Evaluation in Reinforcement Learning

Is the RL method working?

Cathy Wu

6.7920 Reinforcement Learning: Foundations and Methods

Readings

1. Lecture Appendices A-C (see end of slides)

Outline

1. Challenge 1: RL is quite sensitive
2. Solution 1: Standardize RL evaluation
3. Challenge 2: Overfitting to benchmarks
4. Solution 2: Explicitly model generalization

Outline

1. **Challenge 1: RL is quite sensitive**
   a. Performance evaluation in RL
   b. Sensitivity analysis of RL
2. Solution 1: Standardize RL evaluation
3. Challenge 2: Overfitting to benchmarks
4. Solution 2: Explicitly model generalization

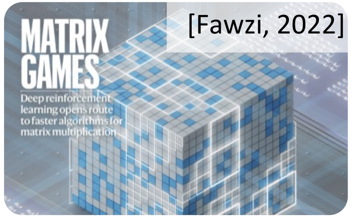
Performance evaluation in RL

- Consider a method (A) and method + modification (A+X)
  - Ex. NPG vs NPG + Truncation
  - Ex. TNPG vs TNPG + KL line search
- How to figure out which is better?
- HW: Emulate performance evaluation in RL

When RL works, it’s great

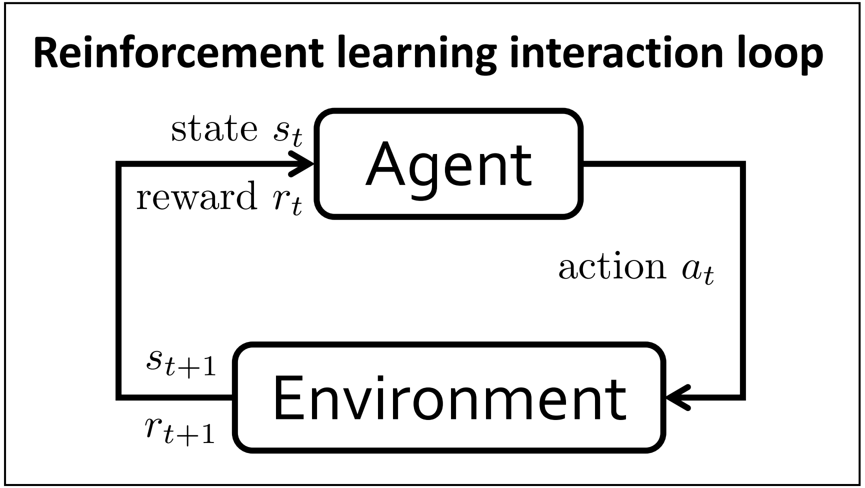
[Bellemare, 2020]

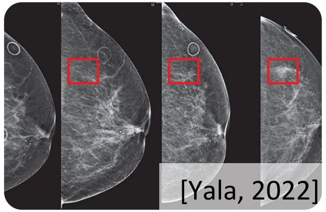
Google Loon



[Fawzi, 2022]

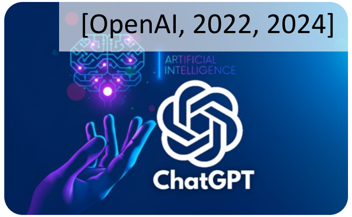
AlphaTensor





[Yala, 2022]

Breast cancer screening



[OpenAI, 2022, 2024]

ChatGPT

[Feng, 2023]

AV safety validation

[Kaufmann, 2023]

Drone racing

But, RL is highly sensitive

- Network architecture
- Inherited codebase
- Code-level optimizations
- Tasks (benchmarks)
- Random seed
- Method hyperparameters
- MDP specification (observation, discount rate, frame skip, etc.)

