Object Detection

Models

Chess Dataset

Link: https://drive.google.com/drive/folders/18SJcs5TN_G5x_HjDYC1QrEI_fBczW2LW?usp=sharing

Data Distribution :

Train dataset : 50 Images
Validation dataset : 200 Images

Classes :

1. pieces
2. bishop
3. black-bishop
4. black-king
5. black-knight
6. black-pawn
7. black-queen
8. black-rook
9. white-bishop
10. white-king
11. white-knight
12. white-pawn
13. white-queen
14. white-rook

Annotation Format: MS-COCO

Model Implementations :

Primarily some baseline models have opted for training and the model is Sparse-RCNN.

Variant : sparse_rcnn_r50_fpn_300_proposals_crop_mstrain_480-800_3x

Scheduler: x3 (36 epochs)

Results :

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.179
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=1000 ] = 0.287
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=1000 ] = 0.197
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = -1.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.001
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.180
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.375
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=300 ] = 0.376
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=1000 ] = 0.376
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = -1.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.200
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.376
```

Looking at the result we can say the detection model has poorly objectified the pieces. In most cases, a few objects have been objectified and others have been left.

In some cases, we can see that there are overlaps too.

But in general the most annoying to see is 90 percent of the black pieces have been left unannotated and this tells that although it is trained on pre-trained weights, still it can not be counted as a good model to prepare predictions.

Failing on Sparse-RCNN, the next model opted for training was VFNet, as it had very high accuracy on SOTA coco-test dev and coco-minival.

Variant : vfnet_x101_64x4d_fpn_mdconv_c3-c5_mstrain_2x

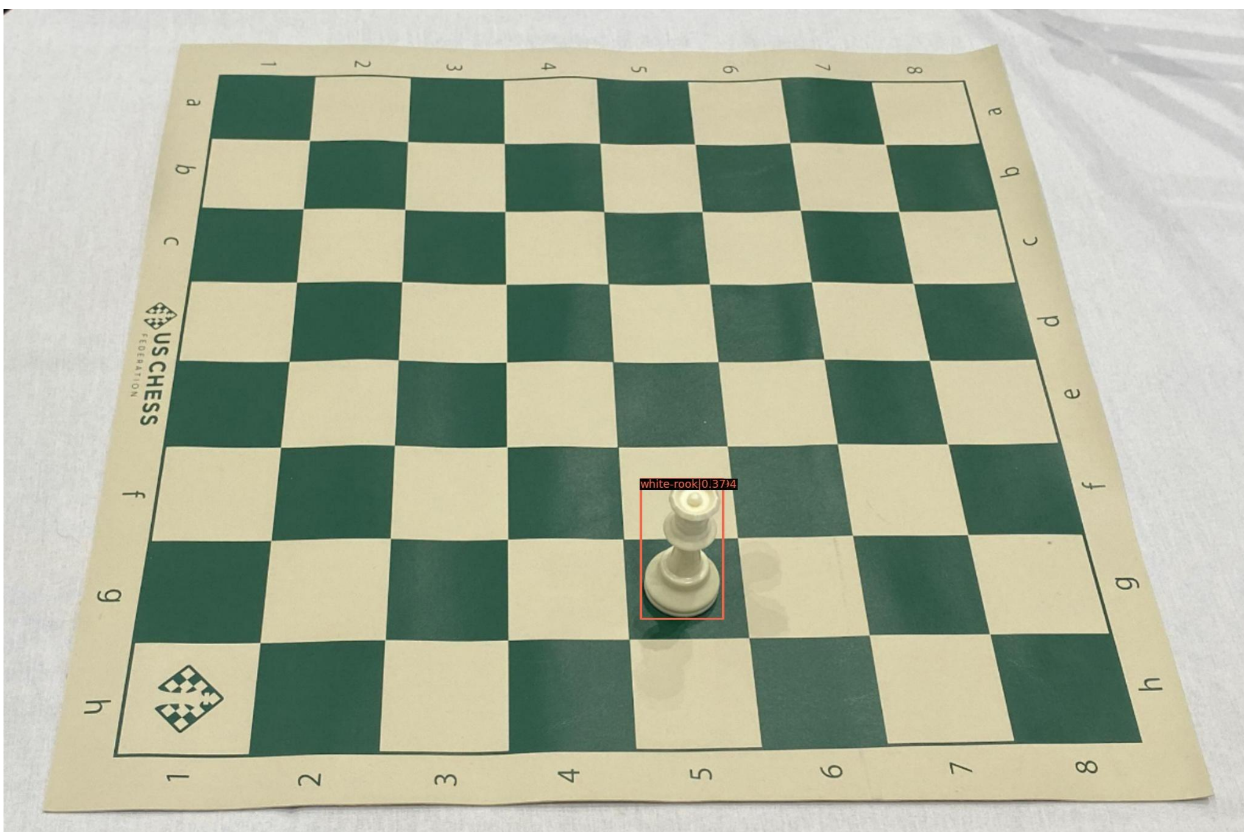
Scheduler: x2 (24 epochs)

Results :

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.658
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=1000 ] = 0.865
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=1000 ] = 0.799
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = -1.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.400
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.658
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.709
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=300 ] = 0.709
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=1000 ] = 0.709
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = -1.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.400
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.709
```

The results have been improved more than 45 percent and this can be stated as a nice improvement. All of the pieces have been predicted with this model.

It also shares that bbox overlapping has also been rectified by this model. But looking at the class predictions, most of the black bishops are having some errors. Some of them are being treated as black queens and some of them are classified as white queens. The other classes are having (in very few cases) different class labels.



