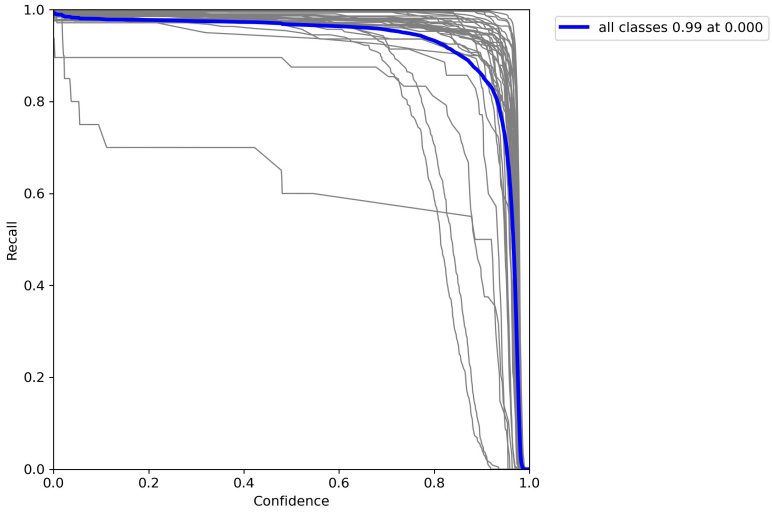


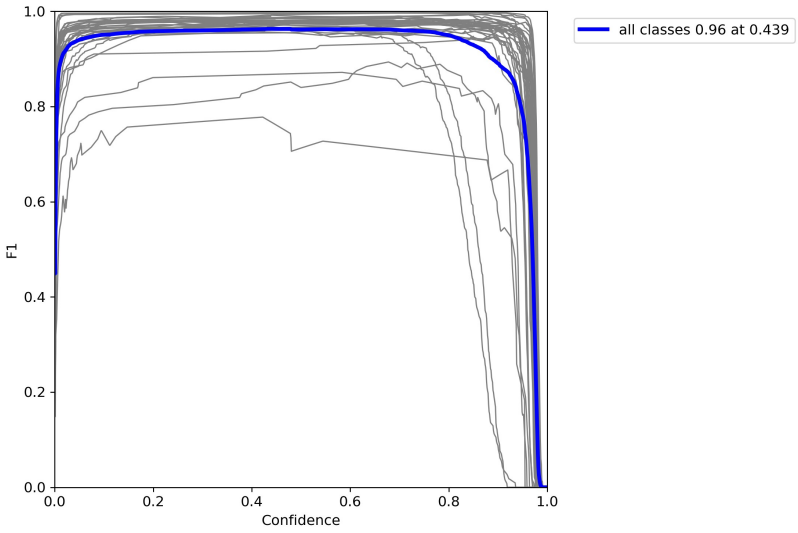

| Epoch | gpu_mem | box | obj | cls | total | labels | img_size | |
|---|---|---|---|---|---|---|---|---|
| 99/99 | 9.49G | 0.007318 | 0.009371 | 0.0009104 | 0.0176 | 92 | 640: | 100% |
| Class | Images | Labels | P | R | mAP@0.5 | mAP@0.5:.95: | | 100% |
| all | 796 | 6696 | 0.956 | 0.972 | 0.979 | 0.876 | | |
| 1 | 796 | 218 | 0.988 | 0.972 | 0.982 | 0.931 | | |
| 2 | 796 | 368 | 0.997 | 0.997 | 0.996 | 0.948 | | |
| 3 | 796 | 20 | 0.872 | 0.685 | 0.843 | 0.684 | | |
| 4 | 796 | 109 | 0.967 | 0.991 | 0.982 | 0.911 | | |
| 5 | 796 | 202 | 0.986 | 0.975 | 0.982 | 0.898 | | |
| 6 | 796 | 266 | 0.978 | 0.977 | 0.992 | 0.945 | | |
| 7 | 796 | 246 | 0.977 | 0.98 | 0.992 | 0.929 | | |
| 8 | 796 | 203 | 0.984 | 0.99 | 0.993 | 0.913 | | |
| 9 | 796 | 127 | 0.961 | 0.982 | 0.993 | 0.931 | | |
| 10 | 796 | 47 | 0.958 | 0.977 | 0.976 | 0.804 | | |
| 11 | 796 | 164 | 0.987 | 0.976 | 0.976 | 0.886 | | |
| 12 | 796 | 193 | 1 | 0.984 | 0.995 | 0.908 | | |
| 13 | 796 | 112 | 0.982 | 0.972 | 0.994 | 0.922 | | |
| 14 | 796 | 80 | 0.987 | 0.943 | 0.99 | 0.93 | | |
