Combining heuristics with counterfactual play in reinforcement learning.

Erik J Peterson (erik.exists@gmail.com)
Department of Psychology, Carnegie Mellon University,
5000 Forbes Ave, Pittsburgh, PA 15213

Necati Alp Myesser (nmuyesse@andrew.cmu.edu)
Department of Mathematical Sciences, Carnegie Mellon University,
5000 Forbes Ave, Pittsburgh, PA 15213

Kyle Dunovan (kdunovan@andrew.cmu.edu)
Department of Psychology, Carnegie Mellon University,
5000 Forbes Ave, Pittsburgh, PA 15213

Timothy Verstynen (timothyv@andrew.cmu.edu)
Department of Psychology, Carnegie Mellon University,
5000 Forbes Ave, Pittsburgh, PA 15213

Abstract

Deep reinforcement learning can sometimes match and exceed human performance, but if even minor changes are introduced artificial networks can't adapt what they've learned to new situations. Two reasons why people are so eminently adaptable is their use of heuristics, and their ability to imagine new environments and learn from them, a kind of counterfactual reasoning. We've developed a model of hierarchical reinforcement learning which includes both these elements. Using a board game with a known optimal strategy–Wythoffs game–we show that this "stumbler-strategist" network promotes generalizability and robustness to new environments and rule changes, while also improving post-training interpretability of learning outcomes.

Keywords: hierarchical reinforcement learning; transfer learning; strategy

Deep reinforcement learning can rival human performance on games of strategy like chess and Go (Silver et al., 2016, 2017), and less structured games like classic Atari video games (Mnih et al., 2015). However unlike humans artificial networks are unable to transfer their performance to new situations, even ones that include trivial changes (Lake, Ullman, Tenenbaum, & Gershman, 2017). Part of the human capacity for generalization may come from our ability to imagine new environments and learn from them, a kind of counterfactual reasoning (Pearl, 2017). Our success also stems from our ability to develop and exploit heuristics (Lake et al., 2017; Toyama, Katahira, & Ohira, 2017). To try and improve transfer learning in artificial networks, we combined these two elements in a novel variation of hierarchical reinforcement learning, which we term the stumbler-strategist network architecture.

Stumbler-strategist networks

We introduce stumbler-strategist networks (Figure 1) a variation on the DYNA-Q models (Sutton, 1990). Stumblers are model-free reinforcement learning agents that observe and act in the environment. Like all model-free systems they can only discover the value of specific input-output associations, without any appreciation for the organizing features of the game that govern these associations.

Strategists are model-based agents that *can only* observe what the stumblers learn, *sampling* and *aggregating* information about many stumblers' actions. Strategists don't act directly but can, when confidence is high, bias stumbler's actions. We imagine that like the prefrontal cortex, strategists are bound to an abstract realm–neither directly observing or acting on the world.

Our key innovation is in how strategists use information: they simulate counterfactual play in *new and more challenging* environments. That is, to learn to transfer knowledge they "imagine" and play new games based on their observation of stumblers activity. The strategist layer can succeed in these new, harder, environments because it is imbued with a loose heuristic strategy (described below).

Wythoff's game

To understand our strategist's heuristic, we must first describe the environment in which it plays.

To isolate strategy learning and transfer, our agents play an (impartial) board game called Wythoff's game. The game is played on a two dimensional grid in which players alternate turns to move an object that is initially on the bottom-right corner towards the top-left corner. The player who gets to place the object in the top-left corner terminates, and thereby wins the game. Every turn, the object can be moved horizontally, vertically, or diagonally towards the top-left corner.

