Image Classification

Object Detection for chess pieces

Linus Fackler and Fernando Colman

About the dataset

The dataset we are using is from [https://public.roboflow.com/object-detection/chess-full (https://public.roboflow.com/object-detection/chess-full)](https://public.roboflow.com/object-detection/chess-full) The reason we chose this dataset was because it has the specific labeling type needed for training a YoloV7 model. The dataset consists of imgages of chessboard with chesspieces. There are 13 different classes: 'bishop', 'black-bishop', 'black-king', 'black-knight', 'black-pawn', 'black-queen', 'black-rook', 'white-bishop', 'white-king', 'white-knight', 'white-pawn', 'white-queen', 'white-rook'

What the model should be able to predict

The model should be able to recognize chess pieces and identify which type of chess piece it is (bishop, pawn, rook, ...)

Preprocessing of Data

There is no preprocessing that needs to be done here.

Split into train, test, val

This step is also not necessary with this dataset, as it already comes with a premade split. There are 202 train images, 58 validation and 29 test.

In [1]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import os
```

Sequential Model

Reading in data

I've had problems with my original chess dataset for this part of the project. That's why I am using a tensorflow example dataset, so that I can at least get some results.

In [38]:
```
import numpy as np
import os
import PIL
import PIL.Image
import tensorflow as tf
import tensorflow_datasets as tfds
import pathlib

from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

In [127]:
```
import pathlib
dataset_url = "https://storage.googleapis.com/download.tensorflow.org/example_
images/flower_photos.tgz"
data_dir = tf.keras.utils.get_file('flower_photos', origin=dataset_url, untar=
True)
data_dir = pathlib.Path(data_dir)
```

In [128]:
```
image_count = len(list(data_dir.glob('*/*.jpg')))
print("Number of images: ", image_count)
```

```
Number of images:  3670
```

Here we took 270 images out of the list and put it into test

In [130]:
```
roses = list(data_dir.glob('roses/*'))
PIL.Image.open(str(roses[0]))
```

Out[130]:

In [41]:
```
# %cd ..
# %ls
# train_url = "/content/drive/MyDrive/Colab Notebooks/Chess dataset/train"
# test_url = "/content/drive/MyDrive/Colab Notebooks/Chess dataset/test"
# val_url = "/content/drive/MyDrive/Colab Notebooks/Chess dataset/valid"

# train_ds = tf.keras.utils.image_dataset_from_directory(train_url, labels=Non
e, validation_split=None, image_size=(200, 134), batch_size=16)
# test_ds = tf.keras.utils.image_dataset_from_directory(test_url, labels=None,
validation_split=None, image_size=(200, 134), batch_size=16)
# val_ds = tf.keras.utils.image_dataset_from_directory(val_url, labels=None, v
alidation_split=None, image_size=(200, 134), batch_size=16)
```

```
Found 202 files belonging to 1 classes.
Found 29 files belonging to 1 classes.
Found 58 files belonging to 1 classes.
```

Creating a dataset

In [100]:
```
batch_size = 32
img_height = 180
img_width = 180

train_ds = tf.keras.utils.image_dataset_from_directory(
  data_dir,
  validation_split=0.4,
  subset="training",
  seed=123,
  image_size=(img_height, img_width),
  batch_size=batch_size)
```

```
Found 3670 files belonging to 5 classes.
Using 2202 files for training.
```

In [101]:
```
val_ds = tf.keras.utils.image_dataset_from_directory(
  data_dir,
  validation_split=0.2,
  subset="validation",
  seed=123,
  image_size=(img_height, img_width),
  batch_size=batch_size)
```

```
Found 3670 files belonging to 5 classes.
Using 734 files for validation.
```

In [107]:
```
test_ds = tf.keras.utils.image_dataset_from_directory(
  data_dir,
  validation_split=None,
  subset="validation",
  seed=123,
  image_size=(img_height, img_width),
  batch_size=batch_size)
```

```
Found 3670 files belonging to 5 classes.
```

In [61]:
```
class_names = train_ds.class_names
print("Classes: ", class_names)
```

```
Classes:  ['daisy', 'dandelion', 'roses', 'sunflowers', 'tulips']
```

Visualizing the Data

In [62]:

```
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 10))
for images, labels in train_ds.take(1):
  for i in range(9):
    ax = plt.subplot(3, 3, i + 1)
    plt.imshow(images[i].numpy().astype("uint8"))
    plt.title(class_names[labels[i]])
    plt.axis("off")
```

Creating a model

In [80]:
```
num_classes = len(class_names)

model = tf.keras.models.Sequential([
  tf.keras.layers.Rescaling(1./255, input_shape=(img_height, img_width, 3)),
  tf.keras.layers.Conv2D(16, 3, padding='same', activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(32, 3, padding='same', activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(64, 3, padding='same', activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dense(num_classes)
])
```

In [73]:
```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 flatten_1 (Flatten)         (None, 784)               0         
                                                                 
 dense_3 (Dense)             (None, 512)               401920    
                                                                 
 dropout_2 (Dropout)         (None, 512)               0         
                                                                 
 dense_4 (Dense)             (None, 512)               262656    
                                                                 
 dropout_3 (Dropout)         (None, 512)               0         
                                                                 
 dense_5 (Dense)             (None, 5)                 2565      
                                                                 
=================================================================
Total params: 667,141
Trainable params: 667,141
Non-trainable params: 0
_________________________________________________________________
```

In [83]:
```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=T
rue),
              metrics=['accuracy'])
```

Training the model

In [84]:
```
history = model.fit(train_ds, batch_size=32, epochs=10, validation_data=val_d
s)
```

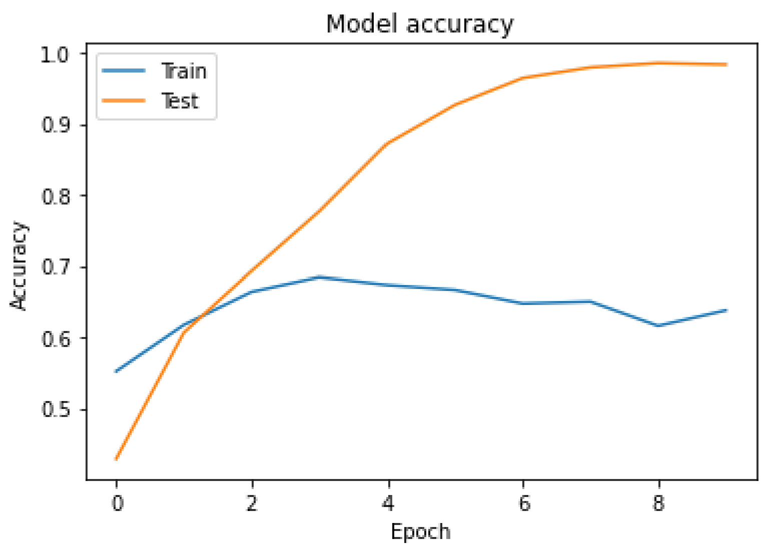
```
Epoch 1/10
92/92 [==============================] - 16s 75ms/step - loss: 1.2986 - accur
acy: 0.4285 - val_loss: 1.0989 - val_accuracy: 0.5518
Epoch 2/10
92/92 [==============================] - 7s 70ms/step - loss: 0.9951 - accura
cy: 0.6063 - val_loss: 0.9612 - val_accuracy: 0.6172
Epoch 3/10
92/92 [==============================] - 7s 72ms/step - loss: 0.8066 - accura
cy: 0.6928 - val_loss: 0.8758 - val_accuracy: 0.6635
Epoch 4/10
92/92 [==============================] - 7s 73ms/step - loss: 0.5961 - accura
cy: 0.7766 - val_loss: 0.8428 - val_accuracy: 0.6839
Epoch 5/10
92/92 [==============================] - 7s 70ms/step - loss: 0.3783 - accura
cy: 0.8719 - val_loss: 1.0283 - val_accuracy: 0.6730
Epoch 6/10
92/92 [==============================] - 7s 71ms/step - loss: 0.2230 - accura
cy: 0.9264 - val_loss: 1.2306 - val_accuracy: 0.6662
Epoch 7/10
92/92 [==============================] - 7s 71ms/step - loss: 0.1255 - accura
cy: 0.9642 - val_loss: 1.4477 - val_accuracy: 0.6471
Epoch 8/10
92/92 [==============================] - 7s 71ms/step - loss: 0.0692 - accura
cy: 0.9792 - val_loss: 1.5392 - val_accuracy: 0.6499
Epoch 9/10
92/92 [==============================] - 7s 71ms/step - loss: 0.0488 - accura
cy: 0.9854 - val_loss: 1.8818 - val_accuracy: 0.6158
Epoch 10/10
92/92 [==============================] - 9s 100ms/step - loss: 0.0634 - accur
acy: 0.9833 - val_loss: 1.7081 - val_accuracy: 0.6376
```