| 15 | 796 | 187 | 0.949 | 0.973 | 0.981 | 0.909 | | |
| 16 | 796 | 380 | 0.996 | 0.997 | 0.994 | 0.937 | | |
| 17 | 796 | 382 | 0.996 | 0.997 | 0.998 | 0.934 | | |
| 18 | 796 | 372 | 0.993 | 0.997 | 0.995 | 0.956 | | |
| 19 | 796 | 251 | 0.971 | 0.976 | 0.99 | 0.603 | | |
| 20 | 796 | 372 | 0.965 | 0.995 | 0.995 | 0.866 | | |
| 21 | 796 | 363 | 0.989 | 0.981 | 0.997 | 0.866 | | |
| 22 | 796 | 184 | 0.962 | 0.989 | 0.985 | 0.935 | | |
| 23 | 796 | 35 | 0.784 | 0.971 | 0.854 | 0.668 | | |
| 25 | 796 | 164 | 0.963 | 0.982 | 0.991 | 0.904 | | |
| 26 | 796 | 232 | 0.973 | 0.974 | 0.994 | 0.942 | | |
| 27 | 796 | 215 | 0.977 | 0.985 | 0.994 | 0.937 | | |
| 28 | 796 | 175 | 0.946 | 0.994 | 0.997 | 0.904 | | |
| 29 | 796 | 112 | 0.929 | 1 | 0.993 | 0.95 | | |
| 30 | 796 | 48 | 0.8 | 0.896 | 0.889 | 0.652 | | |
| 31 | 796 | 145 | 0.986 | 1 | 0.996 | 0.899 | | |
| 32 | 796 | 166 | 0.971 | 1 | 0.996 | 0.919 | | |
| 33 | 796 | 106 | 0.951 | 0.981 | 0.993 | 0.906 | | |
| 34 | 796 | 72 | 0.857 | 0.986 | 0.98 | 0.901 | | |
| 35 | 796 | 161 | 0.924 | 0.988 | 0.982 | 0.919 | | |
| 39 | 796 | 219 | 0.954 | 0.958 | 0.987 | 0.616 | | |

epochs completed in 11.250 hours.


