Research review on adversarial game-playing

Rui Sá Pereira, Udacity AI nanodegree, term 1, project II

In this review, we summarize the work developed by Silver et al. [2], in which a team from Google DeepMind built, with a combination of Monte Carlo tree search (MCTS) and deep neural networks, a Go game-playing agent, AlphaGo, that is not only several orders of magnitude better than previous developed similar agents but, after the publication of their work, beat by the first-time, and by 5-1, the reigning Go World champion. We first briefly describe how AlphaGo selects moves and evaluates positions, training in the meanwhile two networks, the policy and the value network.

In order to select moves, the authors first trained a supervised learning (SL) policy network from expert human moves (recall that a policy is a probability distribution over possible moves, in a given position). More specifically, a convolutional network with 13 layers, the SL policy network, was trained on a dataset containing 30 million human expert positions. This training resulted in an accuracy of around 57%, significantly higher than the maximum of previous state-of-the-art networks, 44%. Following the training of the SL policy network, another similar network, the reinforcement learning (RL) policy network, was trained by auto-playing between the current policy network and a randomly selected previous iteration of the policy network. Such randomization prevents overfitting. The authors then remark that the RL policy network won more than 80% more of the games against the SL policy network. Moreover, even without searching for optimal moves, it won 85% of the games against the strongest open-source Go program at the date of publication, against 11% for the previous state-of-the-art, which was based solely on supervised learning of convolutional networks.

When evaluating positions, the fact that successive positions are strongly correlated, since they differ only in one stone and share the same target, leads to overfitting when predicting positions. Indeed, the authors indicate that such approach memorizes positions, rather than generalizing them. In order to overcome this significant shortcoming, a new training method was devised: Generate new self-play data comprising 30 million distinct positions, each sampled from a different game, where each game is played between the RL policy network and itself until termination. This approach resulted in identical Mean Square Errors (circa 0.23) for training and test data, which indicates minimal overfitting.

In the next part, the authors introduce their innovative search algorithm, which makes use of the policy and value networks above described to guide a Monte Carlo tree search. Indeed, the SL policy networks are used to make a very efficient selection of moves, by processing the leafs of the tree. The value network and the outcome of a random rollout using a fast rollout policy (this rollout policy representing the simulated part of MCTS) are then combined into a leaf evaluation. Such heuristics are very costly to compute, which led the authors to use in the final version of the resulting program, AlphaGo, an asynchronous 40-threaded search that executed simulations on 48 CPUs, and

computed the policy and value networks on 8 GPUs.

Finally, and regarding performance, AlphaGo won 494 out of 495 games played with other agents. Even allowing for 4 handicaps, or free movements for the opponent, AlphaGo won at least 77% of the games played with each of other 3 agents. Even more conclusive of its superior game-ability, the number of positions that AlphaGo required to won the European Go champion was thousands of times fewer than the number of those Deep Blue required to win against Gary Kasparov at chess. This evidence seems to point the justification for such an outperformance directly to the much more intelligent way in which AlphaGo evaluates and selects positions, in comparison to all previous agents.

We finish this report with a reference to the authors' most recent work, where an agent learning only be self-playing, and without recourse to human expert moves was introduced. The resulting program, AlphaGo Zero, beat Alpha go by 100-0.

References

[1] Several authors, Mastering the game of Go with deep neural networks and tree search. Nature, 529, 484–489, January 2016.

[2] Several authors, Mastering the game of Go without human knowledge. Nature, 550, 354–359, October 2017.