Machine Learning Engineer Nanodegree

Capstone Proposal

Jan Sulaiman 24.03.2019

Proposal

Domain Background

Supervised learning, specifically deep learning is a growing field in machine learning that is being used to tackle a range of challenging problems. One area in specific that deep learning, specifically neural networks, is expanding to is computer vision. In my opinion and also much research indicated that object recognition is a critical area of computer vision. It is essential for various applications like self-driving cards, facial recognition or estimation orientation of objects for example in an assembly line. Convolutional neural networks have helped revolutionized this field, replacing previous computer vision methods with more advanced techniques based on supervised learning. (Computer Vision Wikipedia, n.d.) Some challenges do present themselves as now a training requirement is added to image processing, but for a large portion of applications that data is present or can be obtained.

Problem Statement

In modern chess players use hours of video and other material to revisit positions from all kinds of previously played games. This is done primarily done from memory or written notations. Depending on the time formats recording of all moves in a game is not always necessary or just not easy to do. An automated method for capturing game moves or game state would help dramatically. Players would be able to obtain log files afterward that transcribed game moves and board positions at each state, allowing an in-depth study and review of the steps without any transcription errors. Tools which can do similar things already exist but are based on initial board state and not allow chess variants like Chess960 or practicing tactics problems or even end-game setups from known positions. This kind of system is based on a classification problem to identify all the pieces on board correctly. Chess piece identification has been explored in an academic setting using classical computer vision techniques and CNN.

Datasets and Inputs

The dataset used for the project is based on the Chess Board Data available on kaggle: https://www.kaggle.com/dcuomo56/chess-board-data It contains images of a chess board from an above vertical position. All pieces are randomly positioned on the board, always a full set of pieces and moved to ensure each type of piece is at least in each square once. There are a total of 1800 black and white individual square images in the dataset. Each square has a size of 135x135. The split of the dataset is 60% for training, 20% for validation and 20% for testing. The available images will be split into 64 individual photos each representing a square on the 8x8 chess board. These images will then be identified either as a dark or light empty square, or a dark or light pawn, knight, bishop, rook, queen or king. These inputs will then be used to help train the classifier to determine if a piece is present in a square and what part is current.

Solution Statement

For solving the described problem, CNN will be created and trained to accurately identify the stat of a square on a given chess board. If the algorithm is presented a full board, it will examine each square separately and return a digital representation of the board. I will experiment with a custom CNN architecture as well as a transfer learning approach to evaluate both performances.

Benchmark Model

As a benchmark, a simple CNN agent will be used, which most likely performs quite bad, given the number of possibilities present for each square, so the trained CNN should outperform it by quite a lot. The simple agent will be made up of three convolution layers and final dense layer for the classes.

Evaluation Metrics

For evaluation, two key metrics will be used in the assessment of the model. The first will be Keras absolute accuracy, and it will check that the maximal predicted value probability lines up with the categorical label provided with the agent across all classes. At its core, this is an accuracy problem, as failure to identify a piece right would make an effort unusable. Another key evaluation metric that will be used during traing is categorical cross entropy. It is comparable to log loss, and the loss increases as the predicted probability diverge from the truth label. It should help me to prevent that the model is favoring the classed that have more training points available for example white squares.

Project Design

The following software will be used: Python 3 w/ Numpy, MatplotLib, OpenCV and Keras

The flow is made up five steps: Image input which will be given to the image preprocessor then to the CNN agent and followed to the output generator. In the end, there will be an array as output.
A class will take care of the first image preprocessing. It will split each image into the 64 separate pieces and save them as a dictionary with a default value. Next, it will be sent to a CNN agent, the CNN agent assigned a label to each piece by updating the default value. It is followed by the output generator which will take care of generating the digital interpretation of the chess board and generate a matching output file.

The complete setup has two different modes, training and prediction. The three main classes will have all key function as outlined here:

1. Image Preprocessing:

- Training: Split of images into 64 separate images. Display of pictures and ask a user for input on a piece. An image will be saved to a separate folder corresponding to the piece.
- Prediction: Spit of images into 64 different images and store them in a dictionary with a default value. The output than used by CNN agent.

2. CNN Agent:

- Training: Given a set of training, test and validation images containing separate images. Use of Keras to train CNN, two different models will be checked, custom CNN as well as transfer learning approach. The best weights will be exported as a file for prediction
- Prediction: Use exported file from training and take sperate images from a split chess board to predict piece in a square.

3. Output Generator:

- Training: Output of accuracy and other metrics for CNN agent.
- Prediction: Output of digital representation of chess board.

Splitting of images is done multiple steps. Start with using OpenCV build in algorithms to find inner corners of chess board followed by extrapolation of edges. Then perspective transformation to ensure its only a square image which shows the full chess board. Splitting into 64 black and white photos each being 135x135 pixels big. The individual pieces can be used for CNN for training or prediction depending on the mode.