Ingeniería Matemática
FACULTAD DE CIENCIAS
FÍSICAS Y MATEMÁTICAS
UNIVERSIDAD DE CHILE

Crimson Red

Ricardo Ziegele A. Luciano Villarroel S.

20 de diciembre de 2023

Facultad de Ciencias Físicas y Matemáticas

Contenidos

1. Introducción

2. Proyecto

3. Resultados

4. Conclusiones

Introducción

Introducción

- El Ajedrez es uno de los juegos de mesa más populares en la actualidad
- Más de 600 millones de jugadores a nivel mundial

Introducción

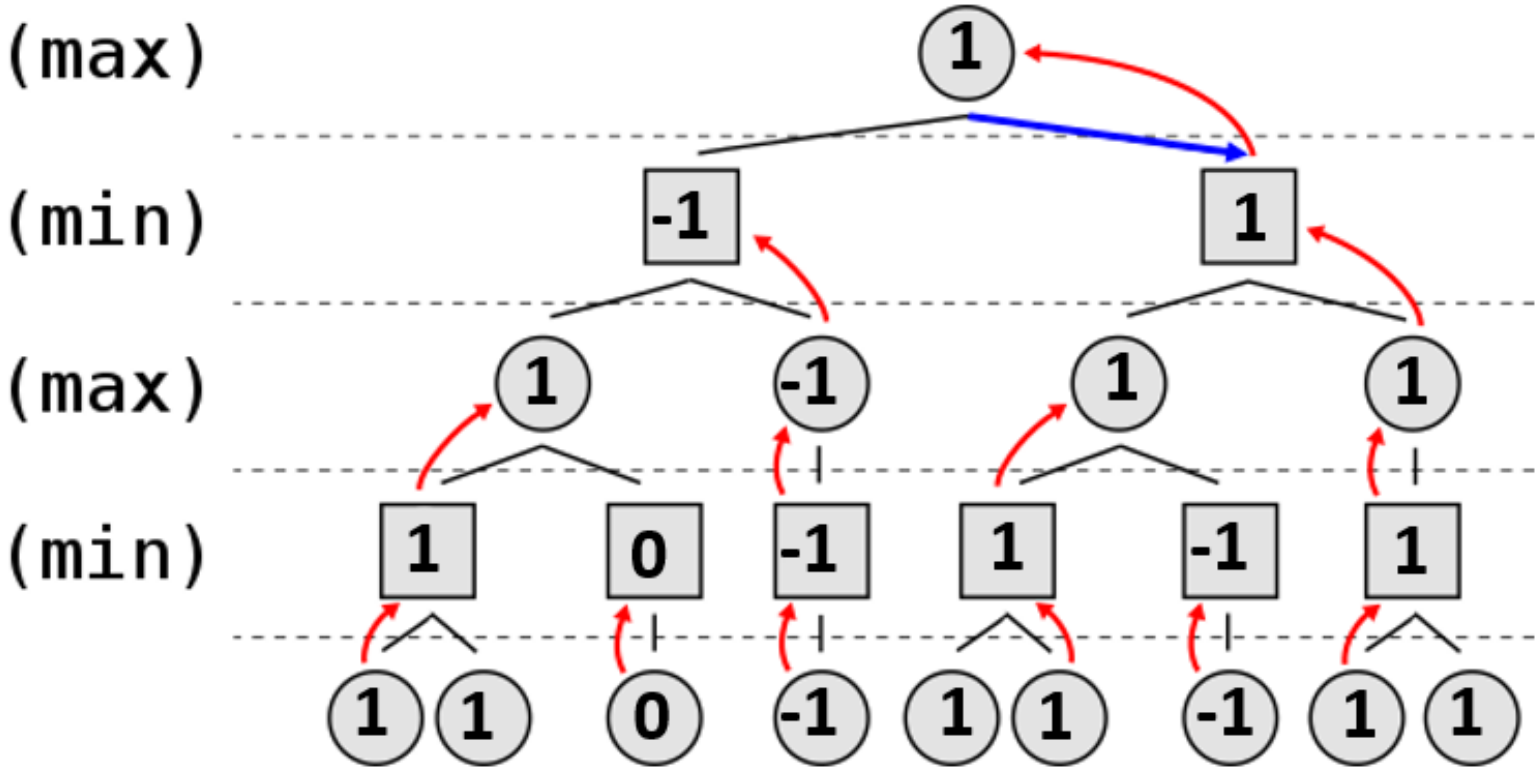
- 1997 la computadora *Deep Blue* venció a Gary Kasparov, marcando un hito en la historia del ajedrez.
- Hoy en día *StockFish* corriendo en un teléfono es capaz de ganarle a cualquier jugador/a de ajedrez.