See Lecture Appendix C for more examples

Example: Sensitivity of RL to network architecture

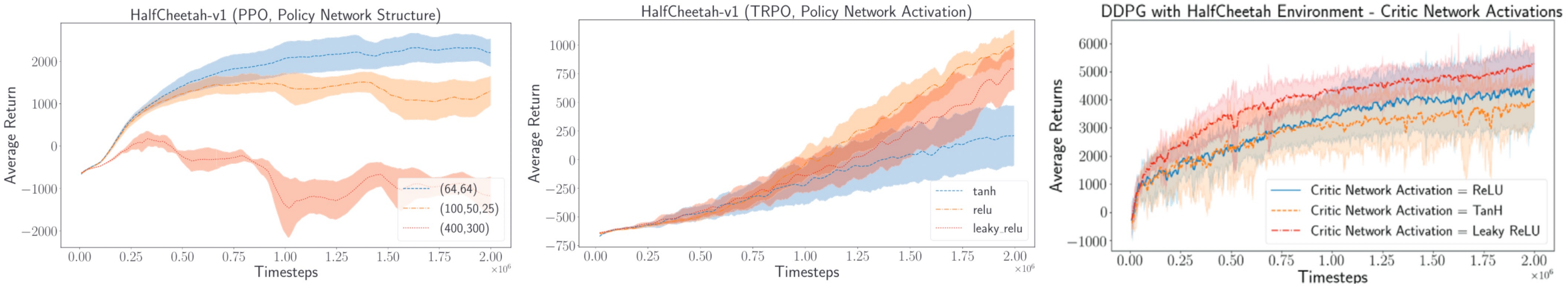


Figure 2: Significance of Policy Network Structure and Activation Functions PPO (left), TRPO (middle) and DDPG (right).

Half Cheetah

Henderson, Islam, Bachman, Pineau, Precup, Meger, **Deep Reinforcement Learning That Matters**, *AAAI Conference on Artificial Intelligence*, 2018.

Example: Sensitivity of RL to codebase

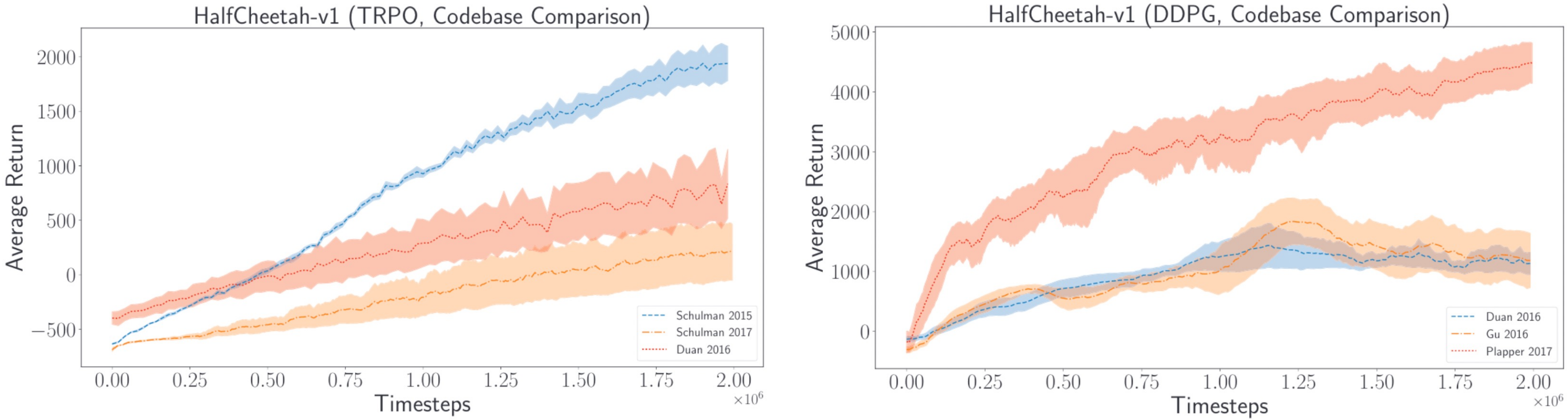


Figure 6: TRPO codebase comparison using our default set of hyperparameters (as used in other experiments).

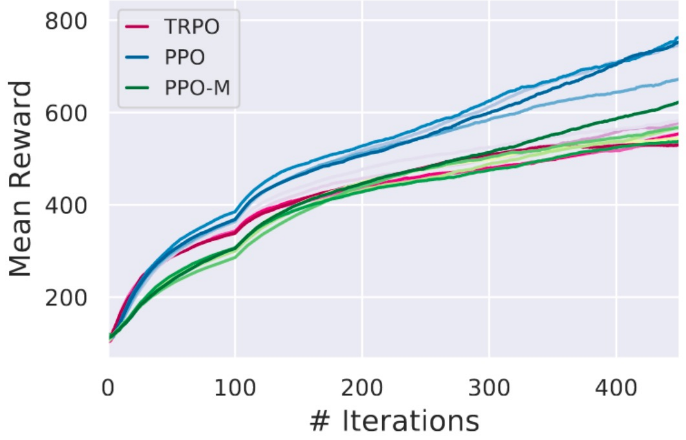
Half Cheetah

Henderson, Islam, Bachman, Pineau, Precup, Meger, **Deep Reinforcement Learning That Matters**, *AAAI Conference on Artificial Intelligence*, 2018.

