Trial Task

Background

For the trial task we'll pick something ridiculous that no one would ever want to build, but that would test your speed and coding skills using AI.

We are intentionally including some custom logic here, and choosing an area (online chess) that you might not know, to test your ability to quickly comprehend unknown things and adapt - which would be useful when we're prototyping other people's ideas in the future.

The Task

Imagine we got a new client who wants to create a chess assistance engine, i.e. a chrome extension that shows him what moves to make when he's playing on chess.com.

The Requirements

The deliverable needs to be a chrome extension that can be loaded unpacked as a zip file + a public code repo on github.

It needs to include the following functionality:

A big enable/disable button.

A slider that goes from 0.01 (weakest) to 1 (strongest) and can be configured by the user in increments of 0.01. Denote this user input as L.

The following mode of operation:

1. Scan the screen and:

   a) parse the chess position that is currently in the user's browser window when the user is on chess.com.

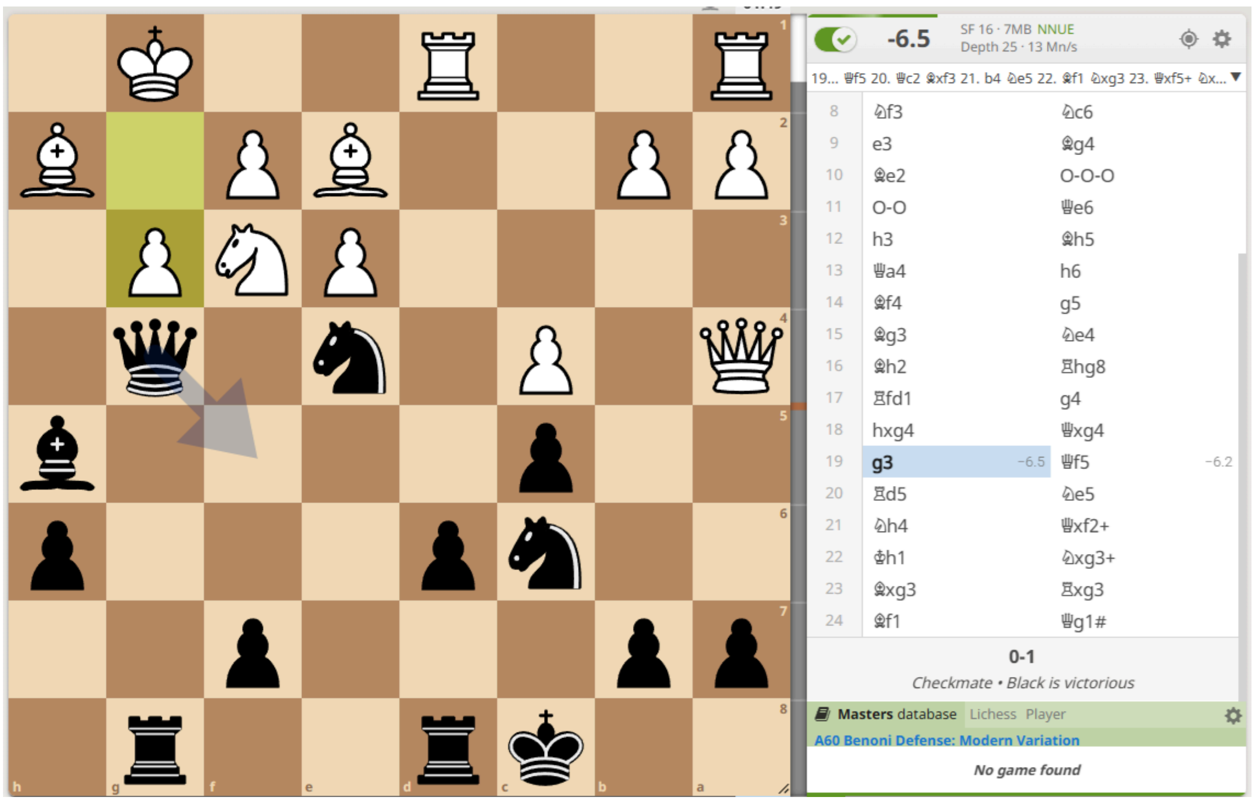
   b) Determine whether the user is playing as white or black, and whether it's the user's move.
2. If it's the user's move and the "enable" button is pressed:
   a. Feed the position into a chess engine (like Stockfish or Leela or anything similar)
   b. Receive back a sorted list of all the possible moves and their evaluations.
   c. If the user is playing black, multiply everything by -1 (to ensure that positive numbers mean the user is better)
   d. Convert all "mate in X" evaluations to 1000000/X to make the scale continuous, if needed
   e. Denote the evaluation of the best move as B.
   f. Determine the maximum error rate E as follows:

      If (B>=0) then E=(1-L)*B;

      Else E=-2*(1-L)*B;
   g. Pick the worst move whose valuation is above B-E.
   h. Draw an arrow that shows the selected move, overlaid on top of the position.

