CHESS DETECTOR

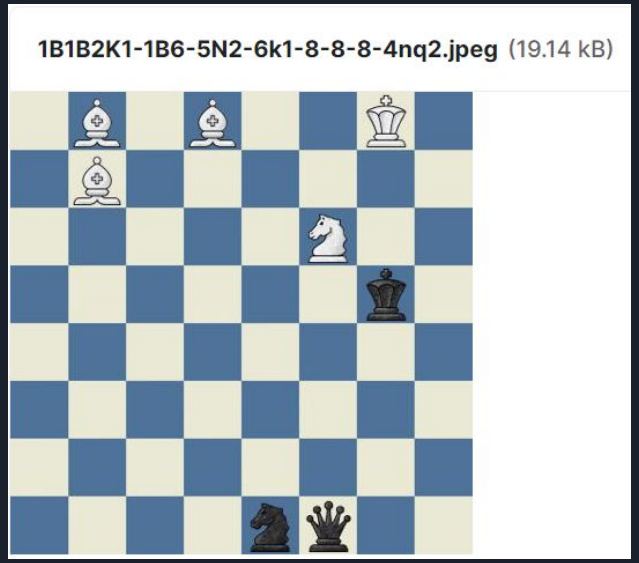
Andrea Morelli 1845525

Vision And Perception Project AY 2021/2022

CHESS DETECTOR

The scope of this project is to extract from a video or an image of a chess board the board pieces positions and save this configuration .

THE DATASET



The dataset that has been used in this project is :
https://www.kaggle.com/datasets/koryakinp/chess-positions
The input is a list of images and each image is labeled with a fem description of the chess pieces positions that has to be taken from the title.
In the left there is an example of an input image.
Every image has its own textures and theme.

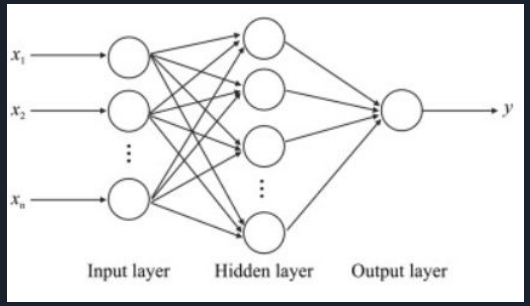
CHESS DETECTOR

I decided to divide the project in 2 parts. The first part of the project has to be able to retrieve the positions from an image that is seen from the top frontally.
The second part of the project has to accomplish the same task but with input images that are transformed using a homography transformation(also called perspective transformation), where the input chess board image is not seen frontally and there are some distorsions.

SOLVE NORMAL TASK

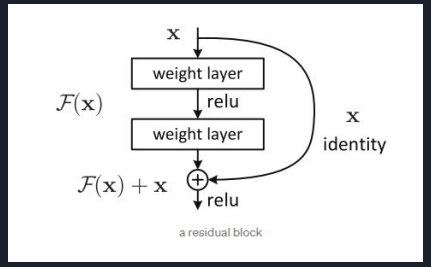
To solve the first task that i denoted as normal task the idea that can be applied is to divide the image in a 8*8 patches containing all the squares and then classify all those boxes in order to retrieve the chess board configuration.
To solve this part i created different types of networks and i monitored the performances of all of them with weights and biases.
The dataset is unbalanced so instead of the accuracy i had to use the f1 score using precision and recall combined.

NETWORKS ARCHITECTURES USED



The first used is a simple deep multilayer perceptron network with relus as activation functions.
This simple networks reaches good accuracy after some epochs but the performances are not optimal and not reliable.
I will show later the performances compared to the other layers

NETWORKS ARCHITECTURES USED



The second network i used is the resnet network that is a deep convolutional network where the main layers are the residual blocks. Those layers allows to mitigate problems like the gradient explosion or the vanishing gradient problem.

I did different types of tests:

