AI Lesson

04.09.2021

Vincent Manier, Jonathan Laurent

Main Take-aways from last week:

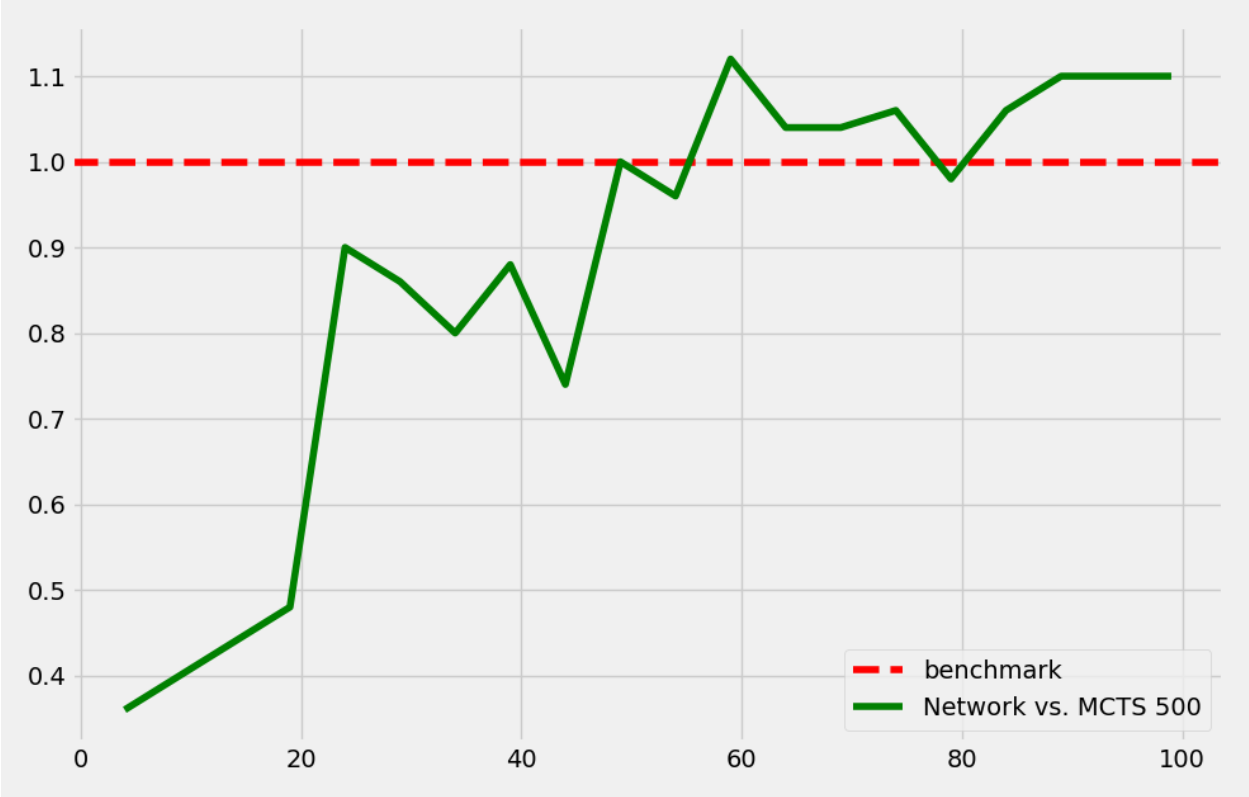
- Dirichlet noise to be applied at the root node only
- Look at bias in action head
- In supervised learning, some people initialize the weights according to the distribution in the training set (e.g. a classification problem with 2 classes with class1 = 90% training set, class 2 = 10% training set)
- Clarify the target on the competition chart (not clear that 1 was the objective)
- Use of github and git:
    - Ignore files like .DS_Store
    - Only put python files in src folder
    - Use branches and tags
- Editor: look at VS code (Jupyter integration, linter, debugger...)
- Re-work MCTS module with an oracle
- Look at the statistical number of games to play in AlphaZero.jl documentation [`necessary_samples(ϵ, β) = log(1 / β) / (2 * ϵ^2)`
- Asynchronous vs. parallel - concepts:
    - Concurrent but not parallel
    - Threads
    - Tasks
    - Scheduler
- Limiting factor: RAM
- Virtual loss