In [85]:
```
history.history.keys()
```

Out[85]:
```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

Evaluate the model

In [88]:
```
import matplotlib.pyplot as plt

# Plot training & validation accuracy values
plt.plot(history.history['val_accuracy'])
plt.plot(history.history['accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```



CNN architecture

We are going to use the same dataset as we did for the sequential.

In [97]:
```
num_filters = 8
filter_size = 3
pool_size = 2

model2 = tf.keras.models.Sequential(
    [
        tf.keras.layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
        tf.keras.layers.MaxPooling2D(pool_size=(2, 2)),
        tf.keras.layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        tf.keras.layers.MaxPooling2D(pool_size=(2, 2)),
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dropout(0.5),
        tf.keras.layers.Dense(num_classes, activation="softmax"),
    ]
)
```

In [94]:
```
model2.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_3 (Conv2D)           (None, 30, 30, 32)        896       
                                                                 
 max_pooling2d_3 (MaxPooling  (None, 15, 15, 32)       0         
 2D)                                                             
                                                                 
 conv2d_4 (Conv2D)           (None, 13, 13, 64)        18496     
                                                                 
 max_pooling2d_4 (MaxPooling  (None, 6, 6, 64)         0         
 2D)                                                             
                                                                 
 flatten_3 (Flatten)         (None, 2304)              0         
                                                                 
 dropout_4 (Dropout)         (None, 2304)              0         
                                                                 
 dense_8 (Dense)             (None, 5)                 11525     
                                                                 
=================================================================
Total params: 30,917
Trainable params: 30,917
Non-trainable params: 0
_________________________________________________________________
```

Train model

In [98]:
```
model2.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])

history2 = model2.fit(train_ds, batch_size=32, epochs=10, validation_data=val_ds)
```

```
Epoch 1/10

/usr/local/lib/python3.8/dist-packages/tensorflow/python/util/dispatch.py:1082: UserWarning: "`sparse_categorical_crossentropy` received `from_logits=True`, but the `output` argument was produced by a sigmoid or softmax activation and thus does not represent logits. Was this intended?"
  return dispatch_target(*args, **kwargs)

92/92 [==============================] - 10s 93ms/step - loss: 55.3183 - accuracy: 0.2708 - val_loss: 1.6131 - val_accuracy: 0.2766
Epoch 2/10
92/92 [==============================] - 8s 79ms/step - loss: 1.4799 - accuracy: 0.3784 - val_loss: 1.6985 - val_accuracy: 0.2779
Epoch 3/10
92/92 [==============================] - 11s 112ms/step - loss: 1.2903 - accuracy: 0.4704 - val_loss: 1.8022 - val_accuracy: 0.3011
Epoch 4/10
92/92 [==============================] - 7s 74ms/step - loss: 1.1900 - accuracy: 0.5228 - val_loss: 2.1110 - val_accuracy: 0.3052
Epoch 5/10
92/92 [==============================] - 7s 75ms/step - loss: 1.0196 - accuracy: 0.6144 - val_loss: 2.2935 - val_accuracy: 0.3542
Epoch 6/10
92/92 [==============================] - 7s 72ms/step - loss: 0.9315 - accuracy: 0.6458 - val_loss: 2.7418 - val_accuracy: 0.3338
Epoch 7/10
92/92 [==============================] - 7s 72ms/step - loss: 0.8230 - accuracy: 0.7006 - val_loss: 2.5255 - val_accuracy: 0.3706
Epoch 8/10
92/92 [==============================] - 7s 74ms/step - loss: 0.7458 - accuracy: 0.7371 - val_loss: 2.5079 - val_accuracy: 0.3896
Epoch 9/10
92/92 [==============================] - 7s 72ms/step - loss: 0.6754 - accuracy: 0.7650 - val_loss: 3.3713 - val_accuracy: 0.3202
Epoch 10/10
92/92 [==============================] - 7s 72ms/step - loss: 0.6966 - accuracy: 0.7619 - val_loss: 3.1006 - val_accuracy: 0.3529
```

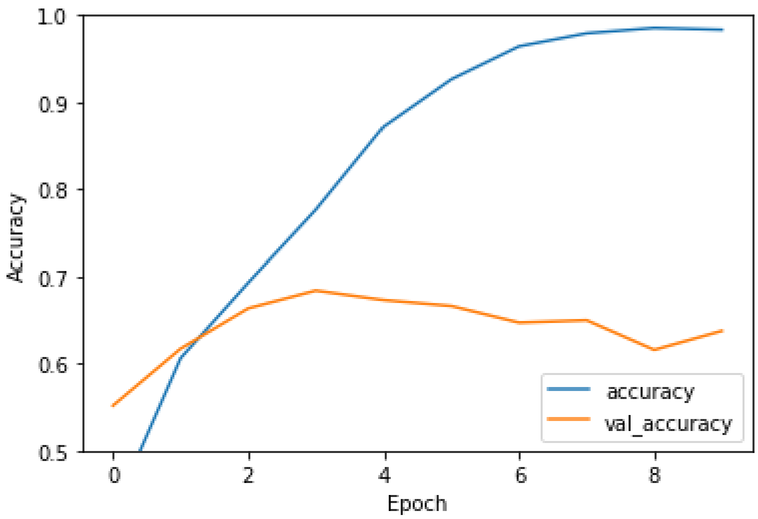
Evaluate the model

In [99]:

```
plt.plot(history.history['accuracy'], label='accuracy')
plt.plot(history.history['val_accuracy'], label = 'val_accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.ylim([0.5, 1])
plt.legend(loc='lower right')
```