Con esto como motivación, surgió nuestro proyecto de crear nuestro propio motor de ajedrez

- Crimson Red
  - Aprender sobre el funcionamiento de los motores de Ajedrez
  - Profundizar en los aspectos prácticos de Redes Neuronales

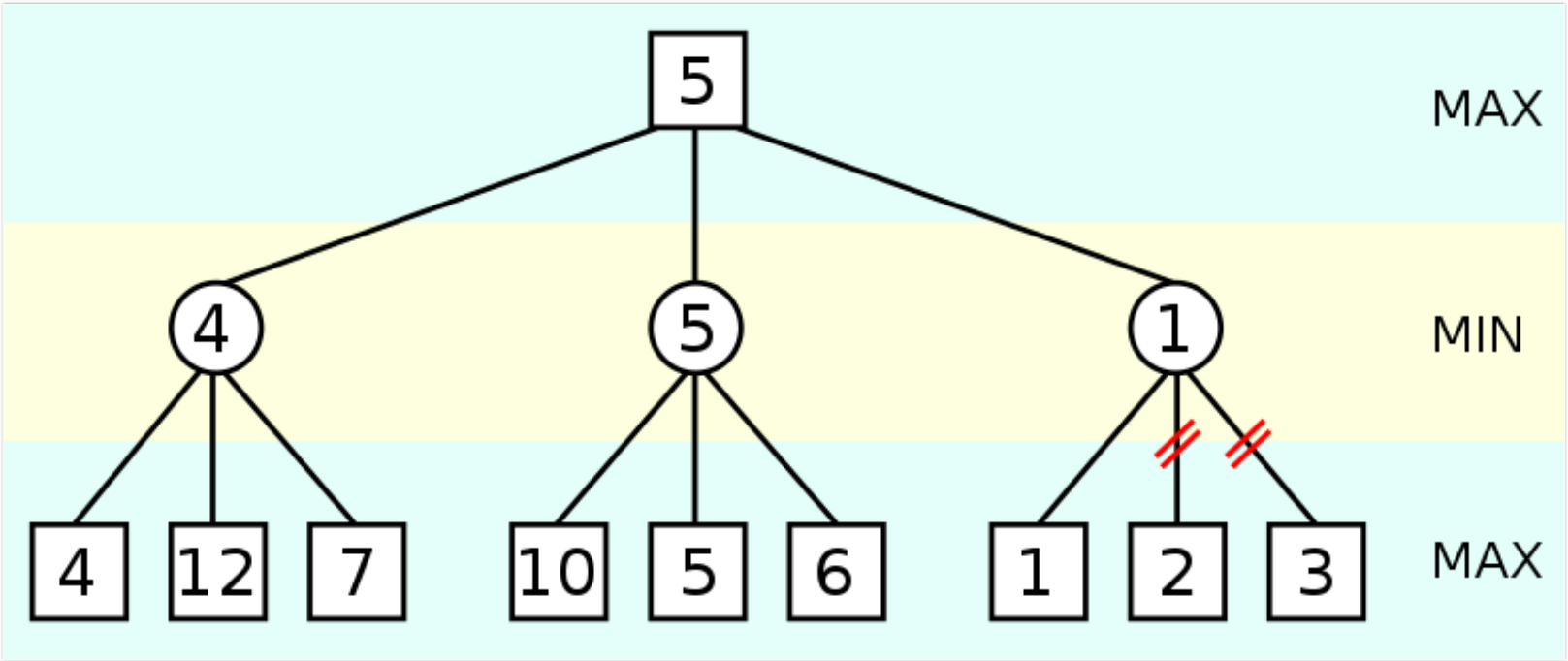
StockFish

- Minimax



StockFish

- *α-β-prunning*



Crimson Red

Con este algoritmo de búsqueda, un motor de ajedrez consiste básicamente en una función de evaluación.