A. Objectives

- Review symmetries functions (2 were not operational)
- Dirichlet to be applied at the root node only + re-run experiences
- Clean up my Github and use branching / tags
- Alternate player order for testing the network
- MCTS module with an oracle - branch new_mcts
- Add a config file for general parameters - branch config_file
- Use vscode
- Look at the use of a profiler (cProfile)
- Adjust the v value to reward faster wins v = win/loss * f_penalty (number of turns played - minimum)
- Inspect biases values - bias_check branch
- Read about asynchronous vs. parallel implementation
    - Pluralsight course: Concurrent programming in Python
- Parallel implementation for self-play and network vs. MCTS competition (time divided by 3 on average - I guess because of 4 processes run in parallel, including 1 dedicated to resources management)
- Review my initial notes about AlphaZero paper

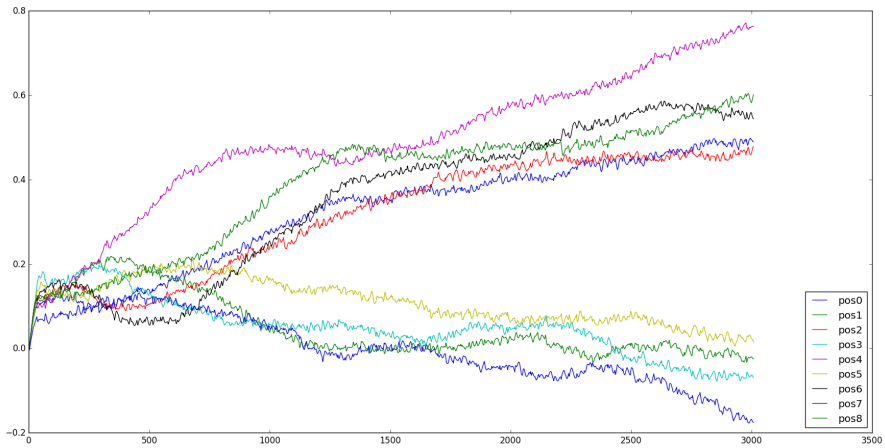
B. Results

Score of the network only vs. MCT-500



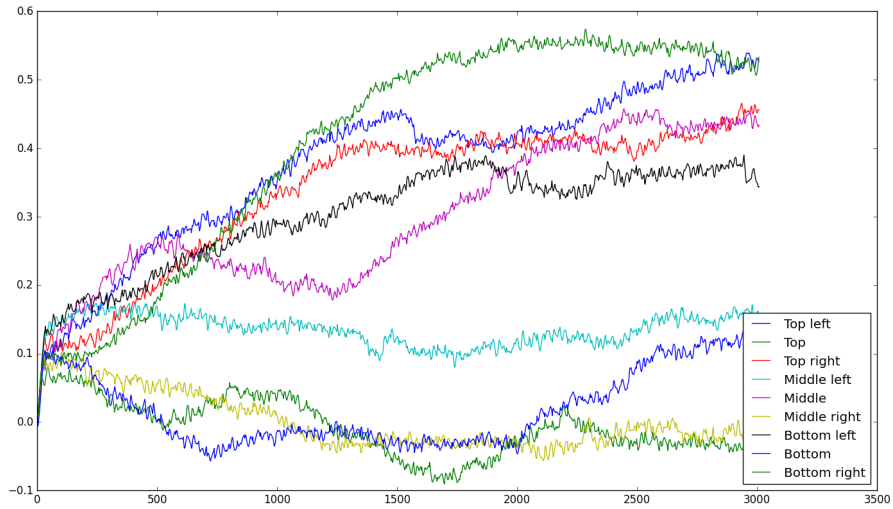
Evolution of the bias value in action head (i.e. probability distribution) layer