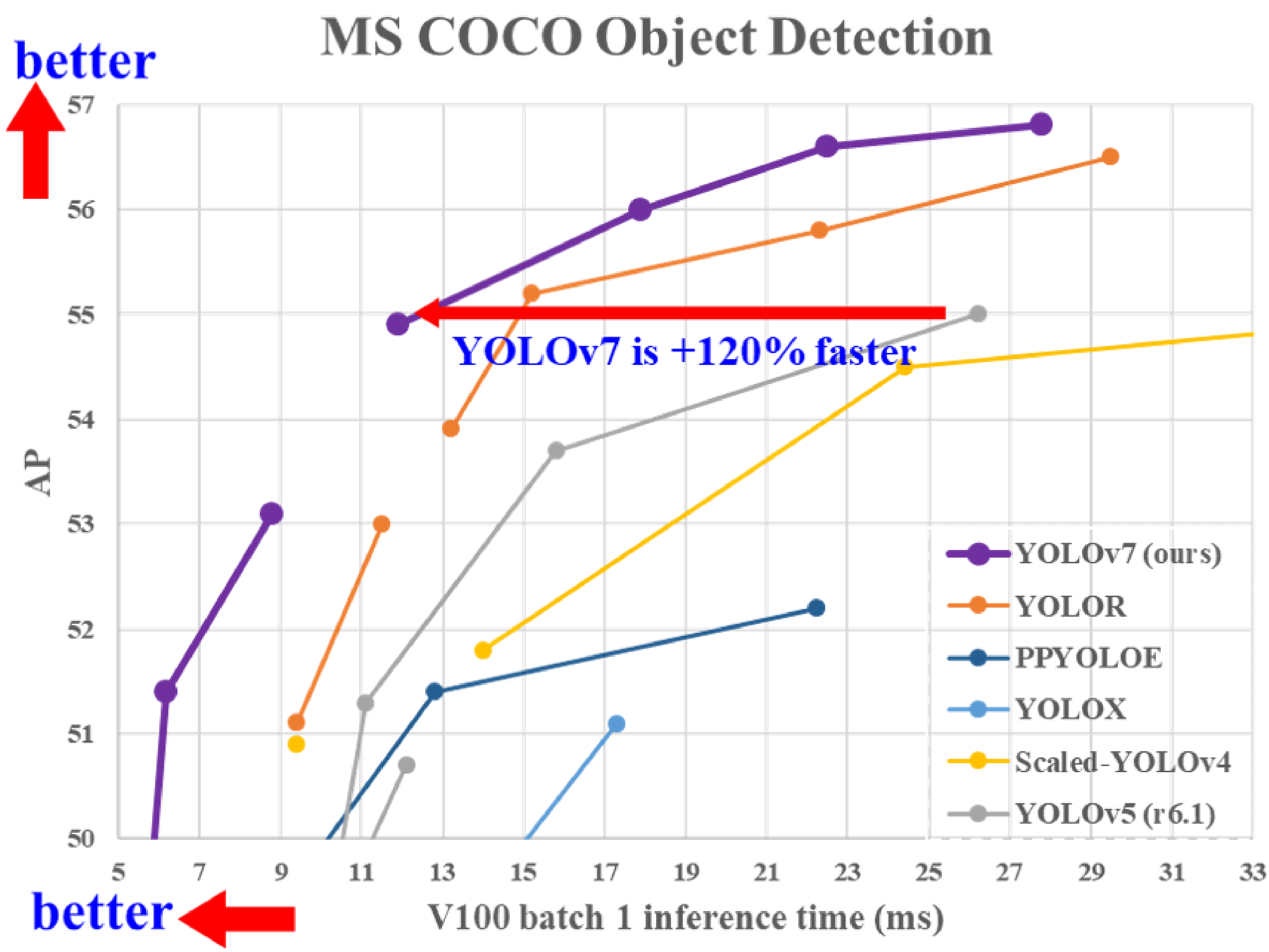
Out[99]:

```
<matplotlib.legend.Legend at 0x7f06f028b610>
```



Pretrained Model

I've decided to use the YOLOv7 model for this project, as it seems to be the best open-source model for objects detection. I personally have experience with it, which was another factor as to why I chose this.



In [2]:

```
# Downloading YOLO v7 Code
!git clone https://github.com/WongKinYiu/yolov7.git
%cd yolov7
!ls
```

```
Cloning into 'yolov7'...
remote: Enumerating objects: 1094, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 1094 (delta 0), reused 2 (delta 0), pack-reused 1091
Receiving objects: 100% (1094/1094), 69.85 MiB | 23.76 MiB/s, done.
Resolving deltas: 100% (521/521), done.
/content/yolov7
cfg     detect.py  hubconf.py  models     requirements.txt  tools         uti
ls
data    export.py  inference   paper      scripts           train_aux.py
deploy  figure     LICENSE.md  README.md  test.py           train.py
```

In [36]:
```
# Pre-trained weights
!wget https://github.com/WongKinbYiu/yolov7/releases/download/v0.1/yolov7.p
t
```

```
--2022-12-05 05:00:24--  https://github.com/WongKinbYiu/yolov7/releases/downl
oad/v0.1/yolov7.pt
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 404 Not Found
2022-12-05 05:00:24 ERROR 404: Not Found.
```

Resizing images

I decided to later comment out this section, as yolov7 does its own image resizing. Originallz I wanted to do this so that it would not train with 2048px images, which would take too long.

I still kept it for future references, as I think the code is very helpful. Also, in case I ever want to use a different pretrained model, this might become helpful.

In [12]:
```
# from PIL import Image

# directory = "/content/drive/MyDrive/Colab Notebooks/Chess/train/images"

# basewidth = 640

# for filename in os.listdir(directory):
#   f = os.path.join(directory, filename)
#   img = Image.open(f)
#   wpercent = (basewidth / float(img.size[0]))
#   hsize = int((float(img.size[1]) * float(wpercent)))
#   img = img.resize((basewidth, hsize), Image.ANTIALIAS)
#   img.save(f)
```

Training

This is where the **transfer learning** comes into play. We will now train the pretrained model with a new type of image. The images won't consist any of the 80 previous classes from the YOLOv7 (which actually is from the coco dataset), now we will only have our 13 classes, which include the chess figures.