La idea para crear el motor **Crimson Red** es:

- Entrenar una red neuronal como función de evaluación
- Utilizar α-β-*prunning*

Proyecto

Idea general

La idea general consiste en, dada una configuración de piezas en el tablero, obtener una calificación cuantitativa (numérica) de cuán favorable es dicha posición, y para quién lo es.

Codificar una posición

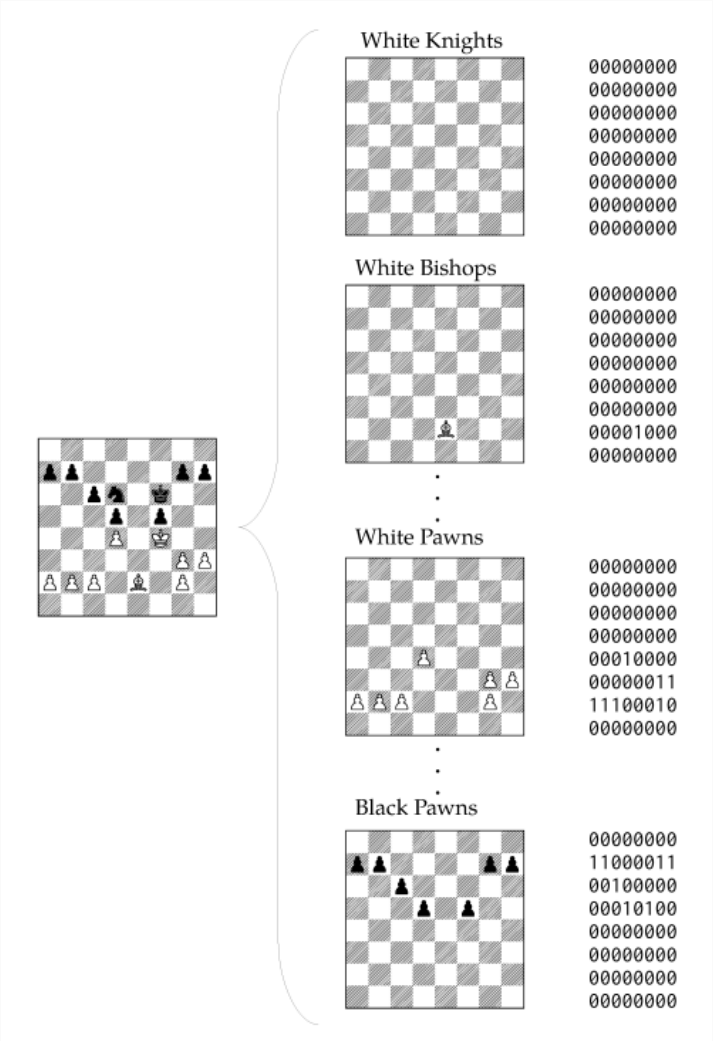


Figura 1: Figura adaptada de *Neural Networks for Chess* (Klein, 2022)

Input y output de Crimson Red

El **input** para el algoritmo se compone de un tensor de $17 \times (8 \times 8)$, donde:

- 12 capas de (8×8) corresponden a la codificación de una posición (cf. fig 2)
- 4 capas de (8×8) corresponden a la codificación de los derechos de enroque en lado rey/lado reina para blancas y negras.
- 1 capa de (8×8) corresponde a codificar a quién le toca mover (blancas o negras)

El **output** del algoritmo corresponde a: Un número real, el cuál indica con su signo quién tiene la ventaja (blancas = positivo, negras = negativo), y con su magnitud cuánta ventaja tiene dicho jugador. I.e.

