NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

SCSE20011 – Deep Learning in Chess

Presented by [redacted]

Supervised by Assoc Prof He Ying

1. Background

Computer chess is one of the traditional computer science research fields regarding AI. Among all the components of a computer chess engine, the evaluation function is one of the most crucial one. The function takes the position as input and gives a numeric representation as result to indicate which side is leading at the current position. Two methodologies have been used to develop the evaluation function. The traditional method evaluates the position using numerous hand-crafted features with long-time tuning. While the neural-network-based method trains and uses a NN as the evaluation functions.

2. Objective and Data Description

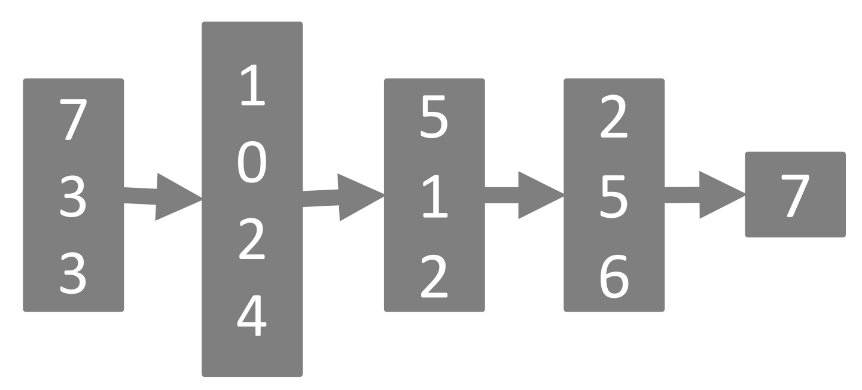
This project aims to develop a NN for the static evaluation function (i.e., without looking ahead) of the computer chess engine. First, we purchase the commercial chess dataset *ChessBase 16* as our training data. To be specific, we randomly sample 5 million distinct positions played by the strong chess players (i.e. ratings > 2200). Then, we label them by Stockfish 13. Third, we represent the positions based on different criteria for different NNs. Then, we train several NNs for classification and regression tasks.

3.1.1 Trivial fully connected NN

We convert the position to the bitboard representation with some additional features like the moving side and the castling right. We get 773 dimensions vector as our input.

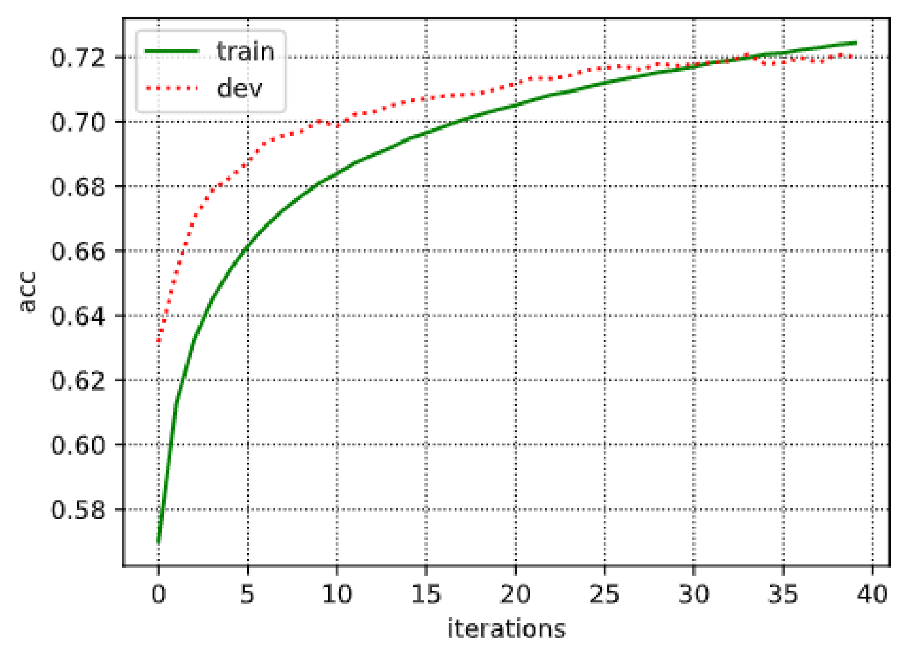
3.1.2 Architecture

The number in each block shows the dimensions of the fully connected layer. We simply reduce the dimensions from 733 to 7.