Despite their simplicity "gridworld" environments (like that in Wythoff's game) can provide a challenging but controllable test-bed for transfer learning. Indeed, in these simple environments two state-of-the-art deep reinforcement learning net-

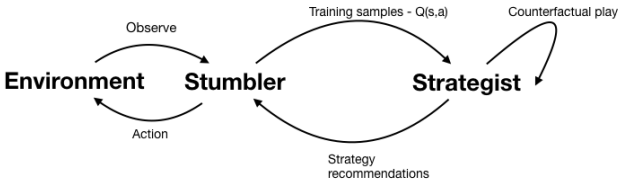

Figure 1: Diagram of the stumbler-strategist architecture. Stumblers, implemented as model-free agents driven by the Q-learning rule and a softmax action-selection policy ($= 0.7$ for all models), interact with the game environment, generating $Q(s,a)$ training samples for the strategist. The strategist uses the current sample, as well as past experiences from a memory replay loop, to simulate counterfactual play on a new (larger) game board.

works could not adapt to subtle differences in the training and testing environments (Leike et al., 2017).

The difference between an impartial game, like Wythoff's, and a strategy game like Go, is that an impartial game has a single *ground truth* solution. Every position in an impartial game is either *hot*, meaning that there exists a winning strategy for the player about to make a move, or *cold*, meaning that under optimal play, the player about to make a move will always lose. The distribution of *hot* and *cold* positions across the state space in an impartial game usually comes with inherent mathematical structure.

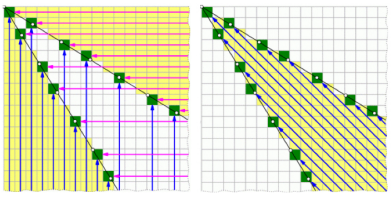

Figure 2: Cold positions in Wythoff's game are distributed along two symmetrical lines. Arrows show how from every other position (hot) there exists a move to a cold position. There does not exist any move from a cold position to another cold position. Figure used with permission from Zachary Abel (Abel, 2014)

The heuristic

The heuristic used by the strategist is that "cold states are symmetric about the diagonal of the game board". To implement this, first max$Q(s,a)$ values are classified for each state s as *cold* if $0 \leq \text{max}Q(s,a) \leq \varepsilon$ and *hot* if $1-\varepsilon \leq \text{max}Q(s,a) \leq 1$, where $0 < \varepsilon < 1$. Second, random samples of this $hot/cold$ dataset are then fed into a separate neural network, but all $cold$ states *always* have their complement included in the training set. That is, if state (i,j) is classified as $cold$, state (j,i) is also included in the training set *even if it was classified as* hot.

Results

The strategist displayed great efficacy in producing generalizable models for Wythoff's game. Figure shows how the two components of the network learned the value of board positions at different stages of learning. Models that are developed in the earlier stages of training remain mostly irrelevant to generalization; however, models that *meaningfully* generalize, although with low accuracy, begin to emerge soon after initial training. Such models are crucial for the learning process because they influence the way the stumbler chooses to explore different action spaces. Without such guidance, the stumbler explores actions without any overall purpose or *insight*. With the guidance from the strategist, the stumbler explores actions that would either contradict or confirm an overall hypothesis about the nature of the learning environment.

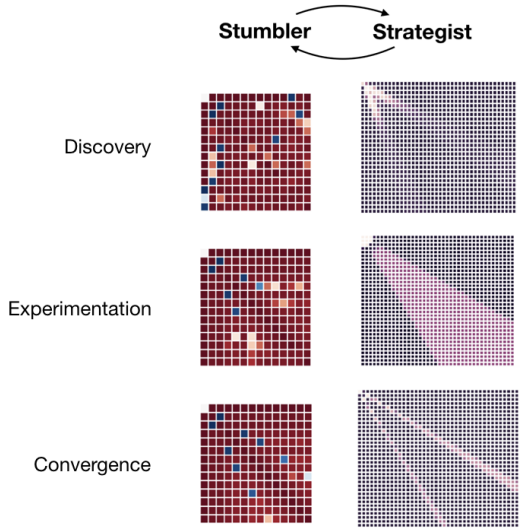

Figure 3: Learning in three stages of training on Wythoff's game. The early models (Discovery) will be largely unsuccessful, while certain inaccurate generalizations (Experimentation) will supply reasonable strategies to the stumbler, allowing the provision of useful datasets into the network that translate into accurate and general models (Convergence). In this example the stumbler trained on a 14 by 14 board for 2000 game-plays, across each strategist time-step. The strategist learned to play on a 50 by 50 game board.