Explanation of the logic

Consider the following position:



-6.5 SF 16 · 7MB NNUE Depth 25 · 13 Mn/s

19… ♕f5 20. ♕c2 ♗xf3 21. b4 ♘e5 22. ♗f1 ♘xg3 23. ♕xf5+ ♘x… ▼

8	♘f3	♘c6
9	e3	♗g4
10	♗e2	O-O-O
11	O-O	♕e6
12	h3	♗h5
13	♕a4	h6
14	♗f4	g5
15	♗g3	♘e4
16	♗h2	♖hg8
17	♖fd1	g4
18	hxg4	♕xg4
19	g3 −6.5	♕f5 −6.2
20	♖d5	♘e5
21	♘h4	♕xf2+
22	♔h1	♘xg3+
23	♗xg3	♖xg3
24	♗f1	♕g1#

0-1

Checkmate • Black is victorious

Masters database Lichess Player

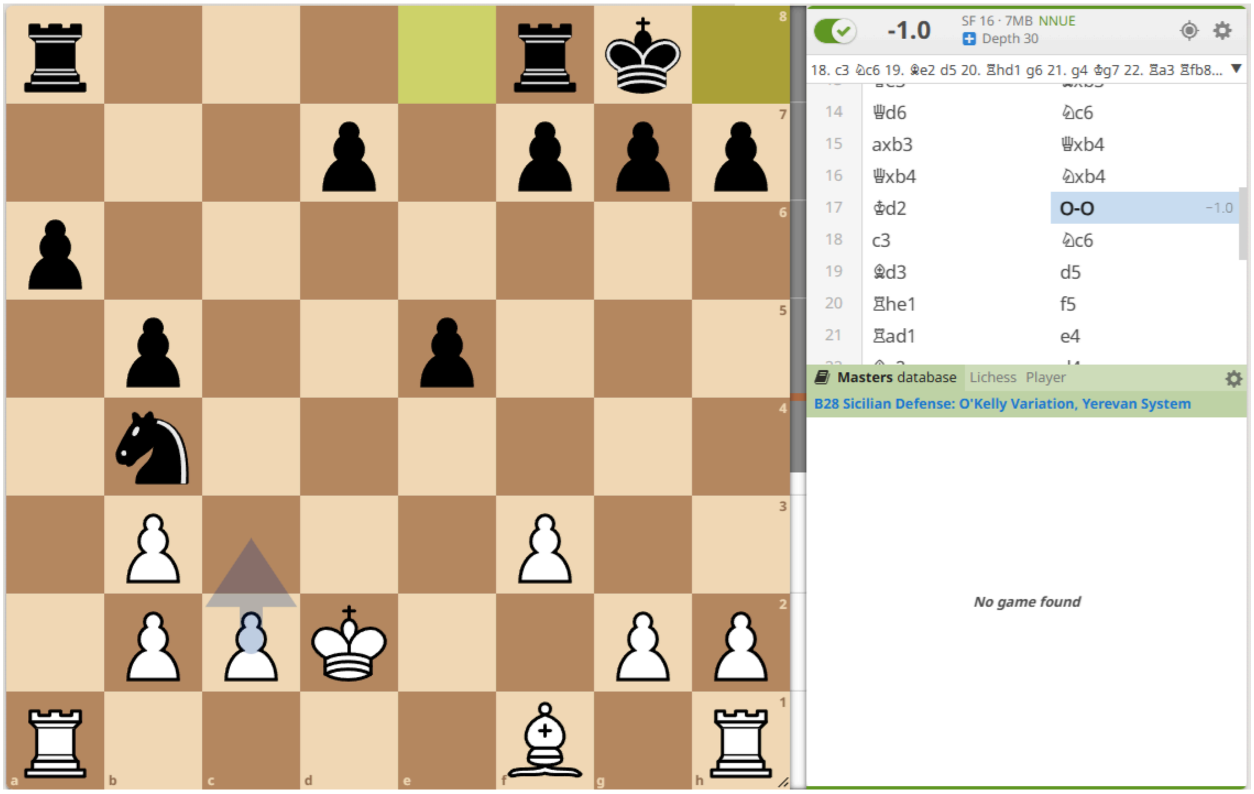
A60 Benoni Defense: Modern Variation

No game found

The user is better (because he's paying black, and the best move gives him a -6.5 evaluation).

If the level is set to 1, the extension should always draw an arrow to show the best move (i.e. queen to F5), but if the level is set to 0.5 (for example) then it should recommend the worst move that has a valuation between -3.25 and -6.5. In other words, it should lead us to fumble up to half the advantage on this move.

Now consider this position:



-1.0 SF 16 · 7MB NNUE
Depth 30

18. c3 ♘c6 19. ♗e2 d5 20. ♖hd1 g6 21. g4 ♔g7 22. ♖a3 ♖fb8...

14	♕d6	♘c6	
15	axb3	♕xb4	
16	♕xb4	♘xb4	
17	♔d2	O-O	−1.0
18	c3	♘c6	
19	♗d3	d5	
20	♖he1	f5	
21	♖ad1	e4	

Masters database Lichess Player

B28 Sicilian Defense: O'Kelly Variation, Yerevan System

No game found

The user is worse (because he's playing white, and the best move gives him a -1.0 valuation).

If the level is set to 1, the extension should always draw an arrow to show the best move (i.e. pawn to C3), but if the level is set to .5 then it should recommend the worst move that has a valuation between -2.0 and -1.0. In other words, it should lead us to make our positions twice as bad on this move.