Example: Sensitivity of RL to code-level optimizations

- Does PPO outperform TRPO?
- Yes and No
- More like: PG + clipping + code-level optimizations >> TRPO
- PPO-M: PPO minus code-level optimizations*
  - *Code-level optimizations: value function clipping, reward scaling, orthogonal initialization & layer scaling, Adam learning rate annealing
- PPO-NoClip: PPO minus clipping

Mujoco Humanoid-v2



	WALKER2D-V2	HOPPER-V2	HUMANOID-V2
PPO	3292 [3157, 3426]	2513 [2391, 2632]	806 [785, 827]
PPO (BASELINES)	3424	2316	—
PPO-M	2735 [2602, 2866]	2142 [2008, 2279]	674 [656, 695]
PPO-NOCLIP	2867 [2701, 3024]	2371 [2316, 2424]	831 [798, 869]

PPO-NoClip >> PPO-M

L. Engstrom et al., “Implementation Matters in Deep Policy Gradients: A Case Study on PPO and TRPO,” in ICLR, 2020.

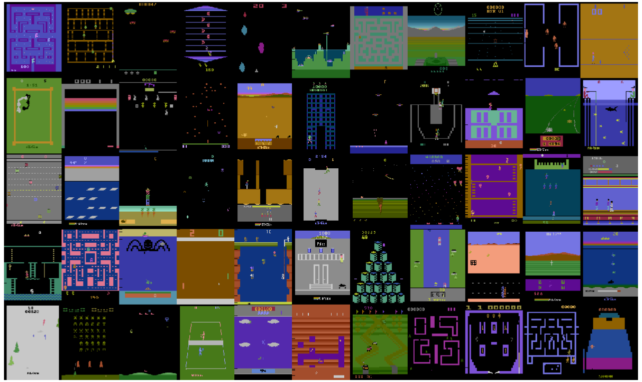
Outline

1. Challenge 1: RL is quite sensitive
2. **Solution 1: Standardize RL evaluation**
    a. Standard benchmarks
    b. Standard libraries
    c. Standard statistical & experimental design techniques
3. Challenge 2: Overfitting to benchmarks
4. Solution 2: Explicitly model generalization

Strategies for handling RL sensitivity

- Control for as many factors as possible
  - Standardize, standardize, standardize
  - Benchmarks
  - RL codebases
  - Hyperparameter tuning
  - …
- Run multiple trials for statistical confidence
  - How many?

Standard benchmarks



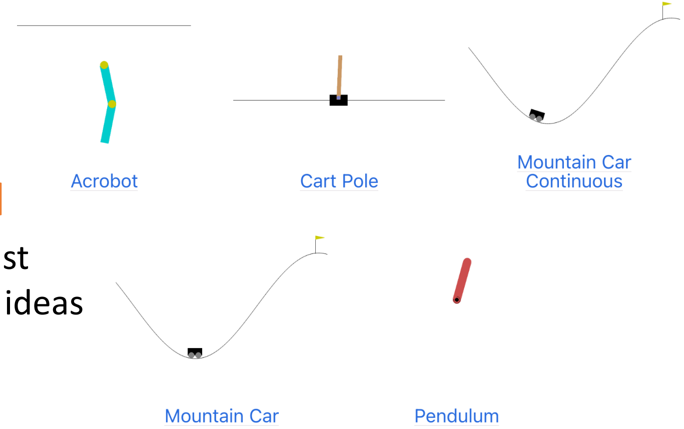
Arcade Learning Environment (ALE)

50+ Atari 2600 games

Various RL challenges: pixel learning, model learning, model-based planning, imitation learning, transfer learning, and intrinsic motivation, exploration, etc.

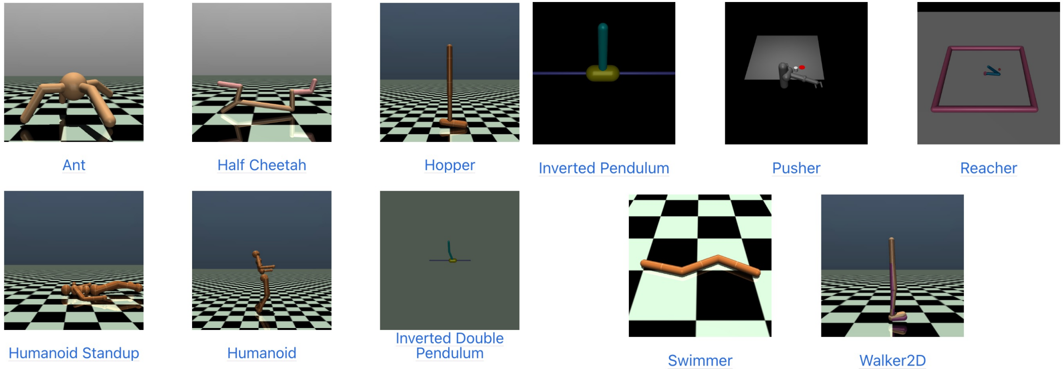
Classic control

Sanity checking, fast experiments, develop ideas



Multi-Joint dynamics with Contact (Mujoco)