We compared the performance of the stumbler-strategist network to two types of stumblers. Stumbler 1 used lookup table approach to Q-learning (described in Alg. 1). Stumbler 2 used a Deep Q Network-style (DQN) design (et al Minh, 2015), which allowed us to asses if an over-parameterized model could capture similar strategic information. Figure 4 shows the accuracy of all three networks during learning. The stumbler-strategist network agent improves performance in discrete jumps as better models replace worse ones over time.

Notably here, in this simple environment the Deep Q Network doesn't appear to learn at all.

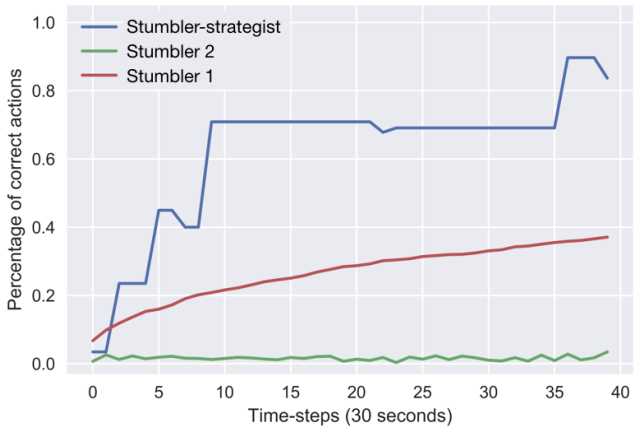

Figure 4: The stumbler-strategist outperforms both stumblers on identical training periods. The strategist trained on a 50 by 50 board on with 5000 counterfactual gameplay simulations per time-step. Stumblers trained on a 12 by 12 board with 1000 simulations each. Change in network architecture, cost functions, or layer count did not have a noticeable effect on the learning outcomes of all networks

To see whether the stumbler-strategist network's generalizability was maintained with a larger board size, we trained the model on a 12 by 12 stumbler board, and on strategist boards ranging from 12 to 300. Though performance declines as board size rises, in the largest size (a 300 by 300 board) the model achieved 70% accuracy (Figure 5).

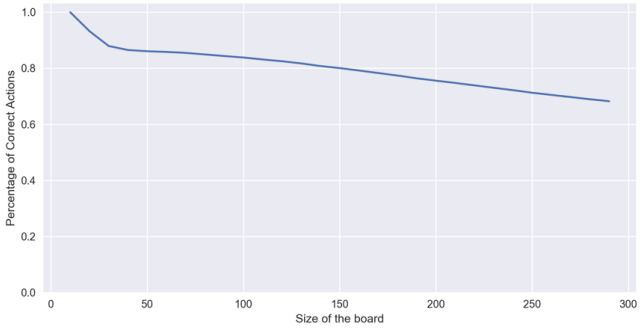

Figure 5: Performance of a stumbler-strategist network as the strategist's board dimensionality rises, but the stumbler board stays the same. The stumbler in this model was trained on a 12 by 12 board. The strategist trained using 2000 counterfactual gameplay simulations.

Discussion

Our key innovation is the introduction of a "strategist" layer, which simulates counterfactual play in *new and more challenging* environments. The strategist layer however only succeeds in these new, harder, environments because it is imbued with a loose heuristic: important moves have a mathematical symmetry on the game board.

We studied our stumbler-strategist in a toy "gridworld" environment, with known optimal play (i.e. we studied Wythoff's game). Moving from simple games such as this to open-ended strategy games like Go and to complex visual environments like classic Atari games will require at least two further innovations.