3.1.3 Training Result

The accuracy converts to 72% after 40 epochs

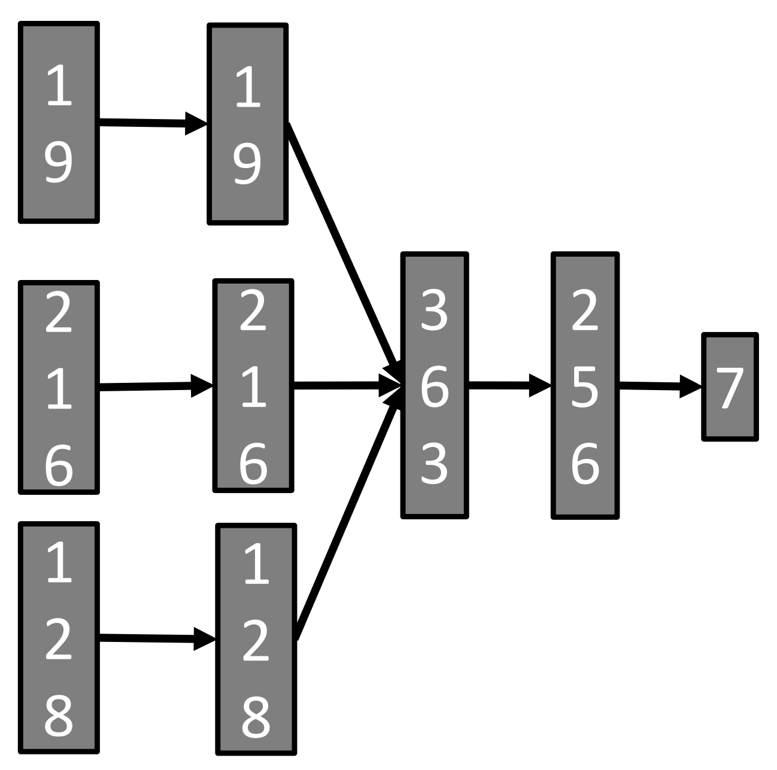


3.2 Manual-feature fully connected NN

Instead of using the above 773 dimensions trivial bitboard representation, we convert the position into 363 dimensions man-craft representation as our input based on our chess domain knowledge which could be grouped into global features, piece-centric features and square-centric features.

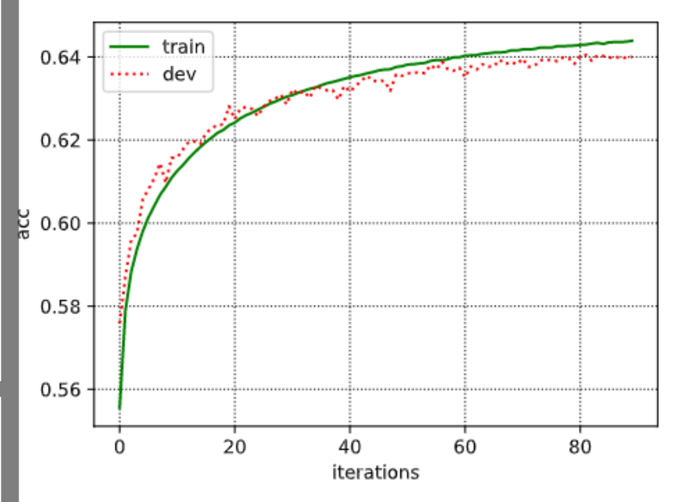
3.2.2 Architecture

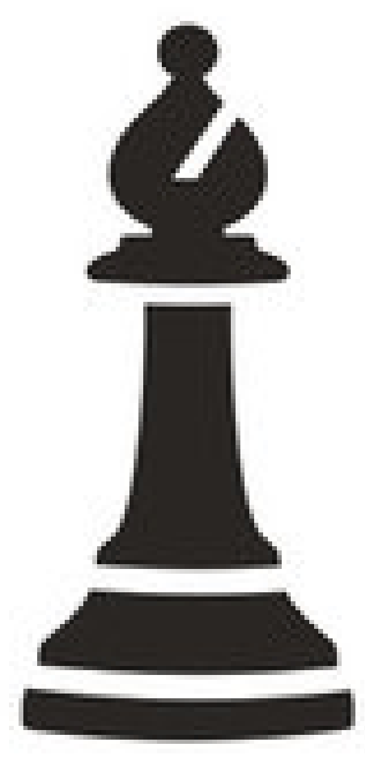
In the first two layers, features from different modalities are kept separate. Then, we combine them in the following fully connected layer.



3.2.3 Training Result

The accuracy converts to 64% after 90 epochs.