In [133]:
```
%cd "/content/drive/MyDrive/Colab Notebooks/yolov7"
!python train.py --epochs 30 --weights yolov7.pt --batch-size 16 --data "/content/drive/MyDrive/Colab Notebooks/Chess/data.yaml" --name yolov7-custom --device 0 --exist-ok
```

```
/content/drive/MyDrive/Colab Notebooks/yolov7
YOLOR 🚀 2022-12-5 torch 1.12.1+cu113 CUDA:0 (Tesla T4, 15109.75MB)

Namespace(adam=False, artifact_alias='latest', batch_size=16, bbox_interval=-
1, bucket='', cache_images=False, cfg='', data='/content/drive/MyDrive/Colab
Notebooks/Chess/data.yaml', device='0', entity=None, epochs=30, evolve=False,
exist_ok=True, freeze=[0], global_rank=-1, hyp='data/hyp.scratch.p5.yaml', im
age_weights=False, img_size=[640, 640], label_smoothing=0.0, linear_lr=False,
local_rank=-1, multi_scale=False, name='yolov7-custom', noautoanchor=False, n
osave=False, notest=False, project='runs/train', quad=False, rect=False, resu
me=False, save_dir='runs/train/yolov7-custom', save_period=-1, single_cls=Fal
se, sync_bn=False, total_batch_size=16, upload_dataset=False, v5_metric=Fals
e, weights='yolov7.pt', workers=8, world_size=1)
tensorboard: Start with 'tensorboard --logdir runs/train', view at http://loc
alhost:6006/
hyperparameters: lr0=0.01, lrf=0.1, momentum=0.937, weight_decay=0.0005, warm
up_epochs=3.0, warmup_momentum=0.8, warmup_bias_lr=0.1, box=0.05, cls=0.3, cl
s_pw=1.0, obj=0.7, obj_pw=1.0, iou_t=0.2, anchor_t=4.0, fl_gamma=0.0, hsv_h=
0.015, hsv_s=0.7, hsv_v=0.4, degrees=0.0, translate=0.2, scale=0.9, shear=0.
0, perspective=0.0, flipud=0.0, fliplr=0.5, mosaic=1.0, mixup=0.15, copy_past
e=0.0, paste_in=0.15, loss_ota=1
wandb: Install Weights & Biases for YOLOR logging with 'pip install wandb' (r
ecommended)
Overriding model.yaml nc=80 with nc=13

                 from  n    params  module                                  a
rguments
  0                -1  1       928  models.common.Conv
[3, 32, 3, 1]
  1                -1  1     18560  models.common.Conv
[32, 64, 3, 2]
  2                -1  1     36992  models.common.Conv
[64, 64, 3, 1]
  3                -1  1     73984  models.common.Conv
[64, 128, 3, 2]
  4                -1  1      8320  models.common.Conv
[128, 64, 1, 1]
  5                -2  1      8320  models.common.Conv
[128, 64, 1, 1]
  6                -1  1     36992  models.common.Conv
[64, 64, 3, 1]
  7                -1  1     36992  models.common.Conv
[64, 64, 3, 1]
  8                -1  1     36992  models.common.Conv
[64, 64, 3, 1]
  9                -1  1     36992  models.common.Conv
[64, 64, 3, 1]
 10  [-1, -3, -5, -6]  1         0  models.common.Concat
[1]
 11                -1  1     66048  models.common.Conv
[256, 256, 1, 1]
 12                -1  1         0  models.common.MP
[]
 13                -1  1     33024  models.common.Conv
[256, 128, 1, 1]
 14                -3  1     33024  models.common.Conv
[256, 128, 1, 1]
```

```
 15                -1  1    147712  models.common.Conv
[128, 128, 3, 2]
 16          [-1, -3]  1         0  models.common.Concat
[1]
 17                -1  1     33024  models.common.Conv
[256, 128, 1, 1]
 18                -2  1     33024  models.common.Conv
[256, 128, 1, 1]
 19                -1  1    147712  models.common.Conv
[128, 128, 3, 1]
 20                -1  1    147712  models.common.Conv
[128, 128, 3, 1]
 21                -1  1    147712  models.common.Conv
[128, 128, 3, 1]
 22                -1  1    147712  models.common.Conv
[128, 128, 3, 1]
 23  [-1, -3, -5, -6]  1         0  models.common.Concat
[1]
 24                -1  1    263168  models.common.Conv
[512, 512, 1, 1]
 25                -1  1         0  models.common.MP
[]
 26                -1  1    131584  models.common.Conv
[512, 256, 1, 1]
 27                -3  1    131584  models.common.Conv
[512, 256, 1, 1]
 28                -1  1    590336  models.common.Conv
[256, 256, 3, 2]
 29          [-1, -3]  1         0  models.common.Concat
[1]
 30                -1  1    131584  models.common.Conv
[512, 256, 1, 1]
 31                -2  1    131584  models.common.Conv
[512, 256, 1, 1]
 32                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
 33                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
 34                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
 35                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
 36  [-1, -3, -5, -6]  1         0  models.common.Concat
[1]
 37                -1  1   1050624  models.common.Conv
[1024, 1024, 1, 1]
 38                -1  1         0  models.common.MP
[]
 39                -1  1    525312  models.common.Conv
[1024, 512, 1, 1]
 40                -3  1    525312  models.common.Conv
[1024, 512, 1, 1]
 41                -1  1   2360320  models.common.Conv
[512, 512, 3, 2]
 42          [-1, -3]  1         0  models.common.Concat
[1]
 43                -1  1    262656  models.common.Conv
```

```
[1024, 256, 1, 1]
 44                -2  1    262656  models.common.Conv
[1024, 256, 1, 1]
 45                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
 46                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
 47                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
 48                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
 49  [-1, -3, -5, -6]  1         0  models.common.Concat
[1]
 50                -1  1   1050624  models.common.Conv
[1024, 1024, 1, 1]
 51                -1  1   7609344  models.common.SPPCSPC
[1024, 512, 1]
 52                -1  1    131584  models.common.Conv
[512, 256, 1, 1]
 53                -1  1         0  torch.nn.modules.upsampling.Upsample
[None, 2, 'nearest']
 54                37  1    262656  models.common.Conv
[1024, 256, 1, 1]
 55          [-1, -2]  1         0  models.common.Concat
[1]
 56                -1  1    131584  models.common.Conv
[512, 256, 1, 1]
 57                -2  1    131584  models.common.Conv
[512, 256, 1, 1]
 58                -1  1    295168  models.common.Conv
[256, 128, 3, 1]
 59                -1  1    147712  models.common.Conv
[128, 128, 3, 1]
 60                -1  1    147712  models.common.Conv
[128, 128, 3, 1]
 61                -1  1    147712  models.common.Conv
[128, 128, 3, 1]
 62[-1, -2, -3, -4, -5, -6]  1         0  models.common.Concat
[1]
 63                -1  1    262656  models.common.Conv
[1024, 256, 1, 1]
 64                -1  1     33024  models.common.Conv
[256, 128, 1, 1]
 65                -1  1         0  torch.nn.modules.upsampling.Upsample
[None, 2, 'nearest']
 66                24  1     65792  models.common.Conv
[512, 128, 1, 1]
 67          [-1, -2]  1         0  models.common.Concat
[1]
 68                -1  1     33024  models.common.Conv
[256, 128, 1, 1]
 69                -2  1     33024  models.common.Conv
[256, 128, 1, 1]
 70                -1  1     73856  models.common.Conv
[128, 64, 3, 1]
 71                -1  1     36992  models.common.Conv
[64, 64, 3, 1]
```

```
 72                -1  1     36992  models.common.Conv
[64, 64, 3, 1]
 73                -1  1     36992  models.common.Conv
[64, 64, 3, 1]
 74[-1, -2, -3, -4, -5, -6]  1         0  models.common.Concat
[1]
 75                -1  1     65792  models.common.Conv
[512, 128, 1, 1]
 76                -1  1         0  models.common.MP
[]
 77                -1  1     16640  models.common.Conv
[128, 128, 1, 1]
 78                -3  1     16640  models.common.Conv
[128, 128, 1, 1]
 79                -1  1    147712  models.common.Conv
[128, 128, 3, 2]
 80      [-1, -3, 63]  1         0  models.common.Concat
[1]
 81                -1  1    131584  models.common.Conv
[512, 256, 1, 1]
 82                -2  1    131584  models.common.Conv
[512, 256, 1, 1]
 83                -1  1    295168  models.common.Conv
[256, 128, 3, 1]
 84                -1  1    147712  models.common.Conv
[128, 128, 3, 1]
 85                -1  1    147712  models.common.Conv
[128, 128, 3, 1]