First, gridworld games share common coordinates. It is therefore simple to move from smaller boards to larger and more challenging boards by just mapping between common coordinates. Using a stumbler-strategist network in visually complex environments, like classic Atari, therefore requires solving this projection problem–which may be nontrivial. There is room for optimism though, as the response in higher-level layers in DQN networks is related to the perceptual similarity of the input images, suggesting strategists could observe not only values, but also critical perceptual relationships by only studying stumblers' responses to the world.

Second, we studied a single strategic heuristic (spatial symmetry). It is not reasonable to expect this heuristic to apply in all but a few select instances. However there exists a substantial literature in game theory representing a large pool of possible heuristics and strategies. While these will certainly not describe optimal performance in all situations, there are perhaps consistent moments to be found in complex games where simple strategies, and matched counterfactual simulations, will allow for information to be efficiently transferred between environments.

Methods

Stumbler learning

Stumbler learning is governed by Q-Learning, extended to allow for "top-down" strategist feedback and interactive game play.

Algorithm 1 Learning algorithm used by stumbler

procedure STUMBLE($Q, Strategist$)
  $\alpha \leftarrow$ learning rate
  $\beta \leftarrow$ exploration-exploitation
  $L \leftarrow$ confidence in Strategist
  $\zeta \leftarrow$ confidence convergence rate to L
  $G \leftarrow$ Initialize Wythoff's game
  while G continues **do**
    $p \leftarrow$ confidence in $Strategist$ given L, ζ
    if $x \sim N(0,1) < p$ **then**
      $action \leftarrow$ greedy move per $Strategist$
    else ▷ do Q_learn
      $action \leftarrow$ softmax$(Q(s,a),))$
    do $action$ on G
    if G ends **then**
      $reward \leftarrow 1$ ▷ Winning move
    else ▷ Opponent plays
      do greedy action using Q on G
      if G ends **then**
        $reward \leftarrow -1$ ▷ Opponent wins
      else
        $reward \leftarrow 0$
    $Q' \leftarrow$ max Q-value from new state s'
    $Q(s,a) \leftarrow \alpha(reward + Q' - Q(s,a))$
  return final Q

Strategist training

Strategists are perceptrons, trained on input coordinates (i, j) and output values derived using the heuristic described above. The Memory is replay module which holds the last 2000 Q values from prior stumbler trials. Strategists were trained using batches x of 100 randomly sampled events from memory.

Algorithm 2 Learning algorithm used by the Strategist

procedure STRATEGIST(Q)
  $(i, j), V \leftarrow$ Heuristic(Q)
  Strategist $\leftarrow$ initialized neural network
  backpropagate $Strategist$ with using cross entropy loss
  return Strategist

References

Abel, Z. (2014). *Putting the why in wythoff.* http://blog.zacharyabel.com/2012/06/putting-the-why-in-wythoff/.

et al Minh, V. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*, 529–533. Retrieved from http://dx.doi.org/10.1038/nature14236 doi: 10.1038/nature14236

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2017). Building machines that learn and think like people. *Behavioral and Brain Sciences*, *40*.

Leike, J., Martic, M., Krakovna, V., Ortega, P. A., Everitt, T., & Lefrancq, A. (2017). Ai safety gridworlds. *arXiv*.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Ostrovski, G. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*, 529-533.

Pearl, J. (2017). Theoretical Impediments to Machine Learning With Seven Sparks from the Causal Revolution Scientific Background. (September), 1–8.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Driessche, L. S. G. V. D., Schrittwieser, J., ... Lanctot, M. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, *529*(7587), 484-489.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., ... Hassabis, D. (2017). Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. , 1–19. Retrieved from http://arxiv.org/abs/1712.01815 doi: 10.1002/acn3.501

Sutton, R. S. (1990). Integrated Architectures for Learning, Planning, and Reacting Based on Approximating Dynamic Programming. *Machine Learning Proceedings 1990*, *02254*(1987), 216–224. doi: 10.1016/B978-1-55860-141-3.50030-4

Toyama, A., Katahira, K., & Ohira, H. (2017). A simple computational algorithm of model-based choice preference. *Cognitive, Affective, & Behavioral Neuroscience*, 1–20.