**AI and ML:**

(Wikipedia, Artificial Neural Network, 2020)

(Wikipedia, Reinforcement Learning, 2020)

**Minimax:**

(Wikipedia, Minimax, 2020)

(Wikipedia, Alpha Beta Pruning, 2020)

**Evaluation Function:**

(Wikipedia, Chess Piece Relative Value, 2020)

(Wikipedia, Evaluation Function, 2020)

(ChessProgrammingWiki, Evaluation, 2020)

(ChessProgrammingWiki, Simplified Evaluation Function, 2018)

Using <https://chessboardjs.com/> to display the board.

Using <https://github.com/niklasf/python-chess> for the chess library.

Using <https://flask.palletsprojects.com/en/1.1.x/> for REST and connecting between JS and Python.

Piece Values:

* Pawn: 1
* Knight: 3
* Bishop: 3
* Rook: 5
* Queen: 9

Note:

* Encourage the engine to have the bishop pair.
* Stick to human chess experience.
* Bishop is always worth more than 3 Pawns
* Knight is always worth more than 3 Pawns
* Rook is always worth more than 5 Pawns
* Sometimes Bishop is worth more than a Knight
* Bishop + Knight > Rook + Pawn
* Rook + 2 Pawn > Bishop + Knight > Rook + Pawn
* Bishop + Knight = Rook + 1.5 Pawn
* Queen + Pawn = 2 Rook

Therefore:

* Bishop > Knight > 3 Pawns
* Bishop + Knight = Rook + 1.5 Pawn
* Queen + Pawn = 2 Rook

Also; Bishop + 2 Pawn > Knight + 2 Pawn > Rook

Pawn structure:

* Isolated
* Doubled
* Backward
* Advanced
* Passed
* protected passed
* connected passed
* holes
* semi-open and open files
* pawn majorities
* phalanxes
* Other formations

Factors to consider in evaluating the board:

* Development of the minor pieces
* Rooks on open files or the seventh rank
* Doubled rooks
* Outpost knights (knights in central locations protected by a pawn and not subject to attack by an opposing pawn)
* Possession of the bishop pair
* Bishops on the long diagonals
* Pieces occupying or bearing on spaces around the opposing king
* Mobility of the kings (kings shouldn't be 'cramped', hence subject to mate-on-the-move).

**Policy Function:**

(Wikipedia, Bellman Equation, 2020)

**Monte Carlo Tree Search:**

Selection – A random move (node) is chosen.

Expansion – Game will expand from that chosen move (node).

Simulation – Different simulations will be played.

Backpropogation – The chosen node is updated by going up the nodes.

(Wikipedia, Monte Carlo Tree Search, 2020)

Exploration – The program will choose a unexplored nodes move that couldve missed an important move.

Exploitation – The program will choose a random move that could be a mistake to find a better move in the end.

(ChessProgrammingWiki, UCT, 2020)

**Useful Links:**

<https://www.quora.com/How-does-Leela-Chess-work-What-are-its-neural-network-inputs-and-output>

<https://stackoverflow.com/questions/46260775/what-is-a-policy-in-reinforcement-learning>

<https://www.youtube.com/watch?v=ikDgyD7nVI8>

**Papers:**

(DeepMind, A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play, 2018)

(DeepMind, Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm, 2017)

(DeepMind, Assessing Game Balance with AlphaZero: Exploring Alternative Rule Sets in Chess, 2020)

# References

ChessProgrammingWiki. (2018, May 16). *Simplified Evaluation Function*. Retrieved from Chess Programming Wiki: https://www.chessprogramming.org/Simplified\_Evaluation\_Function

ChessProgrammingWiki. (2020, August 5). *Evaluation*. Retrieved from Chess Programming Wiki: https://www.chessprogramming.org/Evaluation

ChessProgrammingWiki. (2020, July 16). *UCT*. Retrieved from Chess Programming Wiki: https://www.chessprogramming.org/UCT

DeepMind. (2017). *Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm.* London: DeepMind.

DeepMind. (2018, Dec 07). *A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play.* London: DeepMind. Retrieved from https://science.sciencemag.org/content/362/6419/1140

DeepMind. (2020). *Assessing Game Balance with AlphaZero: Exploring Alternative Rule Sets in Chess.* London: DeepMind.

Wikipedia. (2020, September 1). *Alpha Beta Pruning*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Alpha-beta\_pruning

Wikipedia. (2020, October 12). *Artificial Neural Network*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Artificial\_neural\_network

Wikipedia. (2020, October 14). *Bellman Equation*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Bellman\_equation

Wikipedia. (2020, September 17). *Chess Piece Relative Value*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Chess\_piece\_relative\_value

Wikipedia. (2020, September 10). *Evaluation Function*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Evaluation\_function

Wikipedia. (2020, October 13). *Minimax*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Minimax

Wikipedia. (2020, September 26). *Monte Carlo Tree Search*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Monte\_Carlo\_tree\_search

Wikipedia. (2020, September 26). *Reinforcement Learning*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Reinforcement\_learning