 86                -1  1    147712  models.common.Conv
[128, 128, 3, 1]
 87[-1, -2, -3, -4, -5, -6]  1         0  models.common.Concat
[1]
 88                -1  1    262656  models.common.Conv
[1024, 256, 1, 1]
 89                -1  1         0  models.common.MP
[]
 90                -1  1     66048  models.common.Conv
[256, 256, 1, 1]
 91                -3  1     66048  models.common.Conv
[256, 256, 1, 1]
 92                -1  1    590336  models.common.Conv
[256, 256, 3, 2]
 93      [-1, -3, 51]  1         0  models.common.Concat
[1]
 94                -1  1    525312  models.common.Conv
[1024, 512, 1, 1]
 95                -2  1    525312  models.common.Conv
[1024, 512, 1, 1]
 96                -1  1   1180160  models.common.Conv
[512, 256, 3, 1]
 97                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
 98                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
 99                -1  1    590336  models.common.Conv
[256, 256, 3, 1]
100[-1, -2, -3, -4, -5, -6]  1         0  models.common.Concat
```

```
[1]
101                -1  1   1049600  models.common.Conv
[2048, 512, 1, 1]
102                75  1    328704  models.common.RepConv
[128, 256, 3, 1]
103                88  1   1312768  models.common.RepConv
[256, 512, 3, 1]
104               101  1   5246976  models.common.RepConv
[512, 1024, 3, 1]
105   [102, 103, 104]  1     96930  models.yolo.Detect
[13, [[12, 16, 19, 36, 40, 28], [36, 75, 76, 55, 72, 146], [142, 110, 192, 24
3, 459, 401]], [256, 512, 1024]]
Model Summary: 407 layers, 37259330 parameters, 37259330 gradients

Transferred 554/560 items from yolov7.pt
Scaled weight_decay = 0.0005
Optimizer groups: 95 .bias, 95 conv.weight, 92 other
train: Scanning '/content/drive/MyDrive/Colab Notebooks/Chess/train/labels.ca
che' images and labels... 202 found, 0 missing, 0 empty, 1 corrupted: 100% 20
2/202 [00:00<?, ?it/s]
val: Scanning '/content/drive/MyDrive/Colab Notebooks/Chess/valid/labels.cach
e' images and labels... 58 found, 0 missing, 0 empty, 0 corrupted: 100% 58/58
[00:00<?, ?it/s]

autoanchor: Analyzing anchors... anchors/target = 6.38, Best Possible Recall
(BPR) = 1.0000
Image sizes 640 train, 640 test
Using 2 dataloader workers
Logging results to runs/train/yolov7-custom
Starting training for 30 epochs...

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
      0/29     7.95G   0.08947   0.02657   0.04188    0.1579       207
640: 100% 13/13 [01:18<00:00,  6.07s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95:   0% 0/2 [00:00<?, ?it/s]/usr/local/lib/python3.8/dist-packa
ges/torch/functional.py:478: UserWarning: torch.meshgrid: in an upcoming rele
ase, it will be required to pass the indexing argument. (Triggered internally
at  ../aten/src/ATen/native/TensorShape.cpp:2894.)
  return _VF.meshgrid(tensors, **kwargs)  # type: ignore[attr-defined]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:10<00:00,  5.26s/it]
                 all          58         386       0.139     0.00287     0.00
292     0.00146

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
      1/29      10.9G   0.08159   0.03236   0.04136    0.1553       257
640: 100% 13/13 [00:41<00:00,  3.16s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.15it/s]
                 all          58         386     0.00127      0.0123    0.000
135    4.16e-05

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
```

```
     2/29     10.9G   0.07564   0.03249   0.03979    0.1479       150
640: 100% 13/13 [00:45<00:00,  3.53s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.12it/s]
                 all          58         386      0.0255       0.155      0.0
192     0.00588

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     3/29     10.9G   0.06944   0.03325   0.03913    0.1418       162
640: 100% 13/13 [00:56<00:00,  4.33s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.05it/s]
                 all          58         386      0.0647       0.218      0.0
797      0.0289

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     4/29     10.9G   0.06427   0.03112   0.03728    0.1327       281
640: 100% 13/13 [00:53<00:00,  4.13s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.01it/s]
                 all          58         386       0.152       0.525       0.
117      0.0503

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     5/29     10.9G   0.05963   0.03064   0.03706    0.1273       192
640: 100% 13/13 [00:48<00:00,  3.70s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.04it/s]
                 all          58         386       0.118       0.597       0.
173      0.0735

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     6/29     10.9G   0.05459   0.02981   0.03654    0.1209        68
640: 100% 13/13 [00:49<00:00,  3.78s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:02<00:00,  1.04s/it]
                 all          58         386      0.0963       0.444       0.
178      0.0784

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     7/29     10.9G   0.05172   0.03089   0.03647    0.1191       199
640: 100% 13/13 [00:51<00:00,  3.95s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:02<00:00,  1.01s/it]
                 all          58         386      0.0744        0.79       0.
139      0.0546

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     8/29     10.9G   0.04735   0.03059   0.03579    0.1137       214
640: 100% 13/13 [00:50<00:00,  3.88s/it]
               Class      Images      Labels           P           R      mAP
```

```
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.15it/s]
                 all         58        386     0.0841      0.565      0.
137     0.0744

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
      9/29     10.9G   0.04544   0.03295   0.03565     0.114       207
640: 100% 13/13 [00:43<00:00,  3.38s/it]
               Class     Images     Labels          P          R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.17it/s]
                 all         58        386     0.0976      0.723      0.
199      0.108

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     10/29     10.9G   0.04738   0.03229   0.03611    0.1158       173
640: 100% 13/13 [00:47<00:00,  3.62s/it]
               Class     Images     Labels          P          R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.16it/s]
                 all         58        386      0.285      0.397      0.
254      0.148

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     11/29     10.9G   0.04267   0.03198    0.0354      0.11       162
640: 100% 13/13 [00:40<00:00,  3.13s/it]
               Class     Images     Labels          P          R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.26it/s]
                 all         58        386      0.339      0.368      0.
272       0.15

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     12/29     10.9G   0.04107   0.03301   0.03517    0.1092       163
640: 100% 13/13 [00:46<00:00,  3.55s/it]
               Class     Images     Labels          P          R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.09it/s]
                 all         58        386      0.264      0.429      0.
234     0.0715

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     13/29     10.9G   0.05665    0.0265   0.03527    0.1184       223
640: 100% 13/13 [00:41<00:00,  3.16s/it]
               Class     Images     Labels          P          R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.17it/s]
                 all         58        386      0.971      0.146      0.
202     0.0978

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     14/29     10.9G   0.04885   0.02844   0.03588    0.1132       141
640: 100% 13/13 [00:41<00:00,  3.21s/it]
               Class     Images     Labels          P          R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.30it/s]
                 all         58        386      0.546      0.275
0.28      0.148
```