Continuous control, high-dimensional state spaces

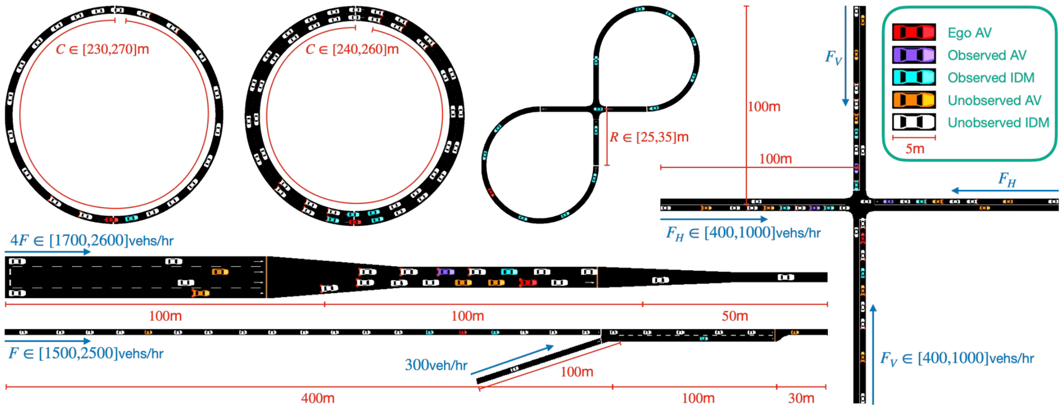


"Gymnasium Documentation." 4 Nov. 2024, gymnasium.farama.org.
M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The Arcade Learning Environment: An Evaluation Platform for General Agents," *JAIR*, 2013, doi: 10.1613/jair.3912.
Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *ICML*, 2016.

Many available benchmarks & tools

Third-party environments with Gymnasium [1]

- Autonomous Driving
- Biological / Medical
- Economic / Financial
- Electrical / Energy
- Game
- Mathematics / Computational
- Robotics
- Telecommunication Systems
- …



Framework for building custom traffic environments using SUMO traffic simulator [2]

[1] "Gymnasium Documentation." 4 Nov. 2024, gymnasium.farama.org/environments/third_party_environments.
[2] Z. Yan, A. R. Kreidieh, E. Vinitsky, A. M. Bayen, and **C. Wu**, "Unified automatic control of vehicular systems with reinforcement learning," *IEEE T-ASE, 2022*, doi: 10.1109/TASE.2022.3168621. Github: github.com/mit-wu-lab/automatic_vehicular_control.

Common RL method libraries

RL Platform	Documentation	Code Coverage	Type Hints	Last Update	Forks	Stars
Stable-Baselines3	docs passing	coverage 96.00%	✓	last update last monday	1.7k	9.1k
Ray/RLlib	docs passing	— (1)	✓	last update today	5.7k*	33.8k*
SpinningUp	docs passing	✗	✗	last update february 2020	2.2k	10.1k
Dopamine	docs passing	✗	✗	last update last monday	1.4k	10.6k
ACME	docs passing	— (1)	✓	last update october	426	3.5k
Sample Factory	—	codecov 78%	✗	last update october	111	822
Tianshou	docs passing	coverage 85%	✓	last update october	1.1k	7.9k
CleanRL					639	5.6k

(1): it has continuous integration but the coverage rate is not available

* Includes parent library Ray

"tianshou." GitHub, 6 Nov. 2024, github.com/thu-ml/tianshou/#comprehensive-functionality.

