Reproducible, Reusable, and Robust
Reinforcement Learning

Joelle Pineau
Facebook AI Research, Montreal
School of Computer Science, McGill University

Neural Information Processing Systems (NeurIPS)
December 5, 2018
“Reproducibility refers to the ability of a researcher to duplicate the results of a prior study….

Reproducibility is a minimum necessary condition for a finding to be believable and informative.”

Reproducibility

Reusability

Robustness

Using the same materials as were used by the original investigator.

Bollen et al.
National Science Foundation, 2015.
Reproducibility crisis in science (2016)

[Diagram showing survey results]

- 3% No, there is no crisis
- 38% Yes, a slight crisis
- 7% Don't know
- 52% Yes, a significant crisis

1,576 researchers surveyed

https://www.nature.com/news/1-500-scientists-lift-the-lid-on-reproducibility-1.19970
Reproducibility crisis in science (2016)

https://www.nature.com/news/1-500-scientists-lift-the-lid-on-reproducibility-1.19970
Reinforcement learning (RL)

- Very general framework for sequential decision-making!
- Learning by trial-and-error, from sparse feedback.
- Improves with experience, in real-time.

Learn $\pi = \text{strategy to find this cheese!}$
Impressive successes in games!
RL applications beyond games

- Robotics
- Video games
- Conversational systems
- Medical intervention
- Algorithm improvement
- Crop management
- Personalized tutoring
- Energy trading
- Autonomous driving
- Prosthetic arm control
- Forest fire management
- Financial trading
- Many more!
Adaptive neurostimulation

Panuccio, Guez, Vincent, Avoli, Pineau, Exp Neurol, 2013
RL in simulation $\rightarrow$ RL in real-world

from $\sim 10^1 - 10^2$ trials
25+ years of RL papers

RL via Policy gradient methods

Maximize expected return, \( \rho(\theta, s_0) = E[r_0 + r_1 + \ldots + r_T | s_0] \)

using gradient ascent:

\[
\frac{\delta \rho(\theta, s_0)}{\delta \theta} = \sum_s \mu_{\pi_\theta}(s | s_0) \sum_a \frac{\delta \pi_\theta(a | s)}{\delta \theta} Q_{\pi_\theta}(s, a)
\]

state distribution

value fn
Policy gradient papers

- Evolution-Guided Policy Gradient in Reinforcement Learning
- On Learning Intrinsic Rewards for Policy Gradient Methods
- Evolved Policy Gradients
- Policy Optimization via Importance Sampling
- Dual Policy Iteration
- Post: Device Placement with Cross-Entropy Minimization and Proximal Policy Optimization
- Genetic-Gated Networks for Deep Reinforcement Learning
- Simple random search of static linear policies is competitive for reinforcement learning
- Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models
- ....

Many more at ICLR’18, ICML’18, AAAI’18, EWRL’18, CoRL’18, …

Most papers use same policy gradient baseline algorithms.
Policy gradient baseline algorithms

Same standard baselines used in all of these papers:

» Trust Region Policy Optimization (TRPO), Schulman et al. 2015.


» Deep Deterministic Policy Gradients (DDPG), Lillicrap et al. 2015.

» Actor-Critic Kronecker-Factored Trust Region (ACKTR), Wu et al. 2017.
Robustness of policy gradient algorithms

Consider Mujoco simulator:

Video taken from: https://gym.openai.com/envs/HalfCheetah-v1
Robustness of policy gradient algorithms

Consider Mujoco simulator:
Codebase comparison

TRPO implementations:

GitHub - joscult/modular rtc: Implementation of TRPO and related ...
https://github.com/joscult/modular rtc
This library is written in a modular way to allow for sharing code between TRPO and PPO variants, and to write the same code for different kinds of action spaces. Dependencies: keras (1.0.1); tensorflow (0.8.2); matplotlib; numpy; scipy. To run the algorithms implemented here, you should put modular rtc on your PYTHONPATH ...

GitHub - wojaremba/trpo
https://github.com/wojaremba/trpo
Join GitHub today. GitHub is home to over 20 million developers working together to host and review code, manage projects, and build software together. Sign up. No description, website, or topics provided. 12 commits 1 branch 0 releases Fetching contributors Python 100.0%. Python. Clone or download ...