$Output = 3$ implica que hay una leve ventaja de las blancas,
$Output = -200$ indica que hay una ventaja significativa de las negras.

Datos de entrenamiento

El entrenamiento para **Crimson Red** se desarrolló de la siguiente forma:

- Se creó una base de datos de 5000 partidas extraídas de la base de datos pública del sitio de internet *Lichess*.
- Por cada movimiento de cada partida, se creo un tensor de $17 \times (8 \times 8)$ y se creó una base de datos nueva donde se almacenaron las evaluaciones de *StockFish* de cada posición. Esto resultó en aproximadamente 400.000 posiciones, y respectivamente, 400.000 *labels* distintos, correspondiendo a dichas evaluaciones.

Datos de entrenamiento

- Se agregó una jugada aleatoria al 20 % de las posiciones totales, eligiendo aleatoriamente a qué posición se le hacía este cambio/reemplazo.
- Se entrenó la red con estos datos, utilizando un optimizador *Adam*, un learning rate de 0.001 y al **Error cuadrático medio (ECM)** como función de pérdida.

Arquitectura de la Red

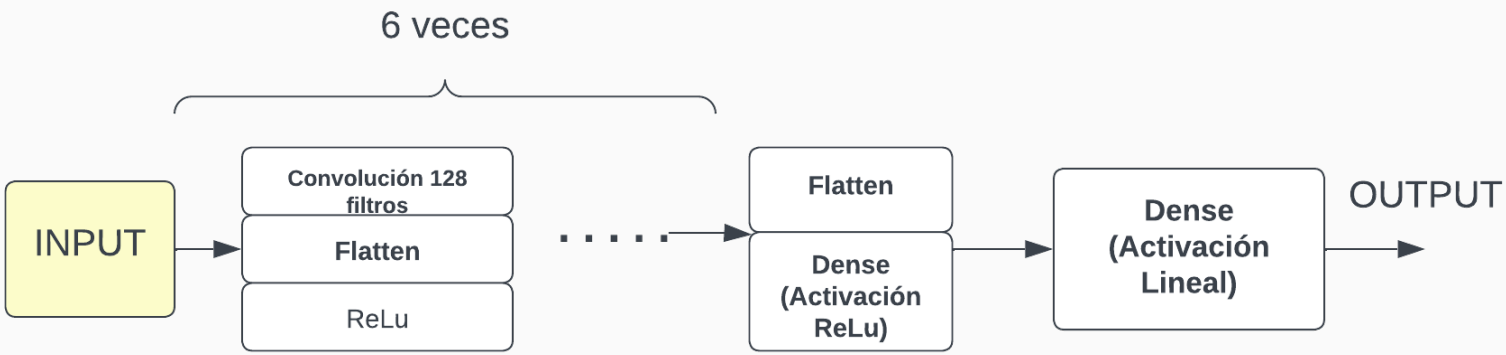
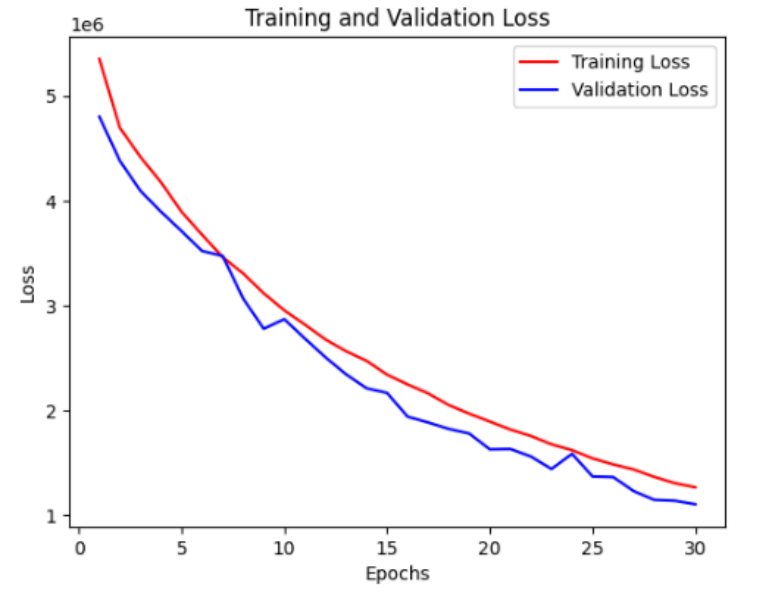