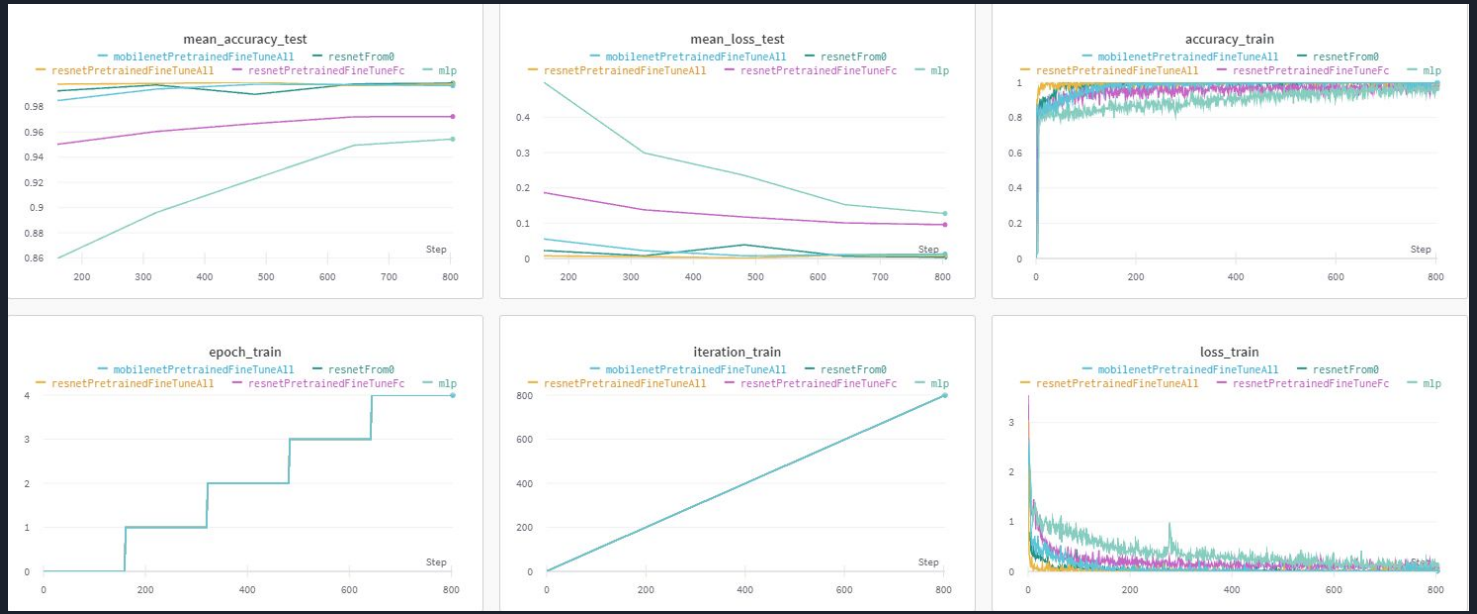
1) i trained the networks from scratch
2) i froze all the layers except the last and i trained just the fully connected layer for classification
3) i finetuned the entire pretrained neural network

NETWORKS ARCHITECTURES USED

The third network that i used is the mobilenetv3 network.
I used this network because is efficient in devices with not optimal performances. As the resnet network also this is heavily based on convolutional layers.

NORMAL TASK PERFORMANCES

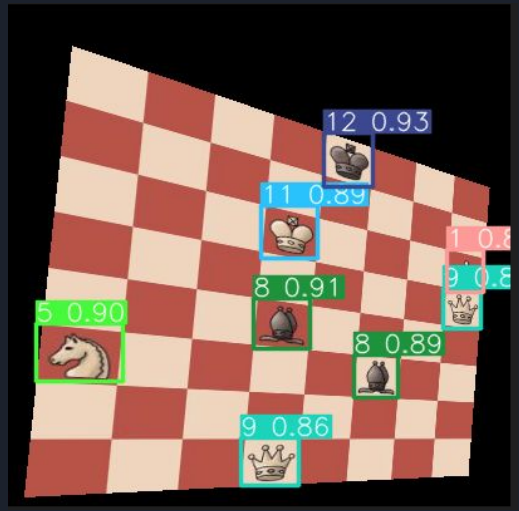
Those are the performances of the normal task using the networks described before. The mlp underperforms as expected and the resnet where just the last layer is trained is slightly better but still not optimal for the task.Maybe the features learned in the pretrained network are not useful for classifying the pieces.



NORMAL TASK DEMO

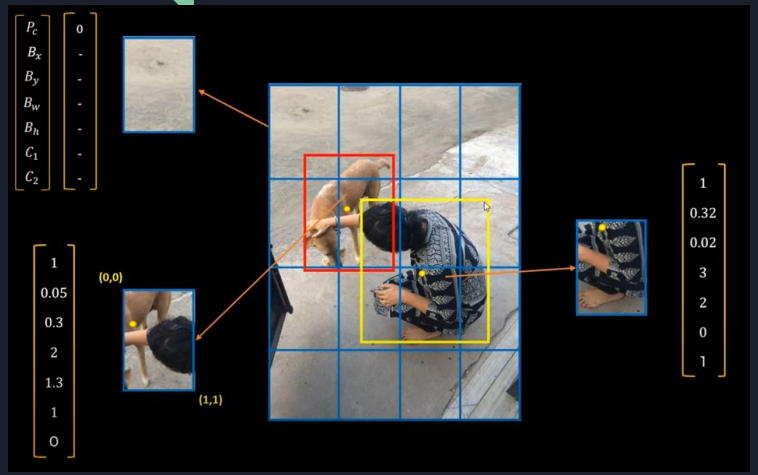
Let’s see the demo in action using the network on lichess and then try to use the model on the modified task.