GitHub - pet-coady/trpo: Trust Region Policy Optimization with ...
https://github.com/pet-coady/trpo
The exact code used to generate the OpenAI Gym submissions is in the algo Gym evaluation branch. Here are the key points: Proximal Policy Optimization (similar to TRPO, but uses gradient descent with KL loss terms) [1][2]. Value function approximated with 3 hidden-layer NN (tanh activations), hid1 size = obs_dim x 10 ...

GitHub - kvfrans/parallel-trpo: A parallel version of Trust Region Policy ...
https://github.com/kvfrans/parallel-trpo
README.md parallel-trpo: A parallel implementation of Trust Region Policy Optimization on environments from OpenAI gym. Now includes hyperparameter adaptation as well! More more info, check my post on this project. I'm working towards the ideas at this openAI research request. The code is based on this ...

GitHub - jkxlee88/trpo: trust region policy optimization base on gym and ...
https://github.com/jkxlee88/trpo
trust region policy optimization base on gym and tensorflow. There are three versions of trpo, one for discrete action space like mountaincar, one for discrete action space task with image as input like atari games, and the last for continuous action space for pendulum. The environment is base on openAI gym, part of code ...

GitHub - woonsangcho/trpo: Trust Region Policy Optimization ...
https://github.com/woonsangcho/trpo
README.md, Proximal Policy Optimization Implementation using Tensorflow and Keras. Code written by Kakaon Cho (Woon Sang Cho): https://github.com/woonsangcho. Summary: This is an implementation of Proximal Policy Optimization (PPO)[1][2], which is a variant of Trust Region Policy Optimization (TRPO)[3]

GitHub - yhong88/TRPO-GAE: Trust Region Policy Optimization with ...
https://github.com/yhong88/TRPO-GAE
GitHub is where people build software. More than 27 million people use GitHub to discover, fork, and contribute to over 80 million projects.
Codebase comparison

TRPO implementations:

GitHub - joschu/modular-rf: Implementation of TRPO and related...
https://github.com/joschu/modular-rf
This library is written in a modular way to allow for sharing code between TRPO and PPO variants, and to write the same code for different kinds of action spaces. Dependencies: keras (1.0.1); theano (0.8.2); tensorflow; numpy; scipy. To run the algorithms implemented here, you should put modular_rf on your PYTHONPATH...

GitHub - wojaremba/trpo
https://github.com/wojaremba/trpo
Join GitHub today. GitHub is home to over 20 million developers working together to host and review code, manage projects, and build software together. Sign up. No description, website, or topics provided. 12 commits · 1 branch · 0 releases · Fetching contributors · Python 100.0%. Python. Clone or download...

GitHub - pat-coady/trpo: Trust Region Policy Optimization with...
https://github.com/pat-coady/trpo
The exact code used to generate the OpenAI Gym submissions is in the algo_gym_evaluation branch. Here are the key points: Proximal Policy Optimization (similar to TRPO), but uses gradient descent with KL loss terms [1][2]. Value function approximated with 3 hidden-layer NN (tanh activations). implementation_size = obs_dim x 10...

GitHub - kvfrans/parallel-trpo: A parallel version of Trust Region Policy...
https://github.com/kvfrans/parallel-trpo
README.md: parallel-trpo: A parallel implementation of Trust Region Policy Optimization on environments from OpenAI gym. Now includes hyperparameter adaptation as well! More more info, check my post on this project. I'm working towards the ideas at this openAI research request. The code is based off of this...

GitHub - jkike888/trpo: trust region policy optimization base on gym and...
https://github.com/jkike888/trpo
trust region policy optimization base on gym and tensorflow. There are three versions of trpo, one for discrete action space like mountaincar; one for discrete action space task with image as input like atari games, and the last for continuous action space for pendulums. The environment is base on openAI gym, part of code...

GitHub - woonsangcho/trpo: Trust Region Policy Optimization...
https://github.com/woonsangcho/trpo
README.md: Proximal Policy Optimization Implementation using Tensorflow and Keras. Code written by Gaken Cho (Woon Sang Cho): https://github.com/woonsangcho. Summary. This is an implementation of Proximal Policy Optimization (PPO) [1][2], which uses a variant of Trust Region Policy Optimization (TRPO) [3].

GitHub - yihong88/TRAPO-GAE: Trust Region Policy Optimization with...
https://github.com/yihong88/TRAPO-GAE
GitHub is where people build software. More than 27 million people use GitHub to discover, fork, and contribute to over 80 million projects.
Effect of hyperparameter configurations

Policy network structure:

Unit activation:
An intricate interplay of hyperparameters!