Figura 2: Estructura básica de la red neuronal de **Red Crimson**

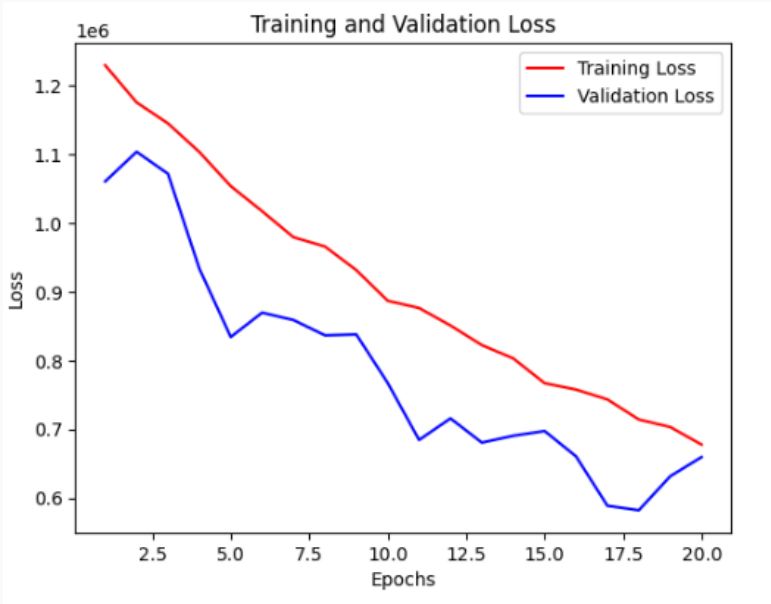
La red está constituida por 7 bloques de capas de convolución, flatten y un activador ReLu, puestos en serie, inspirándose en la estructura de red neuronal de algoritmos existentes, tales como *AlphaZero* (ref aquí), y terminando en una capa de activación lineal, permitiéndonos recuperar el número real que evalúa las posiciones deseado como output.