all classes 0.979 mAP@0.5
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

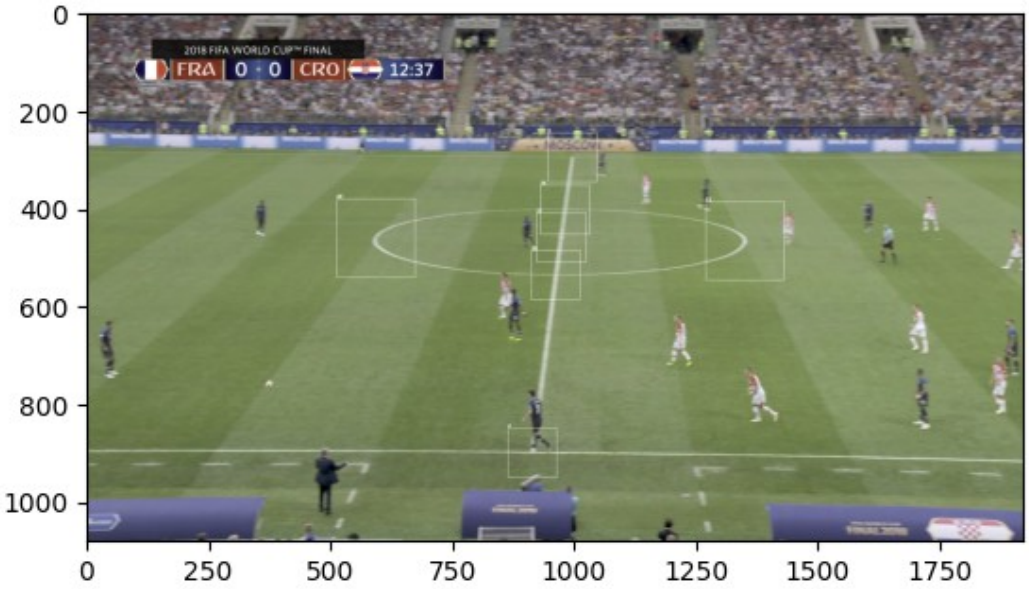

all classes 1.00 at 0.984
Precision
Confidence
0.0
0.2
0.4
0.6
0.8
1.0


all classes 0.99 at 0.000
Recall
Confidence
0.0
0.2
0.4
0.6
0.8
1.0

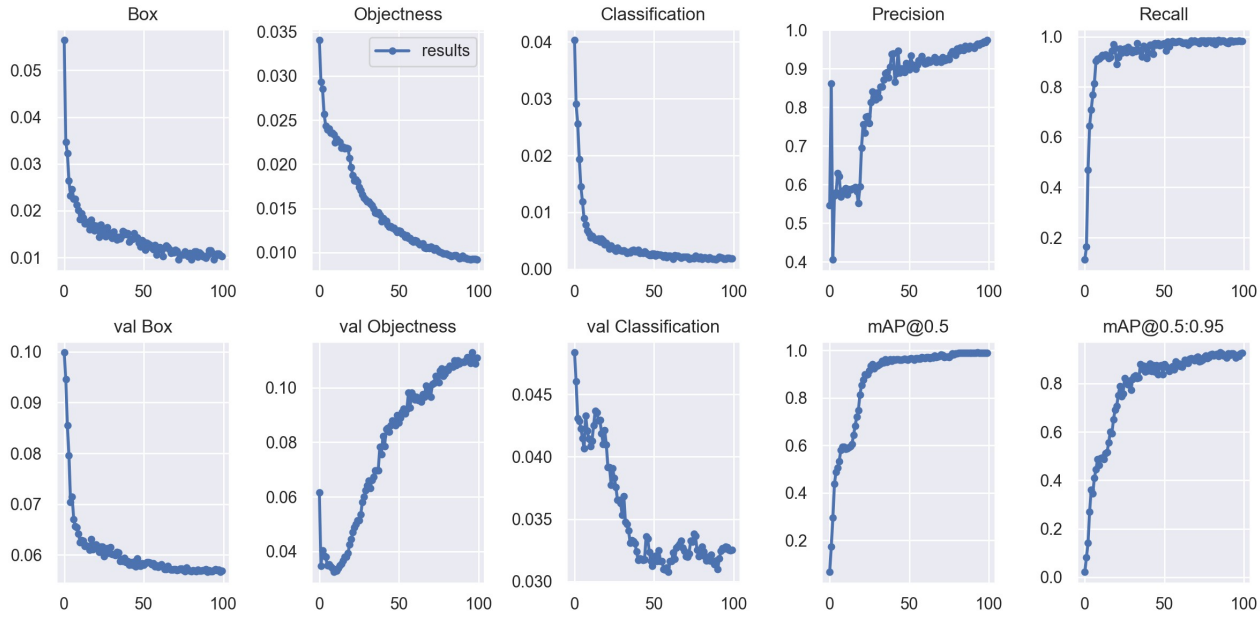



## *Experiment 2 – Bigger bounding box for some classes*

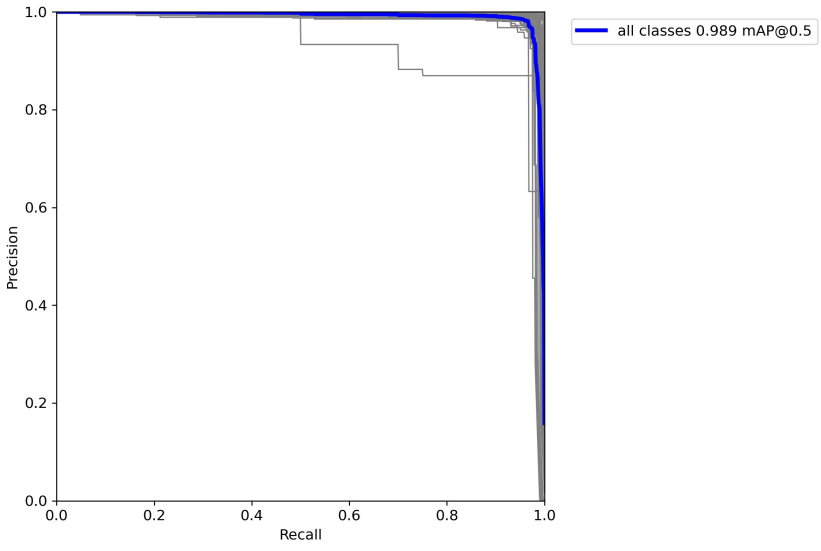
Data augmentation pipeline is included in Jupyter Notebook titled '4_Data_Augmentation.ipynb'. This