Distributional Temporal Difference Learning for Finance: Dealing with Leptokurtic Rewards

Shijie Huang, Nitin Yadav & Peter Bossaerts

Brain, Mind and Markets Laboratory

Time Series and Forecasting Symposium
The University of Sydney
November 2019

Table of Contents

1. Reinforcement Learning
2. RL in finance: our experiment
3. Results

Table of Contents

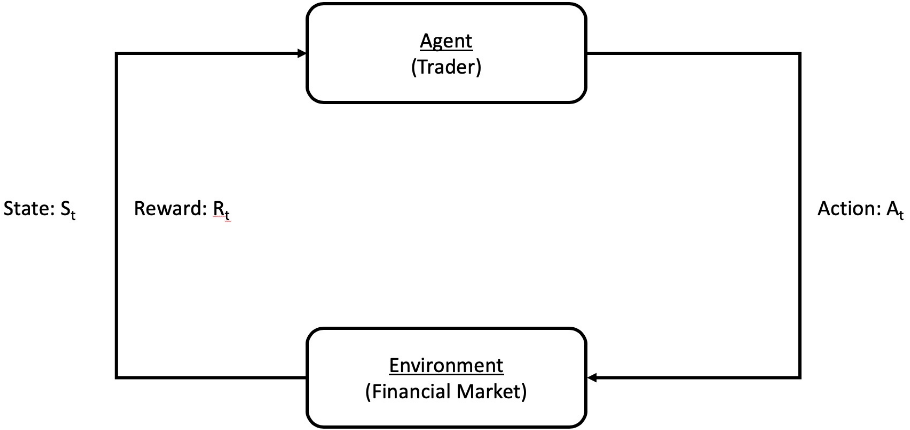
1. Reinforcement Learning

2. RL in finance: our experiment

3. Results

An overview on reinforcement learning (RL)

Brain, Mind and Markets Laboratory



An overview on reinforcement learning (RL)

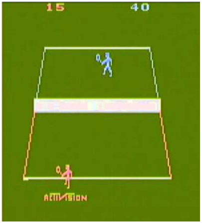


Figure: Atari games (2013) [6]

Figure: Board games (Nature, 2016) [8]

Figure: Poker (Science, 2019) [2]

Figure: RTS games (Nature, 2019) [9]

An overview on reinforcement learning (RL)

Brain, Mind and Markets Laboratory

- We model the interaction as a stationary Markov Decision Process (MDP): (S, A, R, P)
- S: set of states, A: set of actions
- $R: S \times A \rightarrow F$ is a reward function that maps each state-action pair to a reward that lives in outcome space F
- $P(s^{'}|s,a)$: state transition probability distribution
- Policy $\pi(a|s)$: the probability of action a in state s

Bellman principle

Brain, Mind and Markets Laboratory

In economics and finance, we intend to find an optimal value of a state $V(s)$ where

$$V(s) = \max_a \left\{ \mathbb{E}[R|s,a] + \gamma \mathbb{E}[V(s^{'})|s,a] \right\}$$

- We do not know the functional form of V, nor do we know the expectations

Reinforcement Learning provides a solution: we start from state-action "Q" values

$$Q(s,a) = \left\{ \hat{\mathbb{E}}[R|s,a] + \gamma \hat{\mathbb{E}}[Q(s^{'},a^{'})|s,a] \right\}$$

- The question: how do we update Q values dynamically, and ensure that the optimal Q values converge to the Bellman values?

The SARSA approach

- **SARSA** stands for "State, Action, Reward, Next State, Next Action"
- When the agent is in state s, she chooses actions a that is the optimal one (according to learned Q values).
- This generates a reward R, next state $s^{'}$ and she uses this to update Q for state s and action a recursively:

$$Q(s,a) \leftarrow Q(s,a) + \alpha(R + \gamma Q(s^{'}, a^{'}) - Q(s,a)),$$

  where α: learning rate (step size) and γ: discount rate
- With "sufficient" exploration (enough visits in all state-action pairs), the value function estimation converges to true value function [10].

Distributional reinforcement learning (2017-)

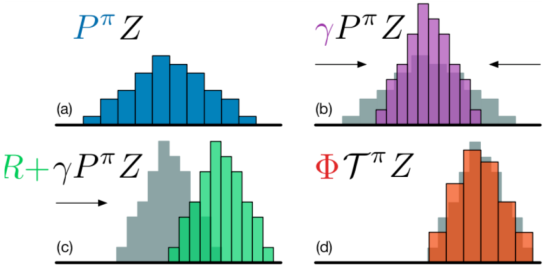
Brain, Mind and Markets Laboratory

Rather than using recursive estimation, we keep track of the **entire distribution** (history) of realized Q values for a state s and action a and take the weighted average to get an estimate of $Q(s, a)$ (weights = discounting the past)

DRL: How to estimate the distribution of Qs?