A typical number of runs

5 or less

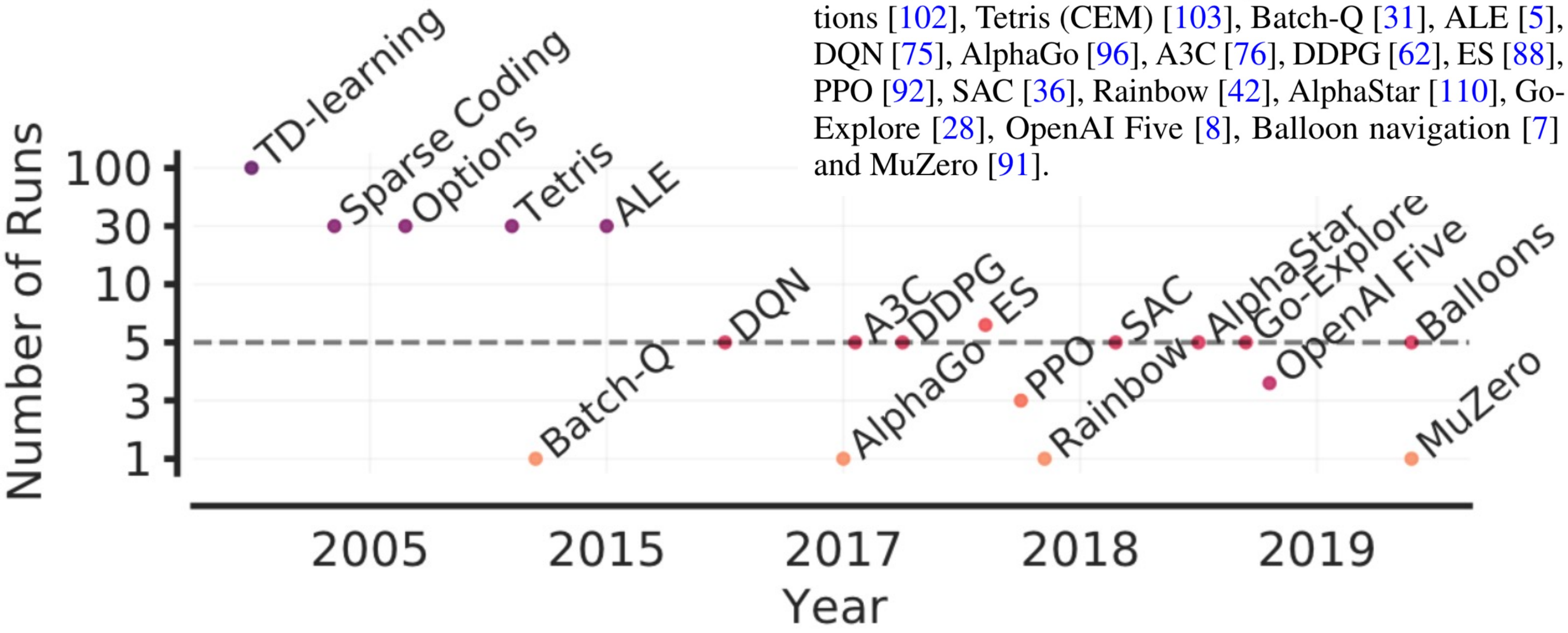


Figure 1: **Number of runs in RL over the years**. Beginning with DQN [75] on the ALE, 5 or less runs are common in the field. Here, we show representative RL papers with empirical results, in the order of their publication year: TD-learning [99], Sparse coding [100], Options [102], Tetris (CEM) [103], Batch-Q [31], ALE [5], DQN [75], AlphaGo [96], A3C [76], DDPG [62], ES [88], PPO [92], SAC [36], Rainbow [42], AlphaStar [110], Go-Explore [28], OpenAI Five [8], Balloon navigation [7] and MuZero [91].

Agarwal, et al. "Deep reinforcement learning at the edge of the statistical precipice." NeurIPS, 2021.

Because training is expensive (recall Rainbow)

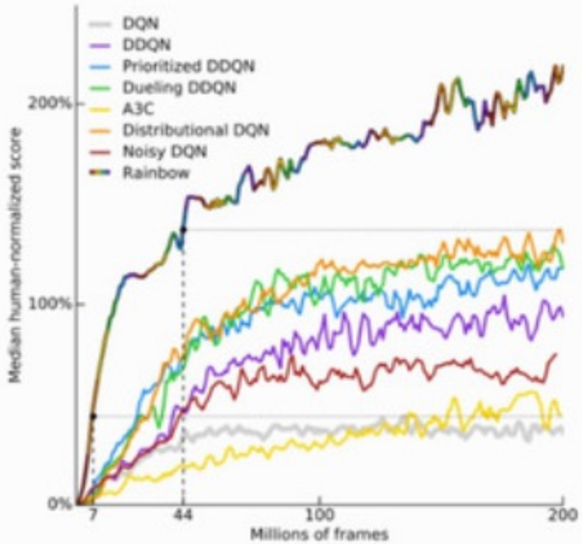


Figure 1: **Median human-normalized performance** across 57 Atari games. We compare our integrated agent (rainbow-colored) to DQN (grey) and six published baselines. Note that we match DQN's best performance after 7M frames, surpass any baseline within 44M frames, and reach substantially improved final performance. Curves are smoothed with a moving average over 5 points.