experiment featured bigger bounding boxes (by 60 %) for classes: 3, 23, 10, 30, 39, 19, 20, 21.

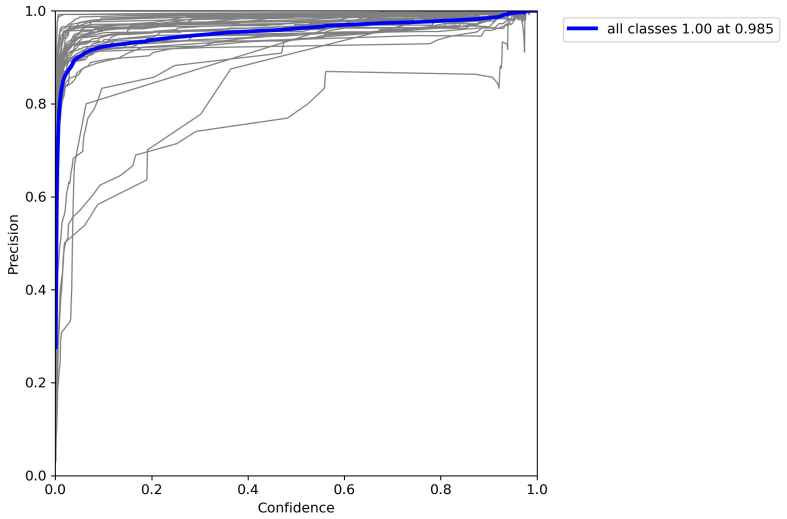


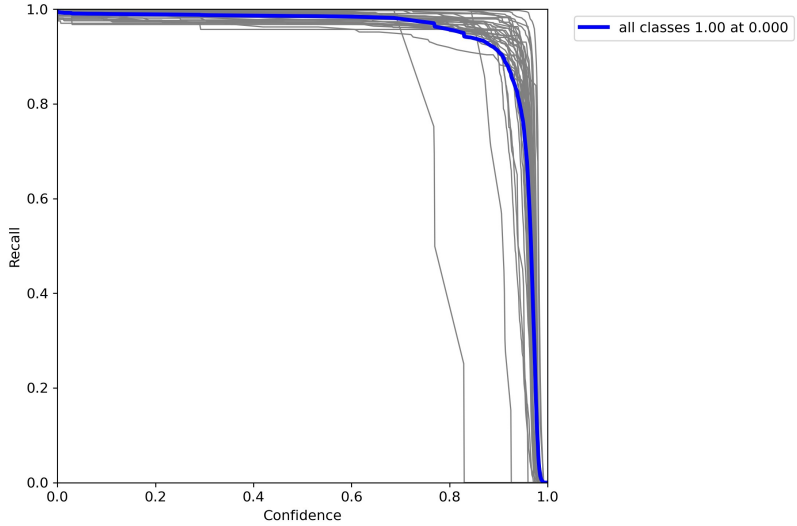

This model might be underfitted too and perhaps more epochs would help, but the results are a bit better and definitaly better balanced (based on mAP@.5:.95). On the other hand average precision for classification is slightly lower – I think it is underfitted especially since precision is still increasing.