[experience1]



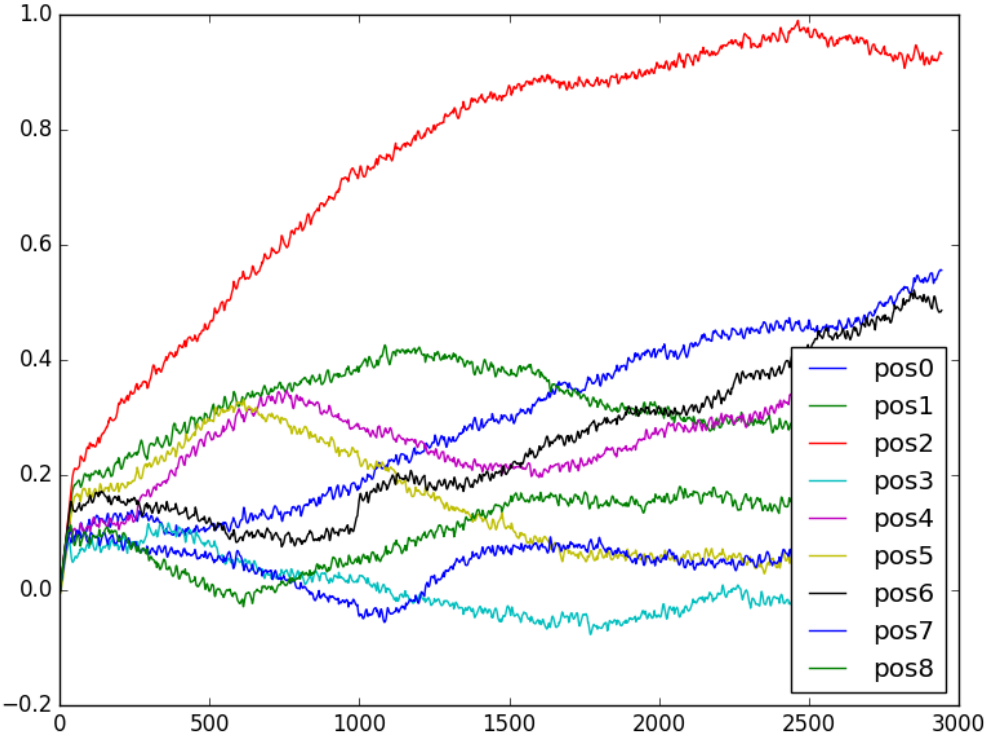
Pos0 = (0,0), pos1 = (0,1), …, pos8 = (2,2)

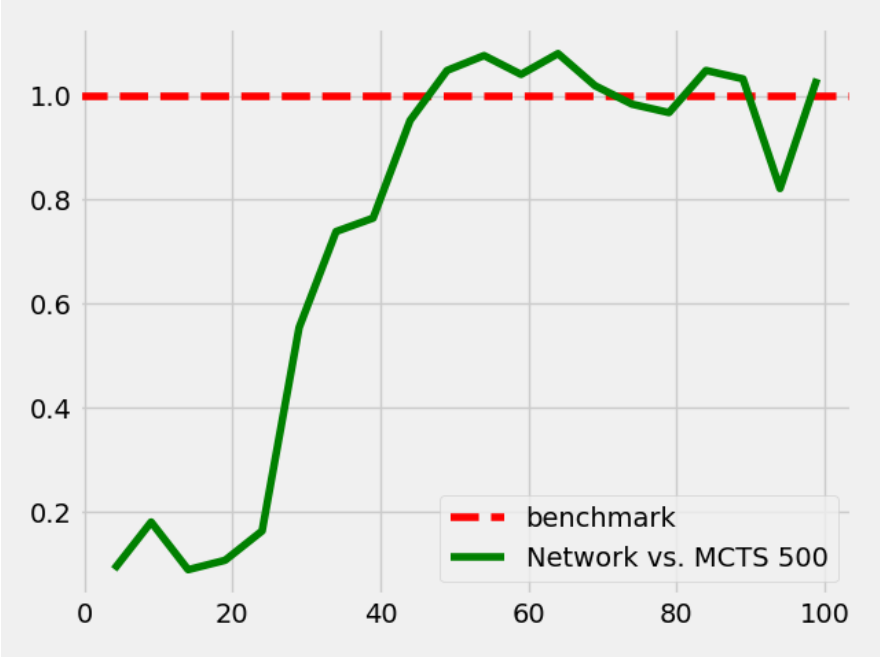
Here, the middle of the board (pos4 = center_position) is favored but it's not always the case…