The cost of Rainbow:

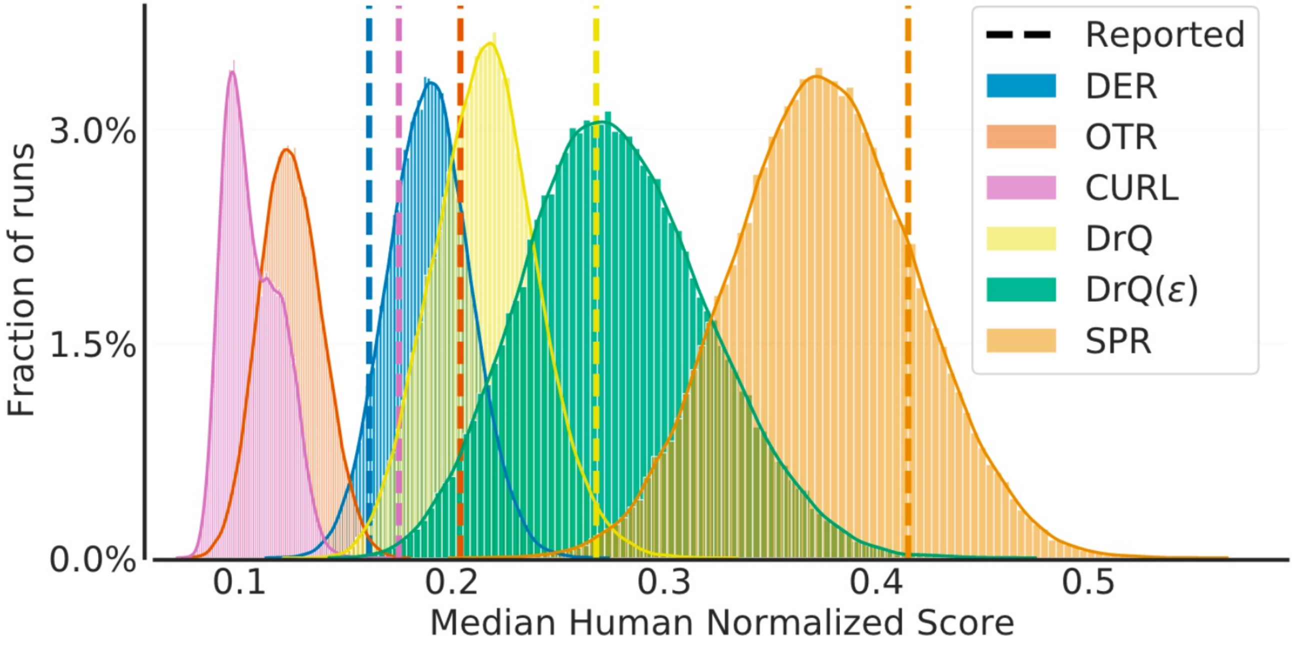
- ~5 days to train a game on a P100
- 57 games
- 5 independent seeds
- P100 costs about US$6000

Lower-bound of cost:

- 34,200 GPU hours
- 1425 days
- Total cost: EXPENSIVE

Obando-Ceron J. S., Castro P. S. Revisiting Rainbow: Promoting more insightful and inclusive deep reinforcement learning research. ICML, 2021.

However, 5 runs are not enough



5-20 runs

Repeatedly sampled from 100 runs (Bootstrap)

Issue 1: Insufficient to conclude which methods are better

Issue 2: Reported numbers severely overestimate the expected mean/median

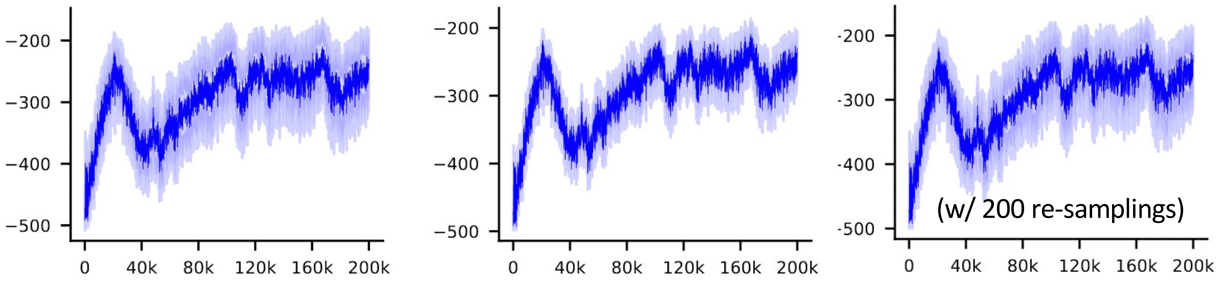
Maximization bias (recall overestimation in DQN)

Agarwal, et al. "Deep reinforcement learning at the edge of the statistical precipice." NeurIPS, 2021.

