Project Proposal

Mortimer von Chappuis

TL;DR — Implementing MuZero to tackle Chess with MCTS modifications

I always liked chess and the first artifical agent I ever created was for playing chess. It used alpha beta search and a handcrafted heuristic which managed to beat me about 70% of the time, making it a quite satisfing opponent. However, chess engines in the 21st century should not be beaten by a human and this has botherd me ever since.

Having implemented Alpha Zero in my Bachelors thesis made me aware of some weaknesses in the MCTS, regarding the initial Q estimates for unvisited nodes which I would like to modify. In short: New nodes are initialized with a Q value of 0. The Q estimate gets updated to average all V estimations from subsequent nodes, once they have been visited and evaluated. This is fine in a zero sum game where both players are playing near the optimum, but when a player starts losing and $V(S_t) < 0$, the Q estimates will be decreased, since all following states S_{t+1} are likely to be $V(S_{t+1}) < 0$ too. This leads MCTS to extreme branching, because every subsequent node visited will likely have $Q(S_t, a) < 0$ now, making all unvisited actions with $Q(S_t, a) = 0$ suddenly very attractive for upper confidence bounds of the type $\mathrm{UCB}(S) = \arg\max_a Q(S, a) + \lambda P(S, a) \cdot \frac{\sqrt{\sum_b N(S,b)}}{N(S,a)+1}$ where N is the visit count for states actions pairs, and P is the policy approximation. I would like to simply evaluate $Q(S_{t+1}, a) \leftarrow V(S_t)$ in the case of $N(S_{t+1}, a) = 0$, and in all other cases according to the standart MCTS evaluation method.

Additionally to those modifications, I would really like to get some experiance in learning world models in DRL. Therefore I want to use the opportunity to implement MuZero for Chess with MCTS modifications. The relevant papers are of cause the Alpha Zero and Mu Zero Papers, as well as some publications on Upper Confidence bounds for solving Multi Armed Bandits.