Brain, Mind and Markets Laboratory

- Categorical: parametric densities fit to probability histograms [1]



- Quantile: parametric fitting of quantiles in cumulative density function [4, 5]
- Expectile: parametric fitting of expectiles in cumulative density function [7]

The Q value is then obtained as a simple integration over the estimated distribution (e.g. in categorical approach, $Q(s,a) = \sum_i z_i p_i(s,a)$ where $p_i(s,a)$ is the estimated probability that Q takes the value z_i)

Efficient estimation and DRL

Brain, Mind and Markets Laboratory

- If we keep track of the entire distribution anyway, why not use the BEST estimation of the expectation?!
- This may be beneficial, particularly in the context of finance, where rewards are heavy tailed (leptokurtosis), and hence the simple average is not an "efficient" estimator.

(Efficient estimator: reaches the Cramér-Rao lower bound, or Chapman-Robbins lower bound [3].)

Leptokurtic reward

Brain, Mind and Markets Laboratory

- Leptokurtosis: daily returns on the S&P500 index generate a distribution with a kurtosis of 10 or higher ("heavy tails")
- Neither SARSA nor distributional RL are "well-behaved" under leptokurtic rewards (see evidence later)
  - Neither method obtains optimal state-action values $Q(s,a)$ due to frequent reward outliers
- *There is more*: Q value updating rule uses the sum of two random variables with (asymptotically) very different distributions:
  1. Rewards are (always) leptokurtic
  2. Realized Q values eventually will only depend on state transitions and actions, so their distributions are NOT leptokurtic (provided state transitions are not)

Our approach: efficient estimation

Brain, Mind and Markets Laboratory

We utilize efficient estimation from mathematical statistics:

- Instead of processing the reward R directly, we estimate the mean reward, and perform standard SARSA with estimated mean reward.

$$Q(s,a) \leftarrow Q(s,a) + \alpha(\hat{\mathbb{E}}[R|s,a] + \gamma Q(s^{'},a^{'}) - Q(s,a))$$

- We use $\hat{\mathbb{E}}[R|s,a]$ instead of (one) realized R; We compare two versions of $\hat{E}$:
  1. **d-RL**: sample average of past n rewards:

  $$\hat{\mathbb{E}}(R(s,a)) = \frac{1}{n}\sum_{i=0}^{n} R_i$$

  2. **d-RL-MLE**: We use MLE, which reaches the Cramér-Rao lower bound (under certain conditions; if MLE is not efficient, we use something better)

MLE

- We need a good assumption about the nature of the reward/return distribution; We use student-t with low degrees of freedom.
- MLE estimator of the mean (reward) is complex; we use the Expectation-Maximization (EM) algorithm
  - MLE does not merely "truncate" samples!!
  - Instead, MLE projects outliers towards the middle.

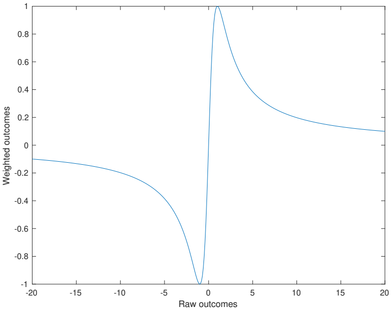


Table of Contents

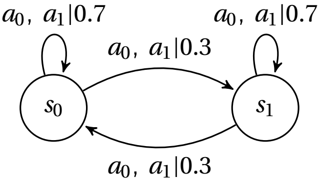
Brain, Mind and Markets Laboratory

1. Reinforcement Learning
2. RL in finance: our experiment
3. Results

Environment: financial decision making task

Brain, Mind and Markets Laboratory

- Two-states, two-actions MDP
- State transition is exogenous: $P(s^{'}|s,a) = P(s^{'}|s)$



$R_t(s,a) \sim$	Gaussian		Leptokurtic		S&P500	
	s_0	s_1	s_0	s_1	s_0	s_1
a_0	$\boldsymbol{N_{0,1}+2}$	$N_{0,1}+1$	$\boldsymbol{T_{1.1}+1.5}$	$T_{1.1}+1$	$\boldsymbol{\mu_d+0.5}$	μ_d
a_1	$N_{0,1}+1$	$\boldsymbol{N_{0,1}+2}$	$T_{1.1}+1$	$\boldsymbol{T_{1.1}+1.5}$	μ_d	$\boldsymbol{\mu_d+0.5}$

Table of Contents

Brain, Mind and Markets Laboratory

1 Reinforcement Learning

2 RL in finance: our experiment

3 Results

"Episodes" in Machine Learning

Brain, Mind
and Markets
Laboratory

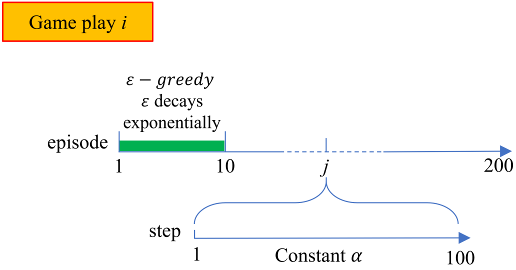
- Step/trial = each frame
- Episode: all steps until pole falls
- Game: the entire play (after pulling up pole again repeatedly).