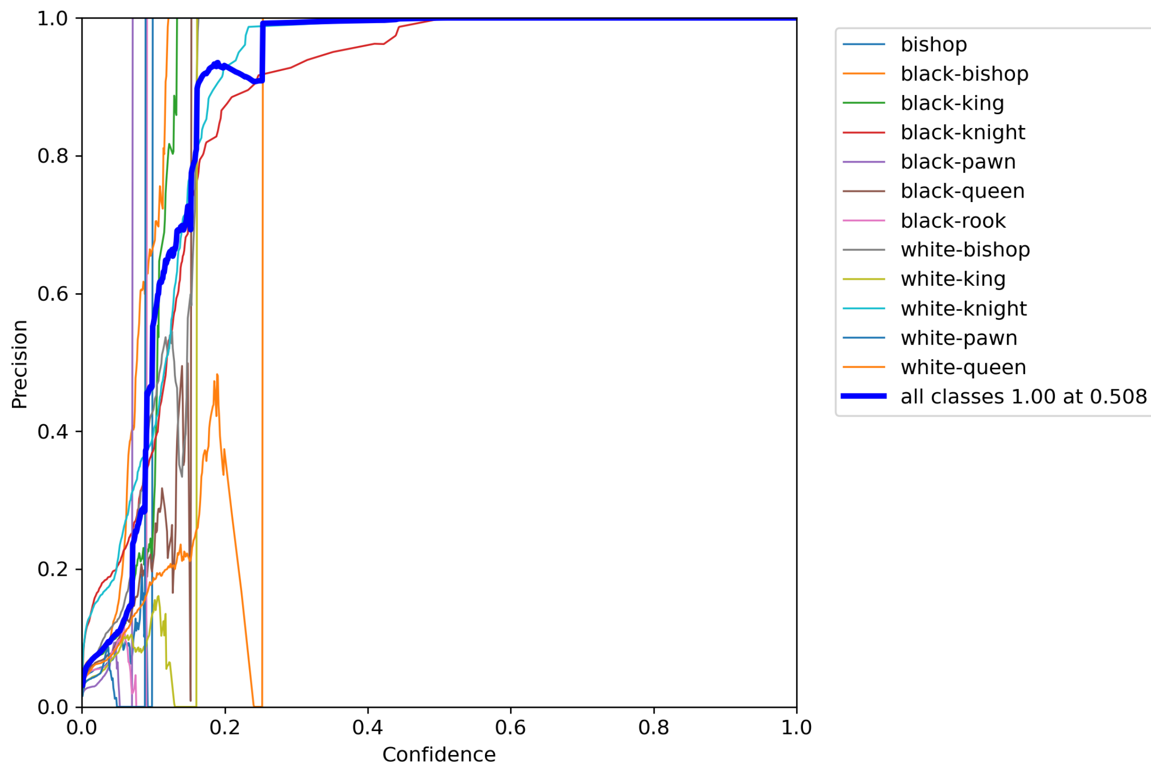
```
     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     15/29     10.9G    0.0465   0.03197   0.03545    0.1139       195
640: 100% 13/13 [00:46<00:00,  3.55s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.11it/s]
                 all          58         386       0.684       0.271       0.
234      0.0663

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     16/29     10.9G    0.0444   0.03241   0.03472    0.1115       172
640: 100% 13/13 [00:44<00:00,  3.45s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.10it/s]
                 all          58         386       0.456       0.488       0.
339        0.16

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     17/29     10.9G   0.04124   0.03679   0.03414    0.1122       186
640: 100% 13/13 [00:46<00:00,  3.58s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.17it/s]
                 all          58         386       0.301       0.372       0.
313       0.167

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     18/29     10.9G   0.03927   0.03395   0.03396    0.1072       112
640: 100% 13/13 [00:39<00:00,  3.05s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.05it/s]
                 all          58         386       0.266       0.434       0.
367       0.185

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     19/29     10.9G   0.04229   0.02978   0.03377    0.1058       191
640: 100% 13/13 [00:40<00:00,  3.12s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.20it/s]
                 all          58         386       0.997       0.161       0.
264       0.144

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     20/29     10.9G   0.04299   0.02566   0.03367    0.1023       133
640: 100% 13/13 [00:43<00:00,  3.32s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.17it/s]
                 all          58         386       0.494       0.261       0.
307       0.162

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
```