Resultados

Resultados



(a) Primeras 30 épocas



(b) Segundas 20 épocas

Figura 3: Pérdida (*ECM*) durante el entrenamiento

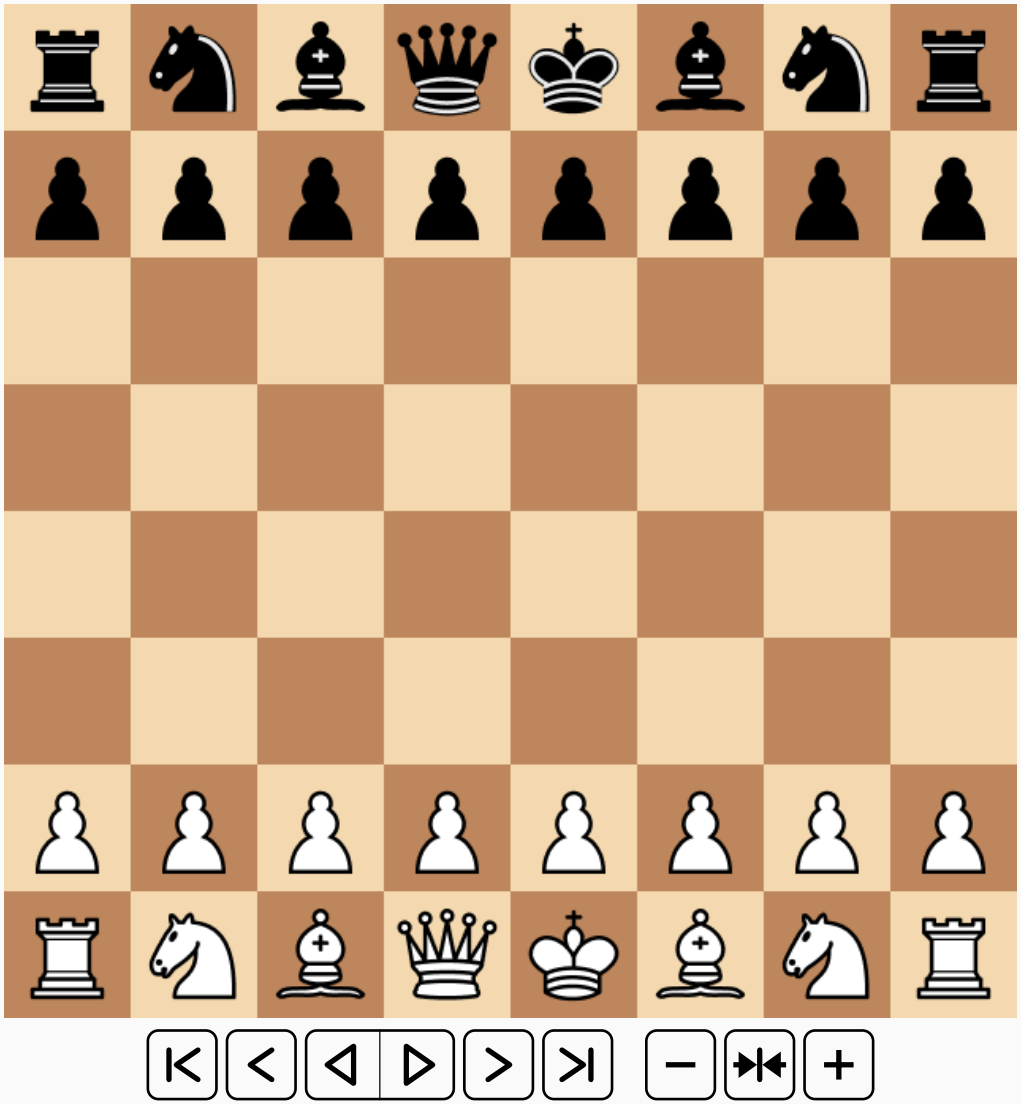
Crimson Red depth 1 V/S StockFish 1320 de elo

- Crimson Red de Blancas
  - Partidas Jugadas: 25
  - Partidas ganadas por StockFish: 25
  - Número promedio de Jugadas por partida: 37.12
  - Número máximo de Jugadas en una partida: 56
- Crimson Red de Negras
  - Partidas Jugadas: 25
  - Partidas ganadas por StockFish: 25
  - Número promedio de Jugadas por partida: 37.4
  - Número máximo de Jugadas en una partida: 65

