The Future of Artificial Intelligence

Tom Rochette <tom.rochette@coreteks.org>

August 30, 2025 — 861fb9d0

0.1 Context

If you've already seen Hamis Hassabis presentation General Learning Algorithms, you may skip to 27:25 since most of the content he covers is already covered in this other presentation. This content will not be covered here, so please refer to the other presentation article.

0.2 Learned in this study

0.3 Things to explore

1 Overview

1.1 AlphaGo

- Pattern recognition with planning

1.2 Go

- 3000 years old
- 40M players
- 10^{170} positions
- 2 rules
  - Rule #1: The capture rule: Stones are captured when they have no liberties
  - Rule #2: The "Ko" Rule: A repeated board position is not allowed

1.3 Why is it hard for computers to play?

- The complexity makes brute force exhaustive search intractable
- Two main challenges
  - The branching factor is huge
  - Writing an evaluation function to determine who is winning is thought to be impossible
- Go is a game of intuition
- Computes are not very good at replicating intuition
- In chess, one can use heuristics to roughly determinate who's winning
- In go, this is not possible as each pieces have the same value

1.4 Training the deep neural networks

- Download 100K games from amateur play
- Train a supervised learning policy network to try and copy those "expert" players
- Train through a reinforcement learning policy network by playing against itself
- Generate new data (30M positions) which will help produce an evaluation function
- Build a value network which indicates, based on the current board, who's winning the game and estimates by how much

1.5 Two networks: Policy and Value Nets

- Policy network: Probability distribution over moves
- Value network: A real number between 0 and 1, where 0 is white winning, 1 is black winning and 0.5 is the game is even
- The policy network can be thought of as reducing the breadth of the search tree
- The value network can be thought of as reducing the depth of the search tree

1.6 Combining Neural Nets with Tree Search and Rollouts

- Q - Action value of a move
- Do a little bit of searching in the tree
- Find a couple of promising moves
- Find the moves that have the maximum Q values
- Follow the trajectory that has a high Q value until we hit a node that hasn't been explored yet
- First, we call the policy network to expand the tree at that point but only for the moves with the highest prior (P) probability of that move occurring
- Second, we call the value network to evaluate that position and give an estimate of who is winning
- We also do another thing if we have time, which is rollouts to the end of the game (maybe a few thousands) to collect "true" statistics about who ends up winning the game from that position
- Then, we combine these two estimates (from the value network and the rollouts) to give a final estimation

1.7 Testing

- Internal testing: Different version of AlphaGo playing against itself 24/7
- Calibration: Play against external programs such as Zen and CrazyStone
- In April 2015: Winning 99% of games against Zen and CrazyStone
- In October 2015: Could beat the April 2015 version 100% of the time and beat Fan Hui 5 out of 5
- In March 2016: No estimate against the October 2015 version. Lee Sedol can beat Fan Hui about 97% of the time

2 See also

- General Learning Algorithms (some of the content of this presentation is already covered in this other presentation)

3 References

- The Future of Artificial Intelligence