SOLVE THE MODIFIED TASK



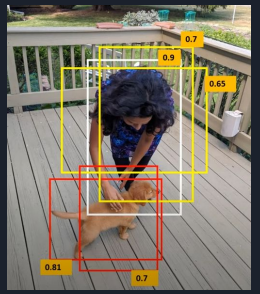
Now what if we want to solve the task of retrieving the board starting from an image that is transformed with an homography transformation?
The idea that i came up with is to change the task to an object detection one.In this way detecting the pieces with the bounding boxes will allow to get the centers of the objects and their class. Having the centeres i can retrieve the board positions

OBJECT DETECTION

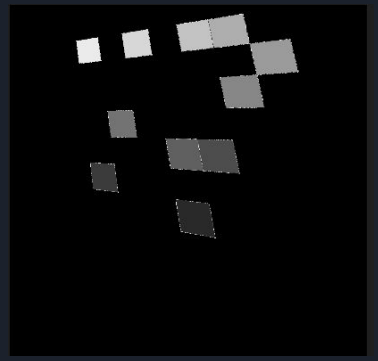


To solve the task of object detection i decided to train the neural network yolov5 (you only look once).
This neural network starting from an image (or a video because its batched) predicts a list of objects that are represented by a bounding box(centers,width and height),objectness score that describes the probability of an object being there(can be thresholded for the results) and a class probability.

To generate the results it uses different patches so it can predict the same object more than one time so form those it gets the intersection of the areas



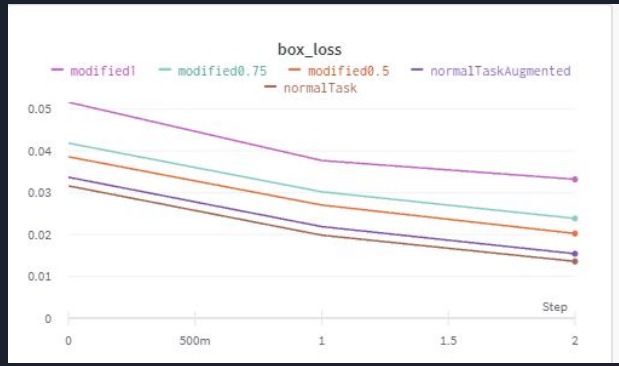
DIFFERENT VERSIONS OF THE DATASETS

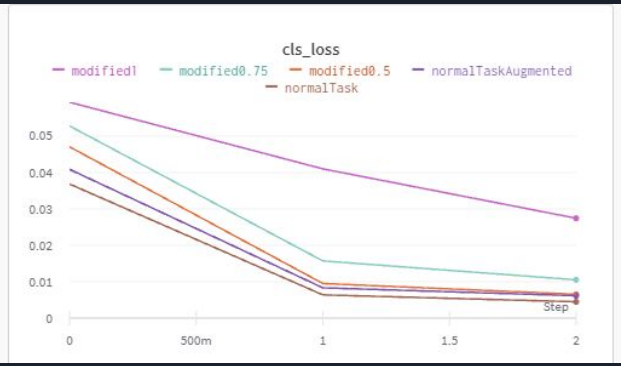


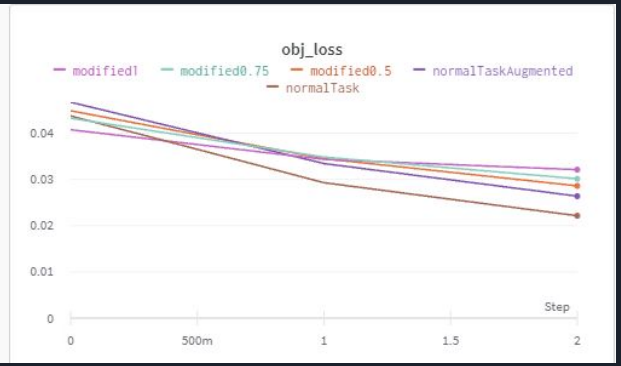
I generated different datasets with different distorsion coefficients in the homography and the images of the datasets have to be in a predifined input structure and all the objects have to be labeled in a specific format.
I wrote some scripts to modify the dataset and label the dataset.
To obtain the positions of the objects once the image is transformed i had to encode in the image the positions

DIFFERENT TRAINING OF YOLO

The training of the yolo network has been performed on different versions of the datasets based on the distorsions.
Here are the performances of the network:







CHESS DETECTOR

The yolo network took a lot of time to train on a gtx 1060 , for each dataset to complete 3 epochs it would take an entire day so the problem is in the time of training.
The network could have been trained longer because the validation loss was still decreasing but i decided to stop for the time it was taking.

The normal task instead was faster to train.

Both the types of tasks are performed with good performances so they could be used for real time applications.

THANKS FOR THE ATTENTION