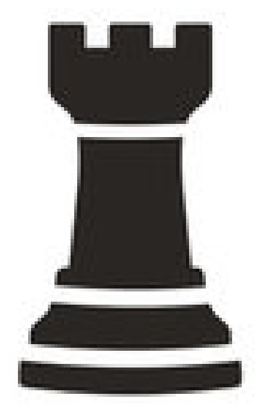


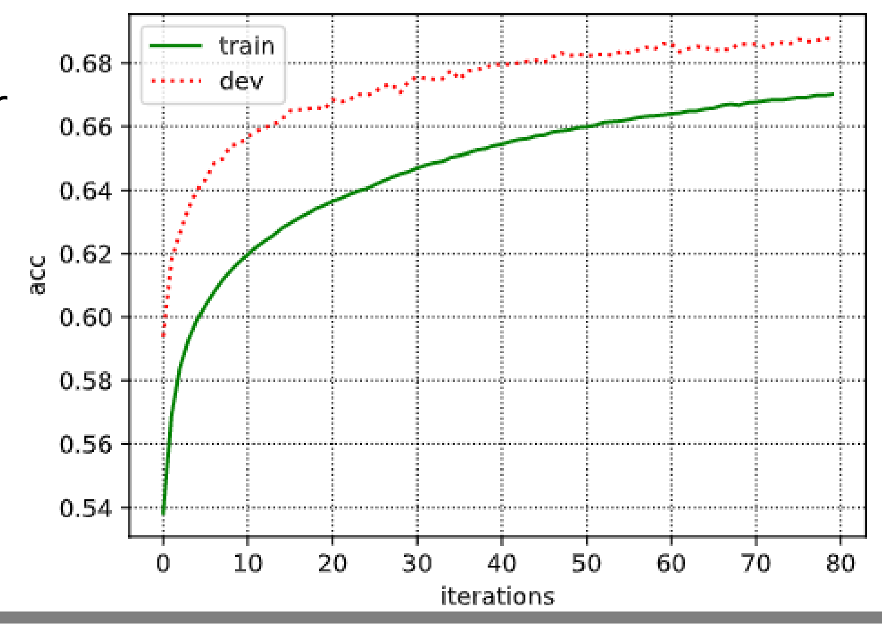
3.3.1 Autoencoder

We first apply unsupervised training to an autoencoder for reducing the above 773 dimensions sparse vector to the 100 dimensions which contains high level features. Second, we use the parameters of the encoder part as our initialization of the classifier NN.

3.3.3 Training Result

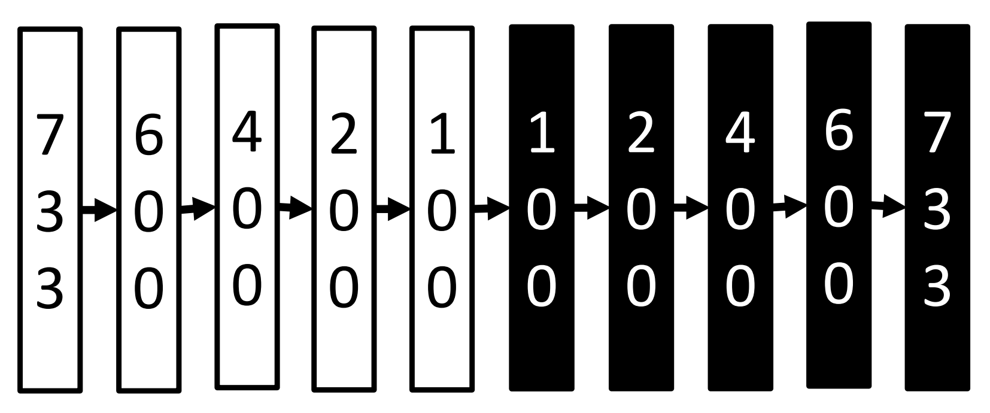
The accuracy converts to 68% after 80 epochs.



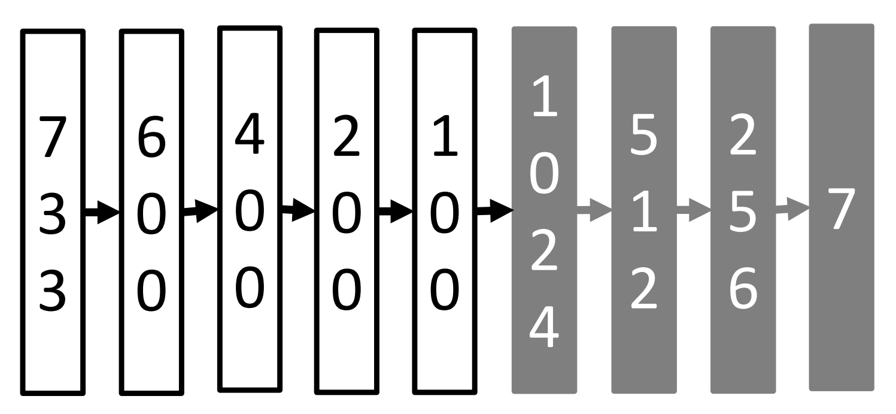


3.3.2 Architecture

The above shows the architecture of the autoencoder. The encoder part is painted white and decoder part is black.



The below shows the architecture of the classifier which takes the encoder part as its initial parameters. The classification part is painted gray.



4. Future development

Currently, the accuracy or loss of both models have room for further development. Due to hardware restriction, it is impossible to add in more training instances. But we still could change the architectures and fine-tune the hyperparameters like the learning rate and the optimization algorithm.