Here we can see a white queen has been predicted as a white rook.

Though it had nice predictions, still improvisations are performed on further models.
The next model opted for was ResNest.It also had nice results in coco competitions.

Variant : cascade_rcnn_s101_fpn_syncbn-backbone+head_mstrain-range_1x

Scheduler: x2 (24 epochs)

Results :

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.676
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=1000 ] = 0.899
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=1000 ] = 0.818
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = -1.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.300
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.676
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.723
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=300 ] = 0.723
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=1000 ] = 0.723
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = -1.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.300
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.723
```

______We can see the results have been increased a bit and looking at the prediction it can be validated boldly.

All the pieces have been predicted and objectified. The overlapping problem is also not present in this scenario. But the major improvement of this model is that the class prediction problem has been rectified up to 95 percent. At most 1 or 2 objects are having wrong class labels and all others are correct.

Also in some cases, there were some pieces going shaded by other pieces, so, the model couldn’t predict those every time. So, it’s a small drawback but all other predictions have been rectified.


white-rook|0.97
white-bishop|0.95
white-pawn|0.72
white-queen|0.87
white-pawn|0.98
black-pawn|0.98
black-bishop|0.99
black-rook|0.99
black-pawn|0.99
white-pawn|0.99
black-pawn|0.96
black-king|0.99
white-knight|0.97
black-bishop|0.99
black-knight|0.99
white-pawn|0.99
black-pawn|0.97
white-king|0.98
black-queen|0.93
white-pawn|0.99
black-pawn|0.99
white-knight|0.94
white-pawn|0.99
black-pawn|0.99
black-knight|0.96
white-bishop|0.91
white-pawn|0.99
black-pawn|0.99
white-rook|0.98
white-pawn|0.98
black-pawn|0.98
black-rook|0.99
US CHESS
FEDERATION


white-rook|0.97
white-pawn|0.98
black-pawn|0.98
black-rook|0.97
white-knight|0.89
white-pawn|0.99
black-pawn|0.96
black-knight|0.72
white-bishop|0.40
white-pawn|0.98
black-pawn|0.98
black-bishop|0.98
white-queen|0.69
white-pawn|0.99
black-pawn|0.99
white-king|0.97
black-king|0.97
white-pawn|0.99
black-pawn|0.99
white-bishop|0.95
black-bishop|0.94
white-pawn|0.99
black-pawn|0.99
white-knight|0.94
white-pawn|0.99
black-pawn|0.99
black-knight|0.85
white-rook|0.97
white-pawn|0.98
black-pawn|0.99
black-rook|0.99
US CHESS
FEDERATION

After ResNest, the next model chosen was DETR. DETR is a transformer-based model, so not choosing any model from the RCNN family can let us know the predictability of different class models.

Variant : deformable_detr_twostage_refine_r50_16x2_50e

Scheduler: 50 epochs

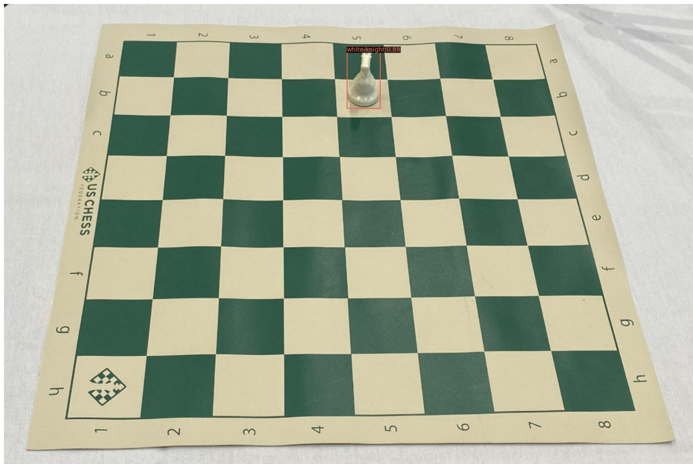
Results :

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.681
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=1000 ] = 0.891
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=1000 ] = 0.845
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = -1.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.500
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.681
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.735
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=300 ] = 0.735
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=1000 ] = 0.735
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = -1.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.500
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.735
```

The results are better than the last time. By looking at the predictions, we can see the previous rectification of Sparse-RCNN and VFNet have also been rectified in this case.

In addition, where some of the partially hidden objects were not identified in the last scenario with REsNest, have been omitted too and this prediction also can be treated as a manual level labeling.

The singular object images also have been predicted nicely.



In addition to all these, this prediction only has objectified the pieces on the board and that is what is needed for the AI chess-playing bot.

At next tried another model that is GFL. The bbox mAP was very near 0.5, so I opted in hope of getting nice predictions.

Variant : gfl_x101_32x4d_fpn_dconv_c4-c5_mstrain_2x

Scheduler: x2 (24 epochs)

Results :

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.693
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=1000 ] = 0.901
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=1000 ] = 0.849
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = -1.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.400
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.693
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.741
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=300 ] = 0.741
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=1000 ] = 0.741
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = -1.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.400
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.741
```

The results have been upgraded a bit, but the prediction is similar to the DETR model. It can be guessed that the bounding box coordinates are now more well predicted.




Conclusion: Using lower class RCNN variant models couldn't reach up to the mark and vice versa. The cascade R CNN variant produces the best results but in the case of transformers, the DETR based model which is also a baseline of the family reached a nice prediction accuracy. As the higher model needs more GPU power it couldn't be trained but give us hopes to find more improvements with it.