Tip 1: Compute confidence intervals

- Capture confidence in estimation of the mean performance
- E.g., Student t-distribution (Gaussian assumption), Percentile bootstrap
- Use hypothesis tests for rigorous comparison
  - Can we reject the null hypothesis that performance of A and B are the same?
- See lecture Appendix A for a concept refresher (and a discussion on whether the Gaussian assumption is OK)

Confidence intervals for DQN on Mountain Car (30 runs) [1]

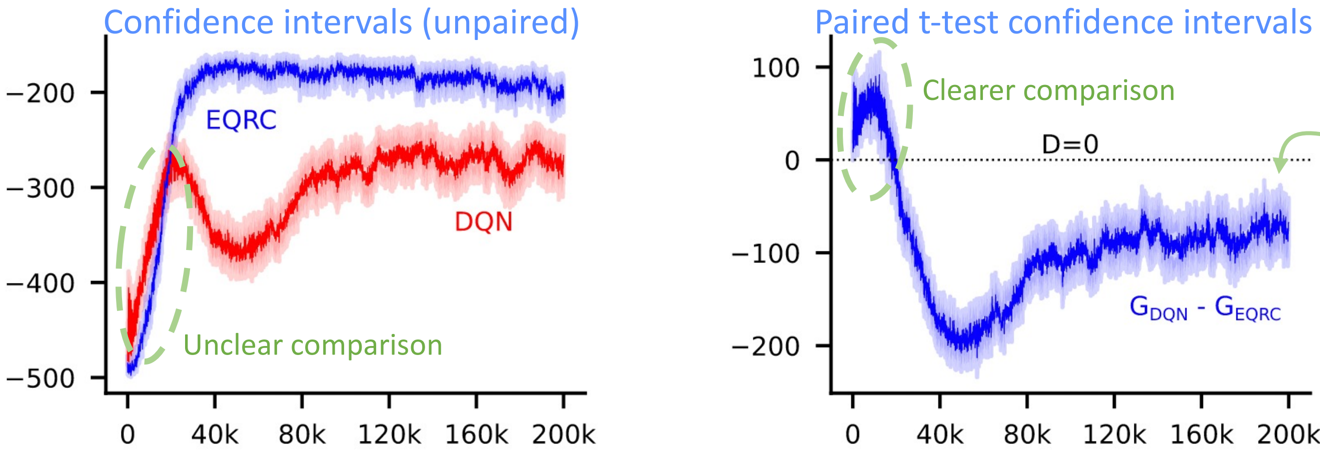


(a) $\alpha = 0.05$ with Student's t (b) $\alpha = 0.3$ with Student's t (c) $\alpha = 0.05$ with bootstrap

[1] A. Patterson, S. Neumann, M. White, and A. White, "Empirical Design in Reinforcement Learning." arXiv, 2023. doi: 10.48550/arXiv.2304.01315.

Tip 1A: Paired t-test for comparing two methods

- Non-blocking design (t-test) → Blocking design (paired t-test)
- Blocking controls for sources of variation
- A vs B → A-B > 0, where A-B are paired



(a) Individual performance of DQN and EQRC. (b) Performance difference $D = G_{\text{DQN}} - G_{\text{EQRC}}$.

Task: Mountain Car [1]

When the shaded region does not include the horizontal line at D = 0, then the reported difference is statistically significant

[1] A. Patterson, S. Neumann, M. White, and A. White, "Empirical Design in Reinforcement Learning." arXiv, 2023. doi: 10.48550/arXiv.2304.01315.

Tip 2: Isolate the modification

- Isolate the precise modification made in the proposed method
- Consider a new method $\widetilde{\boldsymbol{A}}$ which is method $\boldsymbol{A}$ with modification $\boldsymbol{X}$ $(\boldsymbol{A}+\boldsymbol{X})$
  - Example: a novel replay buffer.
- Rather than evaluating $\widetilde{\boldsymbol{A}}$ by comparing with $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, etc., estimate the effect of $\boldsymbol{X}$ by comparing $\boldsymbol{A}$ vs $\boldsymbol{A}+\boldsymbol{X}$, $\boldsymbol{B}$ vs $\boldsymbol{B}+\boldsymbol{X}$, $\boldsymbol{C}$ vs $\boldsymbol{C}+\boldsymbol{X}$.
- Furthermore, apply blocking design (paired t-test) [Tip 1A]
  - Compare $\boldsymbol{A}-(\boldsymbol{A}+\boldsymbol{X})$ vs 0, $\boldsymbol{B}-(\boldsymbol{B}+\boldsymbol{X})$ vs 0, $\boldsymbol{C}-(\boldsymbol{C}+\boldsymbol{X})$ vs 0
  - Estimates whether the relative effect was significant
  - Can pool samples together into a single paired t-test