How motivated are we to find the best hyperparameters for our baselines?
Fair comparison is easy, right?

Same amount of data.

Same amount of computation.
Let’s look a little closer

n=5

n=5
Let’s look a little closer

Both are same TRPO code with best hyperparameter configuration!
How should we measure performance of the learned policy?

- Average return over test trials?

\[ \bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i \]

- Confidence interval?

\[ \bar{X} \pm 1.96 \frac{\sigma}{\sqrt{n}} \]

How do we pick \( n \)?
How many trials?

<table>
<thead>
<tr>
<th>Work</th>
<th>Number of Trials</th>
</tr>
</thead>
<tbody>
<tr>
<td>(et al. 2016)</td>
<td>top-5</td>
</tr>
<tr>
<td>(et al. 2017)</td>
<td>3-9</td>
</tr>
<tr>
<td>(et al. 2016)</td>
<td>5 (5)</td>
</tr>
<tr>
<td>(et al. 2017)</td>
<td>3</td>
</tr>
<tr>
<td>(et al. 2015b)</td>
<td>5</td>
</tr>
<tr>
<td>(et al. 2015a)</td>
<td>5</td>
</tr>
<tr>
<td>(et al. 2017)</td>
<td>top-2, top-3</td>
</tr>
</tbody>
</table>
Consider the case of $n=10$
Consider the case of $n=10$

- Strong positive bias: seems to beat the baseline!
- Variance appears much smaller.

Top-3 results
Reinforcement Learning never worked, and 'deep' only helped a bit.

FEBRUARY 23, 2018

When someone asks me if RL works, I tell them it doesn't

AND 70% OF THE TIME, I'M RIGHT

https://www.alexirpan.com/2018/02/14/rl-hard.html
From **fair** comparisons…

- Different methods have distinct sets of hyperparameters.
- Different methods exhibit variable sensitivity to hyperparams.
- What method is best often depends on data/compute budget.

To **robust** conclusions.
We surveyed 50 RL papers from 2018 (published at NeurIPS, ICML, ICLR)

Yes:

- Paper has experiments 100%
- Paper uses neural networks 90%
- All hyperparams for proposed algorithm are provided. 90%
- All hyperparams for baselines are provided. 60%
- Code is linked. 55%
- Method for choosing hyperparams is specified 20%
- Evaluations on some variation of a hold-out test set 10%
- Significance testing applied 5%
We surveyed 50 RL papers from 2018 (published at NeurIPS, ICML, ICLR)

- Paper has experiments: 100%
- Paper uses neural networks: 90%
- All hyperparams for proposed algorithm are provided: 90%
- All hyperparams for baselines are provided: 60%
- Code is linked: 55%
- Method for choosing hyperparams is specified: 20%
- Evaluations on some variation of a hold-out test set: 10%
- Significance testing applied: 5%

Let’s add a little shade!
How about a reproducibility checklist?
How about a reproducibility checklist?

For all algorithms presented, check if you include:
- A clear description of the algorithm.
- An analysis of the complexity (time, space, sample size) of the algorithm.
- A link to downloadable source code, including all dependencies.

For any theoretical claim, check if you include:
- A statement of the result.
- A clear explanation of any assumptions.
- A complete proof of the claim.
How about a reproducibility checklist?

For all **algorithms** presented, check if you include:
- A clear description of the algorithm.
- An analysis of the complexity (time, space, sample size) of the algorithm.
- A link to downloadable source code, including all dependencies.

For any **theoretical claim**, check if you include:
- A statement of the result.
- A clear explanation of any assumptions.
- A complete proof of the claim.

For all **figures** and **tables** that present empirical results, check if you include:
- A complete description of the data collection process, including sample size.
- A link to downloadable version of the dataset or simulation environment.
- An explanation of how sample were allocated for training / validation / testing.
- An explanation of any data that was excluded.
- The range of hyper-parameters considered, method to select the best hyper-parameter configuration, and specification of all hyper-parameters used to generate results.
- The exact number of evaluation runs.
- A description of how experiments were run.
- A clear definition of the specific measure or statistics used to report results.
- Clearly defined error bars.
- A description of results including **central tendency** (e.g. mean) and **variation** (e.g. stddev).
- The computing infrastructure used.
The role of infrastructure on reproducibility
The role of infrastructure on reproducibility
Myth or fact?

*Reinforcement Learning is the only case of ML where it is acceptable to test on your training set.*
Myth or fact?