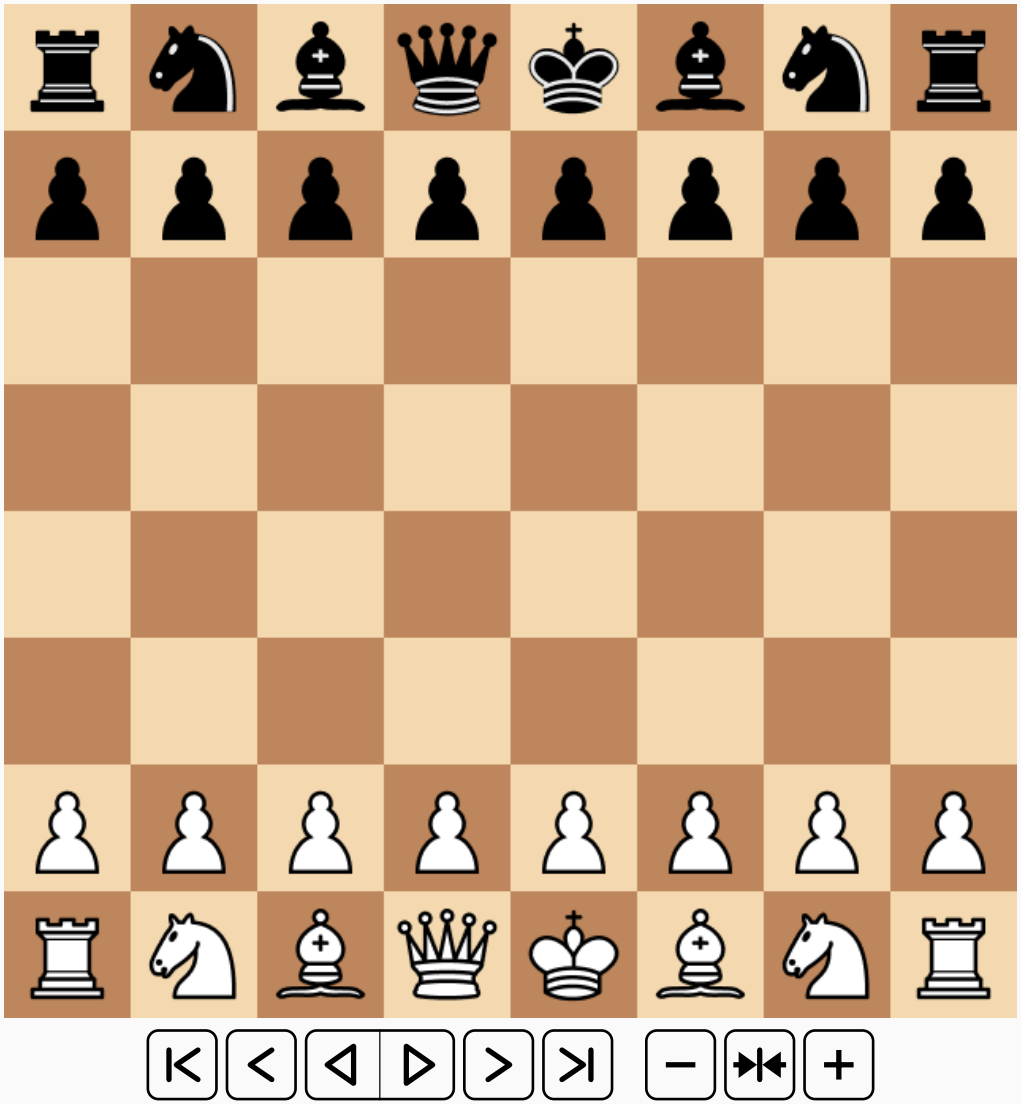
Crimson Red depth 3 V/S StockFish 1320 de elo

- Crimson Red de Blancas
  - Partidas Jugadas: 10
  - Partidas ganadas por StockFish: 10
  - Número promedio de Jugadas por partida: 54.4
  - Número máximo de Jugadas en una partida: 80

Partida de Red sin entrenar depth 3 (blancas) V/S StockFish 1320 de elo



Partida de Crimson Red depth 3 (blancas) V/S StockFish 1320 de elo



Conclusiones

Conclusión y Trabajo futuro

- Se logró crear un motor de ajedrez que aprendiera ciertas cosas del juego
- No se logró un motor de elo alto, lo que se debe posiblemente a:
  - Pocos datos de entrenamiento
  - Poca profundidad en el α-β-*prunning*

Trabajo futuro:

- Hacer una red neuronal que aprenda con autojuego utilizando *Reinforcement Learning*

Referencias

- Block, Marco et al. **“Using Reinforcement Learning in Chess Engines”.** En. En: ().
- Klein, Dominik. **Neural Networks for Chess.** arXiv:2209.01506 [cs]. Sep. de 2022. DOI: 10.48550/arXiv.2209.01506. URL: http://arxiv.org/abs/2209.01506 (visitado 14-12-2023).
- Lai, Matthew. **Giraffe: Using Deep Reinforcement Learning to Play Chess.** arXiv:1509.01549 [cs]. Sep. de 2015. URL: http://arxiv.org/abs/1509.01549 (visitado 09-11-2023).