Play chess with Deep Learning

Thomas MICHEL and Gabriel JEANTET

ENS Paris-Saclay

January 28, 2022

Chess engine

Definition

A chess engine is a computer program that analyzes chess positions, and generates a move or list of moves that it regards as strongest.

Classic approach:

- Exploration of the configuration tree (minimax algorithm)
- Handcrafted evaluation function
- $\alpha\beta$-pruning to increase the exploration efficiency

Chess engine

Recent new approaches:

- Deep reinforcement learning: A neural network is trained to give a score to each action based on previous experiences, essentially through self-play
- Some of the best engines still use the minimax algorithm coupled with an efficient neural network architecture called NNUE as the evaluation function

Popular chess engines :

- Stockfish
- Leela Chess Zero
- Komodo
- Houdini

Our approach

- Approximate a chess engine evaluation function with a neural network model
- Depth 1 evaluation
- Non determinism

Next move choice procedure

INPUT: Current board state and current player's color

1. Evaluate the score of all the valid moves according to the model
2. Eliminate the 75% worst moves
3. Choose randomly a move from the remaining ones with probability weighted by their scores.

Dataset generation

The datasets are based on the database of games : www.pgnmentor.com/files.html

We chose to generate our datasets from the N first games played for each opening in the order of decreasing ELO rating of the players.

Advantages :

- Great variety of early game position
- Position obtained from high level tactics

Drawbacks :

- No control over the distribution of endgame positions
- Some rookie mistakes might not be covered by the dataset

Dataset generation

For every game of the dataset :

- Extract all positions from the game
- Analyze the positions using Stockfish
- Encode the positions and set their respective evaluation scores as targets
- Add the new data points to the dataset and remove duplicates using a hash table.

We built two datasets due to constraint on the generation time:

- Small one with 134629 positions analyzed up to depth 15 by Stockfish
- A bigger one with 595235 positions analyzed up to depth 6

The number of positions depended on the number of games considered for each opening.

Encoding for the MLP

To each chess piece type (12 in total) we associated a 8×8 bitmap representation of the board with 1 if this piece type is present at a given position, else 0.

To this encoding we added some metadata to fully represent the state of the game:

- Castling rights for both black and white
- A bit to represent if there is an active check
- The current turn number

Total encoding dimension : $8 \times 8 \times 12 + 6 = 774$

First model – MLP-based

INPUT : The flattened encoding of the game state

OUTPUT : Evaluation of the game state (scalar)

ARCHITECTURE :

- Fully connected layers
- Three hidden layers of size 2048
- ReLU activations
- Batch normalization added for some experiments

TRAINING :

- MSE loss
- ADAM optimizer
- Batch size $= 256$
- Fixed learning rate $= 0.001$
- Fixed weight decay $= 0.001$

Second model - CNN based

INPUT : Game encoding considered as a 8×8 image with 12 channels and an additional metadata vector of size 6

OUTPUT : Evaluation of the game state (scalar)

ARCHITECTURE :

```
CNN_Net(
  (conv1): Conv2d(12, 128, kernel_size=(5, 5), stride=(1, 1))
  (conv2): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1))
  (fc1): Linear(in_features=1030, out_features=2048, bias=True)
  (fc2): Linear(in_features=2048, out_features=1, bias=True)
)
```

TRAINING :

- MSE loss
- ADAM optimizer
- Batch size = 256
- Fixed learning rate = 0.001
- Fixed weight decay = 1e-5

Deep learning models evaluation – Training

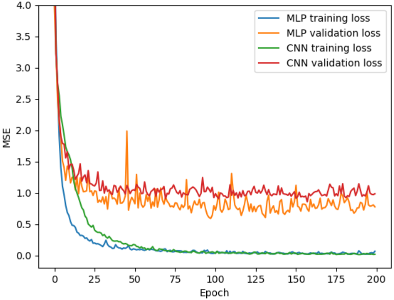

Figure: Training of the models on the smaller dataset for 200 epochs

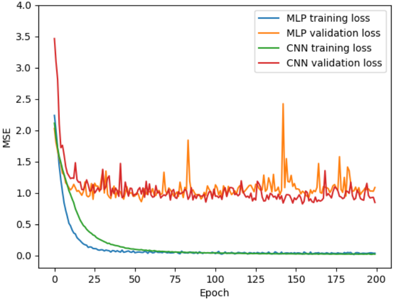

Figure: Training of the models on the bigger dataset for 200 epochs

Deep learning models evaluation – Playing

No win or draw against Stockfish $\rightarrow$ The models can not be compared to Stockfish ELO rating

Comparison of the two models in a 2000 rounds tournament (1000 rounds as white and 1000 as black)

	As white	As black	Total
MLP-based model victories	252	115	367
CNN-based model victories	148	40	188

$+$ 1445 Draws

Which corresponds to a ELO difference of 31 in favor of the MLP-based engine.

Conclusion

- Working chess engines
- Need to find an adapted benchmark to better evaluate the model against chess engines with known levels
- Architectures could be more specialized

References

- Sabatelli, M., Bidoia, F., Codreanu, V., Wiering, M. A. (2018, January). Learning to Evaluate Chess Positions with Deep Neural Networks and Limited Lookahead. In ICPRAM (pp. 276-283).
- David, O. E., Netanyahu, N. S., Wolf, L. (2016, September). Deepchess: End-to-end deep neural network for automatic learning in chess. In International Conference on Artificial Neural Networks (pp. 88-96). Springer, Cham.