*Reinforcement Learning is the only case of ML where it is acceptable to test on your training set.*

Classical RL
*Train/test on same task.*

AGI
*Test on anything!*

The RL generalization roadmap
Myth or fact?

Reinforcement Learning is the only case of ML where it is acceptable to test on your training set.

The RL generalization roadmap

Classical RL
Train/test on same task.

Separate tasks for train / test

AGI
Test on anything!
Myth or fact?

*Reinforcement Learning is the only case of ML where it is acceptable to test on your training set.*

**Classical RL**  
Train/test on same task.  
Separate rnd seeds for train / test  
Separate tasks for train / test

**AGI**  
Test on anything!

The RL generalization roadmap
Generalization in RL

\[
\mathbb{E}_{\text{err}} = \frac{1}{N} \sum_{N} R(s_t | s_0 \sim s_{\text{tr},i}) - \frac{1}{M} \sum_{M} R(s_t | s_0 \sim s_{\text{test},i})
\]
Generalization in RL

\[
\text{Err} = \frac{1}{N} \sum_N R(s_t | s_0 \sim s_{tr,i}) - \frac{1}{M} \sum_M R(s_t | s_0 \sim s_{test,i})
\]

Standard RL Acrobot simulator
Generalization in RL

\[ \text{Err} = \frac{1}{N} \sum_{i=1}^{N} R(s_t|s_0 \sim s_{tr,i}) - \frac{1}{M} \sum_{i=1}^{M} R(s_t|s_0 \sim s_{test,i}) \]

Standard RL Acrobot simulator

From JC Gamboa Higuera, D. Meger, G. Dudek, ICRA’17.
Natural world has incredible complexity!
Many RL benchmarks are ridiculously simple!

- Low-dim state space (Mujoco)
- Small number of actions (ALE)
- Few initial states
- Deterministic transitions and rewards
- Short description length, e.g. <100KB.

Easy to memorize! Brittle to perturbations.
Natural world  =>  RL simulation

Lantana camara!

RL actions
Natural world => RL simulation

Lantana camara!
Real-world video => RL simulation

Breakout (Atari)

Zhang, Wu, Pineau, 2018
Real-world video => RL simulation

What is going on?

• Add random video in background:
  “natural” noise + game strategy.

• Different train/test video
  => clear train/test separation.

• Fast and plentiful data acquisition.

• Easy replication and comparison.

Breakout (Atari)
Multi-task RL in Photorealistic Simulators

Whelan et al., 2018 (Facebook Reality Labs)

Colleagues at FAIR + Georgia Tech + FRL
Myth or fact?

*Reinforcement Learning is the only case of ML where it is acceptable to test on your training set.*

**Not necessarily!**

Classical RL

- Train/test on same task.

Separate
- rnd seeds for train / test
- image/video background
- multi-task photorealistic simulator

AGI

- Test on anything!

The RL generalization roadmap
Step out into the real-world!
Science is a collective institution that aims to understand and explain.
Science is a collective institution that aims to understand and explain.
ICLR Reproducibility Challenge
Second Edition, 2019

Welcome to the 2nd edition of ICLR reproducibility challenge! One of the challenges in machine learning research is to ensure that published results are reliable and reproducible. In support of this, the goal of this challenge is to investigate reproducibility of empirical results submitted to the 2019 International Conference on Learning Representations. We are choosing ICLR for this challenge because the timing is right for course-based participants (see below), and because papers submitted to the conference are automatically made available publicly on Open Review.

Task Description

You should select a paper from the 2019 ICLR submissions, and aim to replicate the experiments described in the paper. The goal is to assess if the experiments are reproducible, and to determine if the conclusions of the paper are supported by your findings. Your results can be either positive (i.e. confirm reproducibility), or negative (i.e. explain what you were unable to reproduce, and potentially explain why).

Essentially, think of your role as an inspector verifying the validity of the experimental results and conclusions of the paper. In some instances, your role will also extend to helping the authors improve the quality of their work and paper.
Major Contributors:

RL Reproducibility:

Peter Henderson  Phil Bachman  Riashat Islam  Doina Precup  Joshua Romoff  David Meger

Natural RL:

Amy Zhang  Nicolas Ballas  Yuxin Wu

Reproducibility Challenge:

G. Fried  R. Nan Ke  H. Larochelle  K. Sinha

MILA (RLab) @ McGill
Thank you!