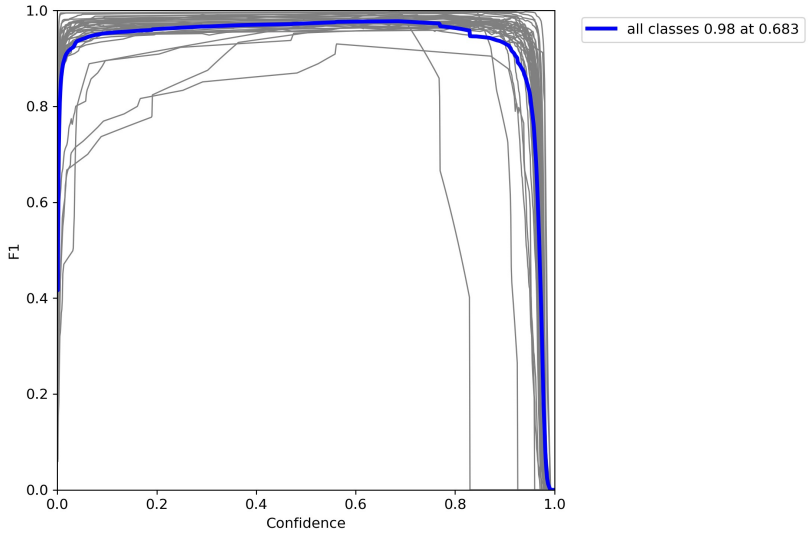


```
     Epoch    gpu_mem       box       obj       cls     total    labels  img_size
     99/99      7.44G   0.01024  0.009215   0.00185    0.0213        12       640: 100%|
                Class      Images      Labels                 P           R      mAP@.5  mAP@.5:.95:
                  all         765        6608             0.974       0.982       0.989       0.925
                    1         765         218             0.958       0.968       0.989        0.96
                    2         765         368             0.981       0.997       0.996       0.983
                    3         765           4             0.999           1       0.995       0.995
                    4         765         109             0.954       0.991       0.992        0.96
                    5         765         202              0.99       0.968       0.977       0.924
                    6         765         266              0.99       0.974       0.994       0.962
                    7         765         246             0.987       0.988       0.989       0.944
                    8         765         203             0.995       0.983        0.99       0.924
                    9         765         127             0.992       0.972       0.995       0.961
                   10         765          31             0.954       0.968       0.982        0.96
                   11         765         164             0.992       0.957       0.983       0.926
                   12         765         193             0.994       0.984       0.995       0.946
                   13         765         112             0.991       0.963       0.995        0.95
                   14         765          80             0.987       0.979       0.991       0.958
                   15         765         187             0.983        0.95       0.978       0.915
                   16         765         380             0.994       0.997       0.995       0.974
                   17         765         382             0.997       0.997       0.997       0.959
                   18         765         372             0.995       0.997       0.995       0.984
                   19         765         251             0.992       0.985       0.996       0.744
                   20         765         372             0.955       0.981       0.993       0.915
                   21         765         363             0.978       0.964       0.995       0.914
                   22         765         184             0.973       0.978       0.985       0.956
                   23         765           7             0.958           1       0.995       0.936
                   25         765         164              0.97       0.971       0.987       0.925
                   26         765         232             0.982       0.983       0.994       0.963
                   27         765         215              0.98       0.991       0.995       0.956
                   28         765         175             0.994       0.989       0.996       0.927
                   29         765         112             0.956           1       0.992       0.951
                   30         765          20             0.867        0.98       0.943       0.681
                   31         765         145              0.96       0.985       0.995       0.919
                   32         765         166             0.986           1        0.99       0.929
                   33         765         106             0.928       0.991        0.99       0.905
                   34         765          72             0.945       0.986       0.978       0.915
                   35         765         161             0.963       0.966       0.983       0.927
                   39         765         219             0.964       0.979       0.988       0.733
100 epochs completed in 11.602 hours.
```




all classes 1.00 at 0.985
Precision
Confidence
0.0
0.2
0.4
0.6
0.8
1.0


all classes 1.00 at 0.000
Recall
Confidence
0.0
0.2
0.4
0.6
0.8
1.0


all classes 0.98 at 0.683
F1
Confidence
0.0
0.2
0.4
0.6
0.8
1.0

After testing with my metrics we can see that the results are much worse for particular classes. Model is probably underfitted. Note that if any problem with precision of bounday box attachment occurs, MSE increases drastically.