Pole Balancing

(https://www.youtube.com/watch?v=46wjA6dqxOM)

Results

- 100 game plays (i). $\alpha = 0.1$, constant over all episodes; Agent explores (chooses random action) with probability ε in the first 10 episodes
- We check at the end of each episode *after* exploration whether the agent learns the correct policy (s_0, a_0) and (s_1, a_1); If the agent passes all 190 "checks" in a game play i, we deem it one "optimal policy convergence"
- We count how many games the agents obtain such convergence



Results

Brain, Mind and Markets Laboratory

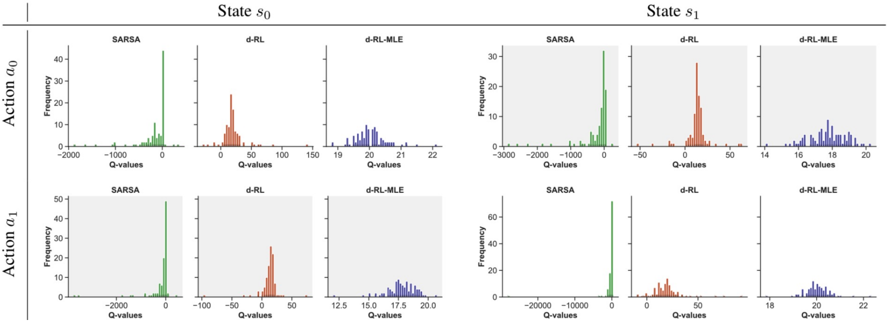
% of optimal policy convergence	Gaussian $N(0,1)$	Student-t $(\upsilon = 1.1)$	Empirical S&P500
SARSA	82%	2%	47%
dis-RL-Cat	80%	4%	-
d-RL	**100%**	33%	95%
d-RL-MLE	**100%**	**95%**	**97%**

Table: Percentage of game plays where the agent computed the optimal policy at the end of *all* 190 episodes, averaged across two states.

Contamination of estimated distribution by outliers except in the case of d-RL-MLE

Brain, Mind and Markets Laboratory

- In an environment with leptokurtic rewards, it is likely that a huge outlier reward (+ve/-ve) can dominate $Q(s,a)$ updates, hence causing the robots to switch between optimal and sub-optimal actions overtime.



Conclusions

Brain, Mind and Markets Laboratory

- Distributional RL can be exploited to obtain the most efficient way to estimate mean action-values across states, and hence, enhance control.
- In a leptokurtic environment, availability of the distribution allows one to estimate the mean in a more efficient way, ensuring that outliers do not cause policy non-convergence or policy instability.
- From a broader perspective, our results show the importance of bringing prior domain-specific knowledge to machine learning algorithms using the tools of mathematical statistics.
  - A non-leptokurtic example: If the reward distribution is shifted-exponential, then the best estimation of the expected reward given an action and a state is to using the running Minimum/Maximum, not the sample average!

References

Brain, Mind and Markets Laboratory

[1] M. G. Bellemare, W. Dabney, and R. Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 449–458. JMLR.org, 2017.

[2] N. Brown and T. Sandholm. Superhuman ai for multiplayer poker. *Science*, 365(6456):885–890, 2019.

[3] G. Casella and R. L. Berger. *Statistical Inference*, volume 2. Duxbury Pacific Grove, CA, 2002.

[4] W. Dabney, G. Ostrovski, D. Silver, and R. Munos. Implicit quantile networks for distributional reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, 2018.

[5] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos. Distributional reinforcement learning with quantile regression. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[7] M. Rowland, M. G. Bellemare, W. Dabney, R. Munos, and Y. W. Teh. An analysis of categorical distributional reinforcement learning. *arXiv preprint arXiv:1802.08163*, 2018.

[8] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[9] O. Vinyals, T. Ewalds, S. Bartunov, P. Georgiev, A. S. Vezhnevets, M. Yeo, A. Makhzani, H. Küttler, J. Agapiou, J. Schrittwieser, et al. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*, 2017.

[10] C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.