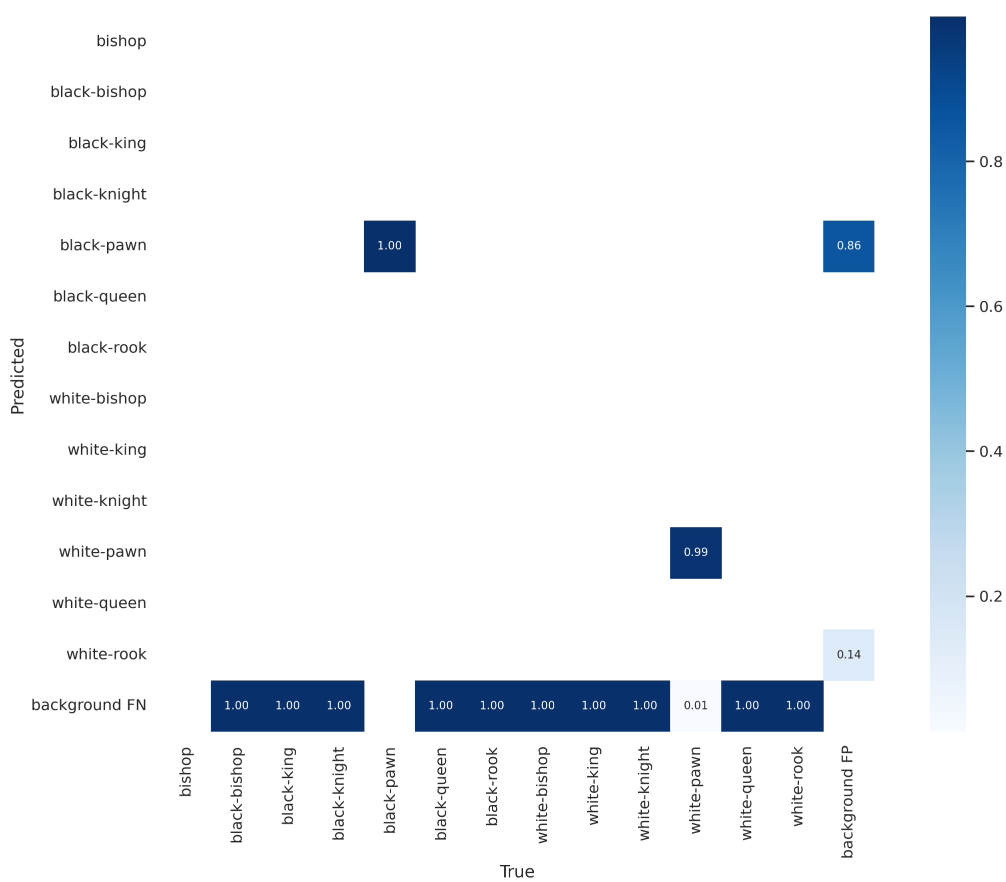
```
     21/29     10.9G   0.03875   0.03302   0.03315    0.1049       152
640: 100% 13/13 [00:44<00:00,  3.44s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.23it/s]
                 all          58         386       0.228       0.608       0.
359       0.201

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     22/29     10.9G   0.03694   0.03152   0.03262    0.1011       223
640: 100% 13/13 [00:45<00:00,  3.46s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.26it/s]
                 all          58         386        0.29       0.499       0.
388       0.239

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     23/29     10.9G   0.03453    0.0353   0.03211    0.1019       169
640: 100% 13/13 [00:42<00:00,  3.26s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.09it/s]
                 all          58         386        0.26       0.496       0.
397       0.228

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     24/29     10.9G   0.03372   0.03461    0.0318    0.1001       208
640: 100% 13/13 [00:45<00:00,  3.52s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.19it/s]
                 all          58         386       0.254       0.627
0.41       0.272

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     25/29     10.9G   0.03151   0.03497   0.03123   0.09772        94
640: 100% 13/13 [00:44<00:00,  3.45s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.17it/s]
                 all          58         386       0.258       0.567
0.4       0.24

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     26/29     10.9G   0.03143   0.03531   0.03146    0.0982       161
640: 100% 13/13 [00:53<00:00,  4.12s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.25it/s]
                 all          58         386       0.304       0.532
0.41       0.277

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     27/29     10.9G   0.02955   0.03537   0.03061   0.09552       274
640: 100% 13/13 [00:42<00:00,  3.25s/it]
               Class      Images      Labels           P           R      mAP
```

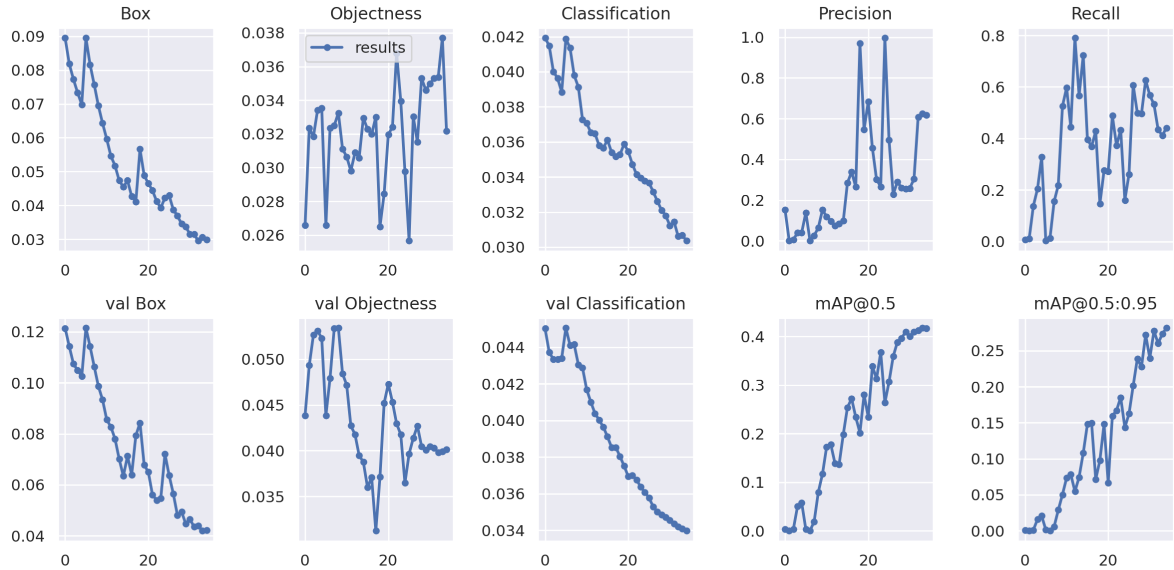
```
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.33it/s]
                 all          58         386       0.608       0.434       0.
413        0.26

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     28/29     10.9G   0.03062    0.0377   0.03067   0.09899       272
640: 100% 13/13 [00:47<00:00,  3.64s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:01<00:00,  1.19it/s]
                 all          58         386       0.626       0.411       0.
418       0.273

     Epoch   gpu_mem       box       obj       cls     total    labels  img_s
ize
     29/29     10.9G   0.02991   0.03217   0.03036   0.09244       141
640: 100% 13/13 [00:44<00:00,  3.43s/it]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 2/2 [00:03<00:00,  1.80s/it]
                 all          58         386       0.617       0.441       0.
417       0.282
        black-bishop          58          22           1           0       0.
117      0.0723
          black-king          58          29       0.756       0.552       0.
754       0.574
        black-knight          58          30       0.653         0.3       0.
452       0.206
          black-pawn          58          77       0.436           1       0.
995       0.684
         black-queen          58          11           1           0      0.0
964      0.0748
          black-rook          58          28       0.283        0.25       0.
243       0.132
        white-bishop          58          22           1           0
0.12       0.089
          white-king          58          29       0.488           1       0.
669        0.49
        white-knight          58          19       0.145        0.32       0.
129      0.0729
          white-pawn          58          77       0.449       0.987       0.
985       0.729
         white-queen          58          16           1           0       0.
091      0.0638
          white-rook          58          26       0.194       0.885
0.35       0.193
30 epochs completed in 0.463 hours.

Optimizer stripped from runs/train/yolov7-custom/weights/last.pt, 74.9MB
Optimizer stripped from runs/train/yolov7-custom/weights/best.pt, 74.9MB
```

