Chessboard Recognition with CNN

By: Mason Rayburn

Abstract

Through the usage of computer imaging techniques, I produced the ability to detect the state of a chessboard and produce notation that describes said state. I utilized Convolution Neural Networks and transfer learning to complete this task and to generate models to detect classes of pieces. I think applied said models to test their accuracy. These models detected roughly the correct board fairly frequently and improved as more epochs were added in the training phase. This will allow for chessboards to be detected without built in chess logic software and can be improves to include pictures of real-life chessboards.

Introduction

Chess is one of the most defining games in the world. Its ruleset has changed very little and has been enjoyed by kings and queens of the past as well as modern day children in elementary school. Through the advent of computers and the internet, chess can be played between two people from across the word. The most prominent of services is a website and app called Chess.com.

Chess.com uses a standard board shape with a variety of different colors and piece shapes. However, if a computer was to try to analyze a picture of a state of the gameboard, how could it accurately tell? This is the problem my program will solve. Now, any chess simulator that uses other standard boards will be able to get an accurate Forsyth–Edwards Notation (FEN) without having the game logic behind it.

When I was first creating my program, I wanted it to accurately maneuverer the board, take apart each square of the board, and then check each square for what piece, or lack thereof, resides there. It will then produce an FEN number that is used by most chess programs. I also wanted the ability to test and display what the program believed each piece was. Therefore, I created a multistep process involving a Convolution Neural Network (CNN) to apply these goals.

Methodology

The start of this process is the preprocessing of the chessboards to testing. Since the boards are relatively the same shape, some minor filling to ensure that each image is the same size is all that is required. The board is then marked into 64 squares for the model to run. The before and after are shown and Figures 1 and 2.

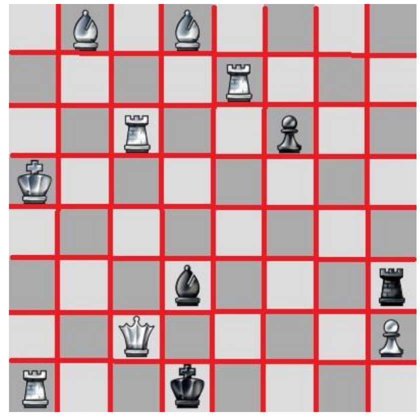

Figure 1. Figure 2.

In order to detect each piece, I decided to use a CNN model. This model produces 13 classes, each piece and their two possible colors as well as a class for an empty space. I trained this model on a variety of different images of chess pieces that are stylized for these Chess.com style games. Even with that, I decided that I didn't have enough data for this specific task, and you'll see the results of that later. Therefore, I applied transfer learning. I used the prebuilt ResNet18 model and froze everything but the bottom few layers.

Afterwards I applied the model on each square of the chessboard. Once every piece was complete, my program computes the FEN number associated with the board. It then begins the comparison to the test FEN to determine the accuracy of each square and whether or not it accurately produced the total FEN number. I then tested each on roughly 1000 images of chessboards and their FEN numbers.

Results

I produced four different models with different amounts of data and epochs when training. Model v0_5 was the first completed and had the least data to train with. I added more data for the others and models one through three have various epochs instead.

Model	Average Board Accuracy	Percentage of Correct FEN	Epochs
Resnet18_v0_5	51.94%	0.00%	10
Resnet18_v1	94.91%	25.67%	10
Resnet18_v2	93.03%	35.96%	15
Resnet18_v3	98.70%	61.63%	20

As shown in the table above, models one through three are the vastly superior to the earliest model, outpacing it in in both the average accuracy of each board and its rate of predicting each FEN, with version 0.5 failing to get even one correct FEN. As we focus on the next three models, their average board accuracy moves very little between versions one and two, actually going down between the two. Where it grows is its ability to fully complete a board correctly, proven even more so between versions two and three with a growth of 25.67. This means that by version three, the model is only making around one or two mistakes and it’s making a mistake one less than half the boards. The level of epochs clearly impacts how accurate each model is.

Discussion

At the onset of this project, I had the idea to, rather than take images from Chess.com, to use real life images of chessboards, warp them, extract the squares, then use the model. However, I had nowhere near the amount of data required to train a model even with transfer learning. I also couldn't find datasets that contained the FEN number required to test the model. Therefore, I had to change what I was looking for and it went much better.

Even with the change to a Chess.com style, I still had to manually grab data of chess pieces. This is the data that is added after the 0.5 version. As you can see, it greatly increased the accuracy.

Transfer learning using Resnet18 greatly improved both the speed and accuracy of the model, especially after I tuned it post version 0.5. If I was to expand the model, I would continue to use transfer learning, especially with the lack of data I could find.

Overall, most of my struggles can be attributed to data and finding data. I even began working on a synthetic data generator to produce varying colors and orientations of my pieces however I never ended up using it.

Conclusion

Overall, I learned that through the usage of transfer learning with models such as Resnet18, the creation of accurate and efficient models becomes much easier and requires less data collection. I think the next steps would be to apply this to actual pictures of chessboards and real life setting and I think I could do it with more data and test images. In conclusion, this project gave me a better understanding of the high-level techniques, such as the usage of CNNs and transfer learning, that is common practice in computer imaging today.