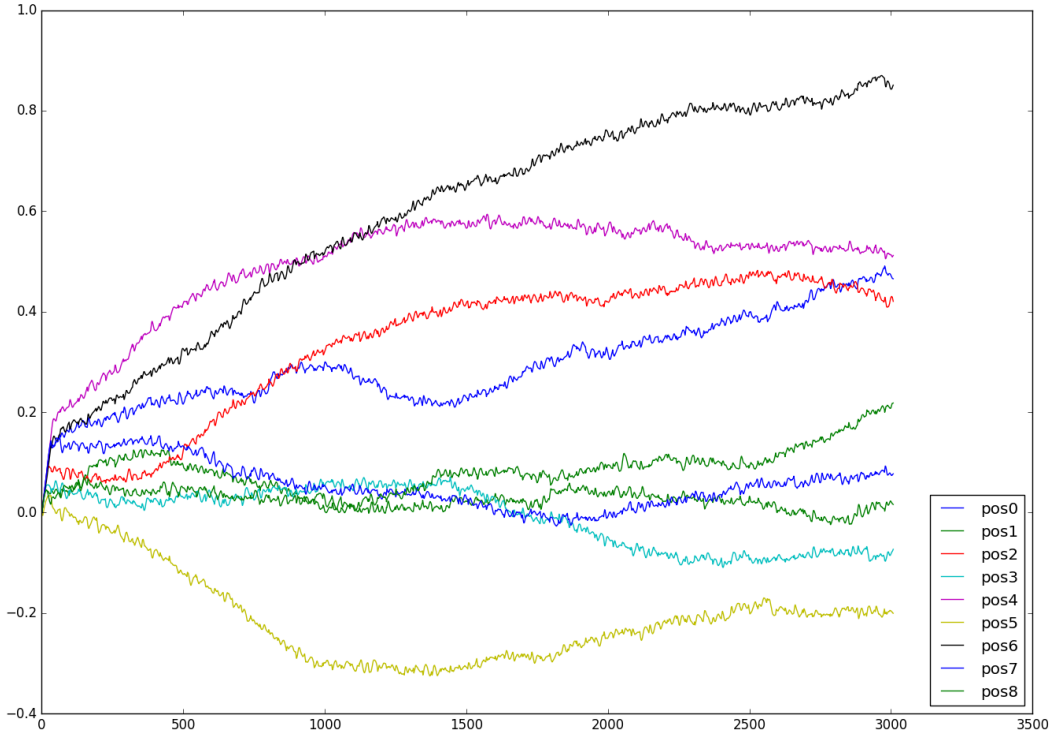


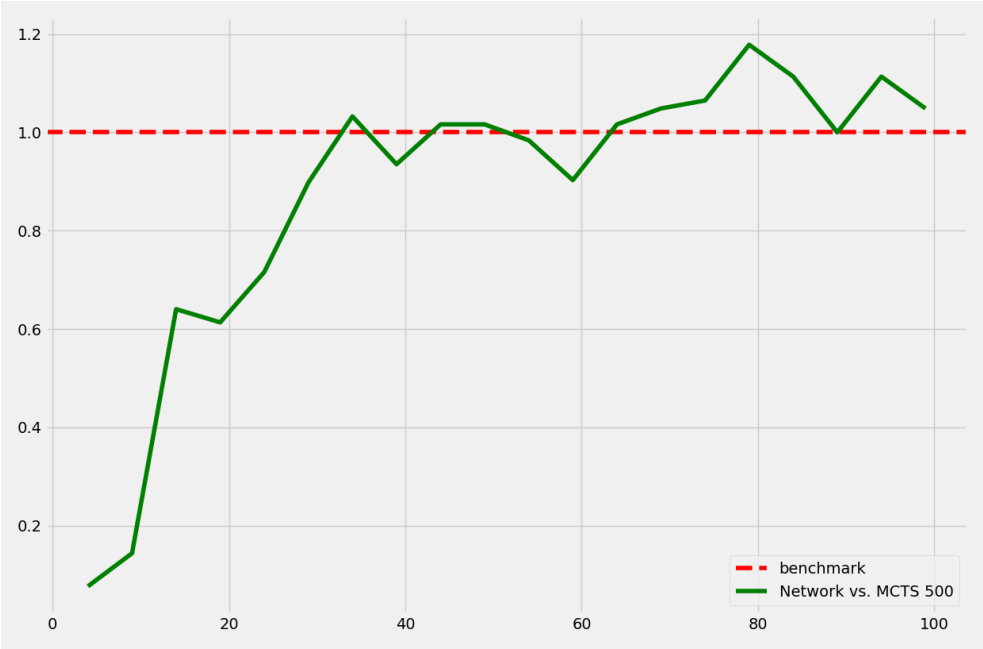
In general, corners and middle positions are favored

Training with discounted score to encourage faster wins



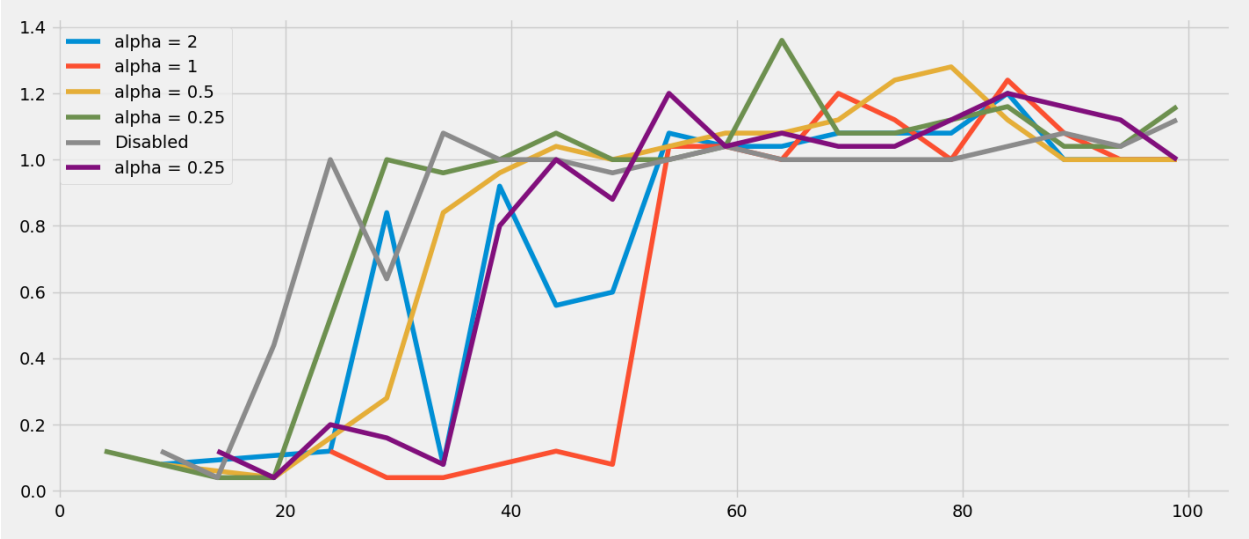



1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0
500
1000
1500
2000
2500
3000
3500
pos0
pos1
pos2
pos3
pos4
pos5
pos6
pos7
pos8


1.2
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
benchmark
Network vs. MCTS 500

Tests on Dirichlet Alpha

Before fixing the bug



After fixing the bug