

Jugando al Ajedrez con la IA: Clasificación de Tácticas con Aprendizaje Automático y Redes Neuronales

Introducción

Recopilación de datos

Transformación de datos

Redes Neuronales

Demo

Introducción

¡Bienvenidos a todos! Hoy hablaremos sobre cómo la inteligencia artificial (IA) está revolucionando el mundo del ajedrez. Veremos cómo se utilizan técnicas de aprendizaje automático y redes neuronales para clasificar tácticas de ajedrez, lo que permite a los jugadores mejorar su juego y tomar decisiones más informadas.



Recopilación de datos

Para empezar, necesitamos un buen dataset con el que entrenar el modelo que vamos a crear.

Por suerte, lichess.org nos proporciona de manera pública una colección de mas de 3 millones de puzzles, muchos de ellos centrados en las tácticas que queremos analizar.



lichess.org
open database

Lichess games and puzzles are released under the [Creative Commons CC0 license](#).
Use them for research, commercial purpose, publication, anything you like.
You can download, modify and redistribute them, without asking for permission.

STANDARD CHESS VARIANTS **PUZZLES**

3,314,253 chess puzzles, rated and tagged. [See them in action on Lichess](#).

[Download lichess_db_puzzle.csv.zst](#)

Format

Puzzles are formatted as standard CSV. The fields are as follows:

PuzzleId	FEN	Moves	Rating	RatingDeviation	Popularity	NbPlays	Theme
----------	-----	-------	--------	-----------------	------------	---------	-------

Sample

00sHx,q3k1nr/1pp1nOpp/3p4/1p2p3/4P3/B1PP1b2/B5PP/5K2 b k - 0 17,e8
00sJ9,r3r1k1/p4ppp/2p2n2/1p6/3P1qb1/2NQR3/PPB2PP1/R1B3K1 w - - 5 1
00sJb,Q1b2r1k/p2np2p/5bp1/q7/5P2/4B3/PPP3PP/2KR1B1R w - - 1 17,d1d
00s01,1k1r4/pp3pp1/2p1p3/4b3/P3n1P1/8/KPP2PN1/3rBR1R b - - 2 31,b8

Transformación de datos

El dataset nos proporciona las posiciones de los puzzles en formato FEN, un formato común para programas que te permiten jugar en línea, pero nos temíamos que no fuese el adecuado para nuestro modelo, así que lo transformamos en uno parecido al de una imagen 8x9, con la idea de usar redes neuronales convolucionales

Además, añadimos las 10 siguientes jugadas de cada puzzle calculadas por stockfish (una IA capaz de jugar al ajedrez) para ayudar a nuestro modelo a entender la situación, ya que no pensamos enseñarle a jugar al ajedrez.

```
[ ] generate_X_Y_sequence(df, 2)
[ 0, 0, 0, ..., 0, 0, 0]]],  
[[[ 4, 7, 7, ..., 7, 6, 7],  
[ 3, 7, 1, ..., 1, 1, 1],  
[ 1, 1, 7, ..., 7, 2, 7],  
...,  
[ 7, 7, 7, ..., 8, 7, 8],  
[11, 9, 7, ..., 7, 13, 7],  
[ 1, 0, 0, ..., 0, 0, 0]],  
[[ 4, 7, 7, ..., 7, 6, 7],  
[ 3, 7, 1, ..., 1, 1, 1],  
[ 1, 1, 7, ..., 7, 2, 7],  
...,  
[ 7, 7, 7, ..., 2, 7, 8],  
[11, 9, 7, ..., 7, 13, 7],  
[ 0, 0, 0, ..., 0, 0, 0]],  
[[ 4, 7, 7, ..., 7, 6, 7],  
[ 3, 7, 1, ..., 1, 1, 1],  
[ 1, 1, 7, ..., 7, 2, 7],  
...,  
[ 7, 7, 12, ..., 2, 7, 8],  
[11, 9, 7, ..., 7, 13, 7],  
[ 1, 0, 0, ..., 0, 0, 0]],
```

Redes Neuronales

Una de las mayores decisiones que tuvimos que tomar es la función de pérdida que usaríamos con nuestro modelo.

La primera opción que probamos fue una `binary_crossentropy`, que nos permite resolver un problema de clasificación con la posibilidad de que varias opciones sean correctas (Ya que una posición de tablero puede tener varias tácticas distintas).



Demo

1/1 [=====] - 0s 134ms/step

middlegame, short

Previous move Next move