MarketsAI

A Deep Reinforcement Learning Approach to Simulating High-Dimensional and Imperfect Information Economies

Matias Covarrubias

June 25, 2021

Abstract

This paper presents a new framework to set and solve economic models that uses Reinforcement Learning (RL) as a description of the agents' decision making process.

1 Single-Agent Reinforcement Learning

1.1 An introduction to Reinforcement Learning

- Define Markov Decision Problems.
- Example 1: Monopolist problem.
- Algorithm: Q-learning.
- Proof of convergence and theory.
  - Ma et al. [2020] show formally that the recursive specification of the q values can be considered as a transformation of the bellman operator. The authors use this q-transform to solve problems with unbounded rewards.
  - Ma and Stachurski [2021] use computational complexity theory and numerical experiments to conclude that the q-transform leads to gains in computational efficiency. NOTES: WHY?
  - Bertsekas and Yu [2012] introduce a mix of policy iteration and q-learning that improves efficiency even for approximate methods. It seems great check it.
  - Tsitsiklis [1994] perform an extensive convergence analysis of q-learning algorithms. His proof of convergence, which we will review in chapter 1, is based on contraction mapping theorem,

1.2 Deep Reinforcement Learning

- Explain DQN.
- Explain SAC.

1.3 Solving Stochastic Growth Models

- Example 2: Stochastic Growth Model.
- Q-learning vs Deep-Qlearning as the dimensionality increases.

2 Multi-Agent Reinforcement Learning

2.1 Introduction to multi-agent Reinforcement learning

- Define Partially Observed Markov Games.
- Example 1: Algorithmic Collusion.
- Algorithm: Independent learning.
- Challenges of multi-agent.
- Soutions: QMix and other algorithms.
- Famous examples.
  - Berner et al. [2019] achieve super-human performance at the game Dota 2. The game is challenging for multiple reasons. It is multi-agent, stochastic and it has a large action and observation space and it requires long term strategy. They authors used a single neural net to approzimate both the value function and the policiy function. In particular, they used a single-layer 4096-unit LSTM.
  - Baker et al. [2019] show that agents self-playing hide and seek in a complex environment exhibit emergent curricular learning, in which they progressively discover strategies and the device counter-strategies. They use PPO with GAE. The distributed computation framework is called rapid.
  - Vinyals et al. [2019] achieve grand-master level at the game StarCraft 2. "Observations of player and opponent units are processed using a self-attention mechanism. To integrate spatial and non-spatial information, we introduce scatter connections. To deal with partial observability, the temporal sequence of observations is processed by a deep long short-term memory (LSTM) system. To manage the structured, combinatorial action space, the agent uses an auto-regressive policy and recurrent pointer network.". "For every training agent in the league, we run 16,000 concurrent StarCraft II matches and 16 actor tasks (each using a TPU v3 device with eight TPU cores23) to perform inference. The game instances progress asynchronously on preemptible CPUs (roughly equivalent to 150 processors with 28 physical cores each), but requests for agent steps are batched together dynamically to make efficient use of the TPU. Using TPUs for batched inference provides large efficiency gains over previous work14,29. Actors send sequences of observations, actions, and rewards over the network to a central 128-core TPU learner worker,

which updates the parameters of the training agent. The received data are buffered in memory and replayed twice. The learner worker performs large-batch synchronous updates. Each TPU core processes a mini-batch of four sequences, for a total batch size of 512. The learner processes about 50,000 agent steps per second. The actors update their copy of the parameters from the learner every 10 s. We instantiate 12 separate copies of this actor–learner setup: one main agent, one main exploiter and two league exploiter agents for each StarCraft race. One central coordinator maintains an estimate of the payoff matrix, samples new matches on request, and resets main and league exploiters. Additional evaluator workers (running on the CPU) are used to supplement the payoff estimates. See Extended Data Fig. 6 for an overview of the training setup."

2.2 Market solution of the Stochastic Growth Model

- Do it

3 Scaling up: Heterogenous Agents

Intro

3.1 Centralized learning, decentralized execution

- Explain centralized critic algorithm.
- Explain parallelized computation framework.
  - Espeholt et al. [2018] develop the IMPALA framework, which is dfesigned to learn in parallel in a stable and efficient way.

3.2 Solving the Krussel and Smidt framework

Present model and solutions

4 Multi-Market Environemnts: The Economy Constructor

- Explain economy constructor.
- Example 1: Two-sector model with labor.
- Explain markets.

References

Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. Emergent tool use from multi-agent autocurricula. *arXiv preprint arXiv:1909.07528*, 2019.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Dimitri P Bertsekas and Huizhen Yu. Q-learning and enhanced policy iteration in discounted dynamic programming. *Mathematics of Operations Research*, 37(1):66–94, 2012.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pages 1407–1416. PMLR, 2018.

Qingyin Ma and John Stachurski. Dynamic programming deconstructed: Transformations of the bellman equation and computational efficiency. *Operations Research*, 2021.

Qingyin Ma, John Stachurski, and Alexis Akira Toda. Unbounded dynamic programming via the q-learning transform. *arXiv preprint arXiv:2012.00219*, 2020.

John N Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine learning*, 16(3):185–202, 1994.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.