In [134]:

```
from PIL import Image
pcurve = Image.open('/content/drive/MyDrive/Colab Notebooks/yolov7/runs/train/yolov7-custom/P_curve.png')
confusionmatrix = Image.open('/content/drive/MyDrive/Colab Notebooks/yolov7/runs/train/yolov7-custom/confusion_matrix.png')
results = Image.open('/content/drive/MyDrive/Colab Notebooks/yolov7/runs/train/yolov7-custom/results.png')

display(pcurve)
display(confusionmatrix)
display(results)
```


Precision
Confidence
bishop
black-bishop
black-king
black-knight
black-pawn
black-queen
black-rook
white-bishop
white-king
white-knight
white-pawn
white-queen
all classes 1.00 at 0.508


Predicted
True
bishop
black-bishop
black-king
black-knight
black-pawn
black-queen
black-rook
white-bishop
white-king
white-knight
white-pawn
white-queen
white-rook
background FN
background FP
1.00
0.86
0.99
0.14
0.01


Box
Objectness
results
Classification
Precision
Recall
val Box
val Objectness
val Classification
mAP@0.5
mAP@0.5:0.95

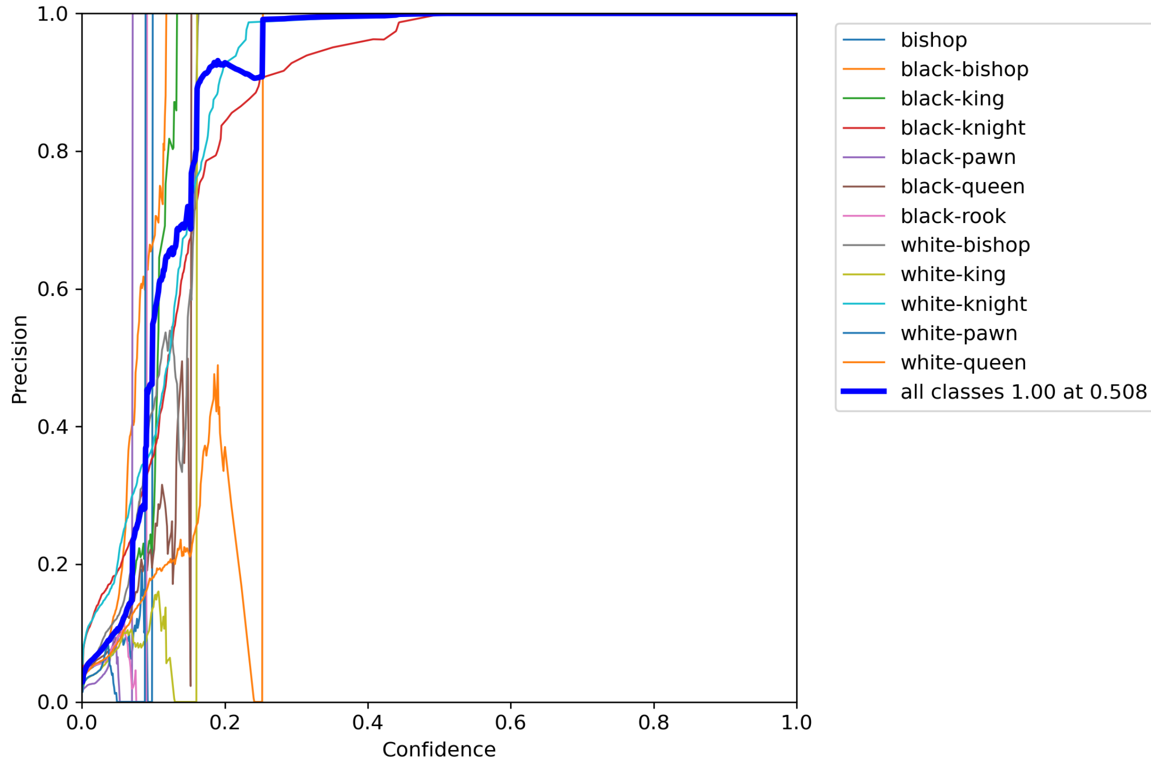
Testing

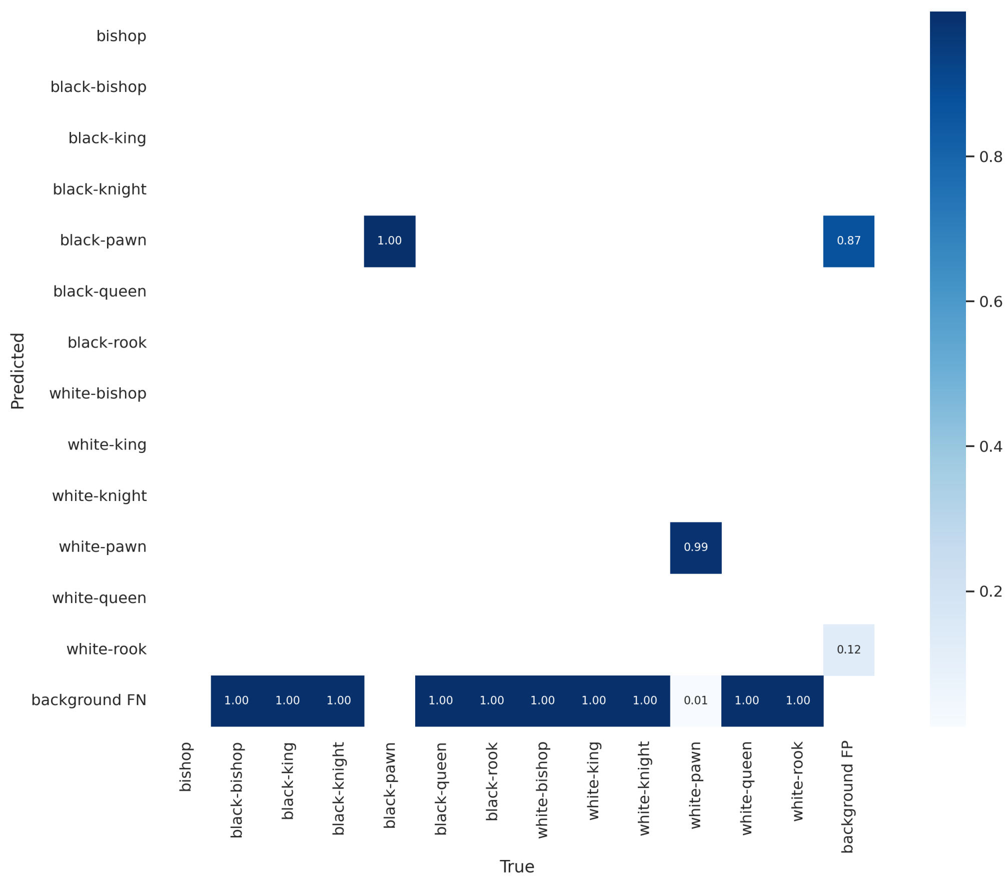
In [135]:
```
!python test.py --data "/content/drive/MyDrive/Colab Notebooks/Chess/data.yaml" --batch-size 16 --weights "/content/drive/MyDrive/Colab Notebooks/yolov7/runs/train/yolov7-custom/weights/best.pt" --exist-ok
```

```
Namespace(augment=False, batch_size=16, conf_thres=0.001, data='/content/driv
e/MyDrive/Colab Notebooks/Chess/data.yaml', device='', exist_ok=True, img_siz
e=640, iou_thres=0.65, name='exp', no_trace=False, project='runs/test', save_
conf=False, save_hybrid=False, save_json=False, save_txt=False, single_cls=Fa
lse, task='val', v5_metric=False, verbose=False, weights=['/content/drive/MyD
rive/Colab Notebooks/yolov7/runs/train/yolov7-custom/weights/best.pt'])
YOLOR 🚀 2022-12-5 torch 1.12.1+cu113 CUDA:0 (Tesla T4, 15109.75MB)

Fusing layers...
RepConv.fuse_repvgg_block
RepConv.fuse_repvgg_block
RepConv.fuse_repvgg_block
Model Summary: 306 layers, 36544546 parameters, 6194944 gradients
 Convert model to Traced-model...
 traced_script_module saved!
 model is traced!

/usr/local/lib/python3.8/dist-packages/torch/functional.py:478: UserWarning:
torch.meshgrid: in an upcoming release, it will be required to pass the index
ing argument. (Triggered internally at  ../aten/src/ATen/native/TensorShape.c
pp:2894.)
  return _VF.meshgrid(tensors, **kwargs)  # type: ignore[attr-defined]
val: Scanning '/content/drive/MyDrive/Colab Notebooks/Chess/valid/labels.cach
e' images and labels... 58 found, 0 missing, 0 empty, 0 corrupted: 100% 58/58
[00:00<?, ?it/s]
               Class      Images      Labels           P           R      mAP
@.5  mAP@.5:.95: 100% 4/4 [00:05<00:00,  1.41s/it]
                 all          58         386       0.612       0.439        0.
416       0.281
          black-bishop        58          22           1           0        0.
117       0.072
            black-king        58          29        0.75       0.552
0.76       0.578
          black-knight        58          30       0.651         0.3        0.
452       0.206
            black-pawn        58          77       0.418           1        0.
995       0.684
           black-queen        58          11           1           0      0.0
919      0.0707
            black-rook        58          28       0.283        0.25        0.
244       0.135
          white-bishop        58          22           1           0
0.12      0.0891
            white-king        58          29       0.488           1        0.
669        0.49
          white-knight        58          19       0.136       0.298        0.
129      0.0713
            white-pawn        58          77       0.429       0.987        0.
985       0.729
           white-queen        58          16           1           0      0.0
807      0.0561
            white-rook        58          26       0.194       0.885        0.
343       0.192
Speed: 18.8/5.1/23.9 ms inference/NMS/total per 640x640 image at batch-size 1
6
Results saved to runs/test/exp
```

In [138]:

```
pcurve = Image.open('/content/drive/MyDrive/Colab Notebooks/yolov7/runs/test/exp/P_curve.png')
confusionmatrix = Image.open('/content/drive/MyDrive/Colab Notebooks/yolov7/runs/test/exp/confusion_matrix.png')
#results = Image.open('/content/drive/MyDrive/Colab Notebooks/yolov7/runs/test/exp/results.png')

display(pcurve)
display(confusionmatrix)
#display(results)
```





The results here look terrible, *but* that was probably just because of the low number of epochs, which still took me forever to train. At first I tried it with 5 epochs, and it was even worse (I know, hard to imagine). With 300 epochs it would have probably been very good.

Detecting

Running sample images on the model on which we used transfer learning on.

In [139]:
```
!python detect.py --weights "/content/drive/MyDrive/Colab Notebooks/yolov7/run
s/train/yolov7-custom/weights/best.pt" --conf 0.25 --img-size 640 --source "/c
ontent/drive/MyDrive/Colab Notebooks/Chess/test/images/a3863d0be6002c21b20ac88
817b2c56f_jpg.rf.e421134b139d57e02e7df9468a35c1fb.jpg" --exist-ok
```

```
Namespace(agnostic_nms=False, augment=False, classes=None, conf_thres=0.25, d
evice='', exist_ok=True, img_size=640, iou_thres=0.45, name='exp', no_trace=F
alse, nosave=False, project='runs/detect', save_conf=False, save_txt=False, s
ource='/content/drive/MyDrive/Colab Notebooks/Chess/test/images/a3863d0be6002
c21b20ac88817b2c56f_jpg.rf.e421134b139d57e02e7df9468a35c1fb.jpg', update=Fals
e, view_img=False, weights=['/content/drive/MyDrive/Colab Notebooks/yolov7/ru
ns/train/yolov7-custom/weights/best.pt'])
YOLOR 🚀 2022-12-5 torch 1.12.1+cu113 CUDA:0 (Tesla T4, 15109.75MB)

Fusing layers...
RepConv.fuse_repvgg_block
RepConv.fuse_repvgg_block
RepConv.fuse_repvgg_block
Model Summary: 306 layers, 36544546 parameters, 6194944 gradients
 Convert model to Traced-model...
 traced_script_module saved!
 model is traced!

/usr/local/lib/python3.8/dist-packages/torch/functional.py:478: UserWarning:
torch.meshgrid: in an upcoming release, it will be required to pass the index
ing argument. (Triggered internally at  ../aten/src/ATen/native/TensorShape.c
pp:2894.)
  return _VF.meshgrid(tensors, **kwargs)  # type: ignore[attr-defined]
6 black-pawns, 5 white-pawns, Done. (17.9ms) Inference, (1.6ms) NMS
 The image with the result is saved in: runs/detect/exp/a3863d0be6002c21b20ac
88817b2c56f_jpg.rf.e421134b139d57e02e7df9468a35c1fb.jpg
Done. (0.323s)
```

In [141]:
```
pil_im1 = Image.open('/content/drive/MyDrive/Colab Notebooks/yolov7/runs/test/
exp/test_batch0_labels.jpg')
pil_im2 = Image.open('/content/drive/MyDrive/Colab Notebooks/yolov7/runs/test/
exp/test_batch2_labels.jpg')
display(pil_im1)
display(pil_im2)
```

```
Output hidden; open in https://colab.research.google.com to view.
```

Analysis

Since I have had problems with my originally chosen dataset, I will not be able to compare the Sequential/RNN with the pretrained yolov7 model.

Generally, I can say, after working for quite a few time with a pretrained model, such as the YOLOv7, and before that the YOLOv5, I enjoy this more than building a model from scratch/using tensorflows sequential models. Not only are pretrained models higher in accuracy (well, I couldn't prove it this time, but there's a reason as to why those are so popular), but also you will save time by just going with the (probably objectively) better choice. Especially, if you want to use features like opencv and detect objects in a real life setting with your camera or a video recording.

There is still the option to, as we have done it here, use transfer learning to use an existing model, like the YOLOv7, and "customize it" for your dataset. I have personally done this over the course of this semester, as I was part of the ACM Research. We used the FLIR thermal imaging dataset to make our own model (YOLOvCAPY), which works best for thermal images, especially when detecting objects while driving a car, which is what the dataset was made for.

For a general purpose like this here, the Sequential or RNN model seemed alright, but as soon as you want to expand, it seems impossible with these limited resources.