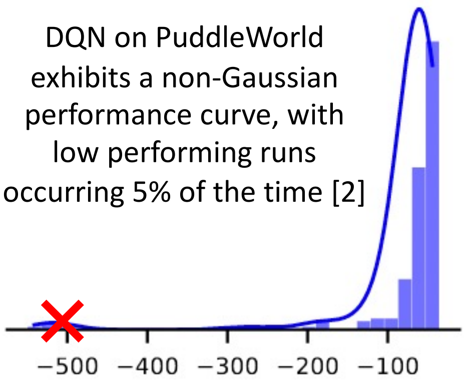
A. Patterson, S. Neumann, M. White, and A. White, "Empirical Design in Reinforcement Learning." arXiv, 2023. doi: 10.48550/arXiv.2304.01315.

Tip 3: Use robust statistics

Use of **robust statistics** can ease estimation of *typical* performance [1].

- Interquartile-mean (IQM) drops the highest and lowest 25% of samples, before computing the mean of the remaining 50% of the data.
- The median can be used directly, but that drops almost nearly all of the data.

DQN on PuddleWorld exhibits a non-Gaussian performance curve, with low performing runs occurring 5% of the time [2]

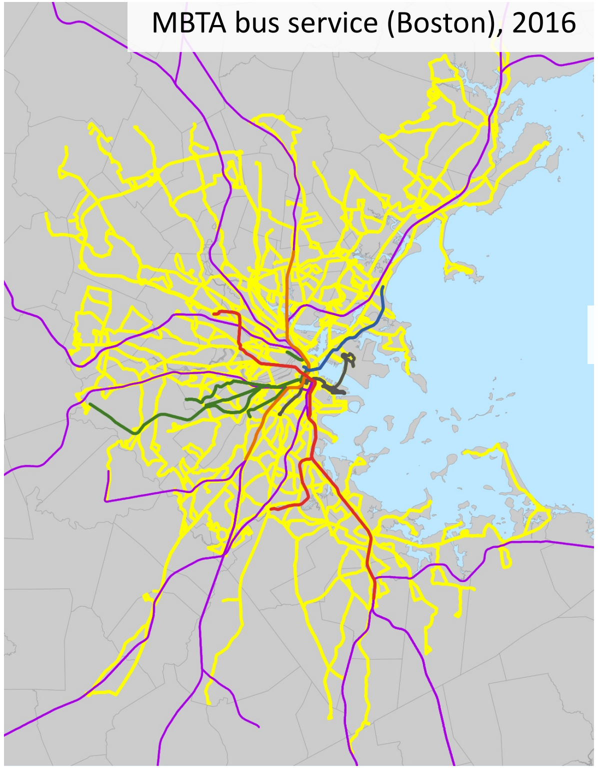


[1] Agarwal, et al. "Deep reinforcement learning at the edge of the statistical precipice." NeurIPS, 2021.
[2] A. Patterson, S. Neumann, M. White, and A. White, "Empirical Design in Reinforcement Learning." arXiv, 2023. doi: 10.48550/arXiv.2304.01315.

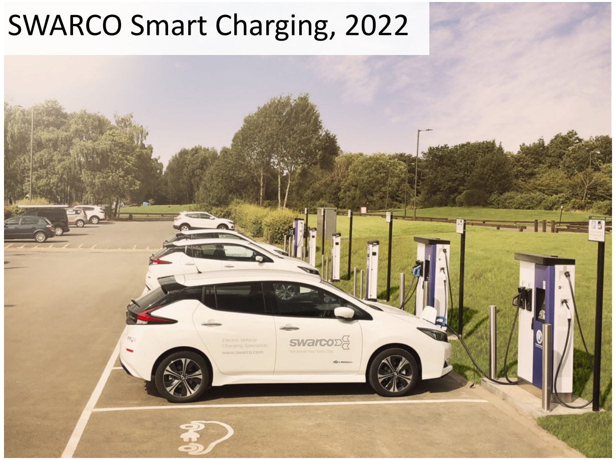
Outline

1. Challenge 1: RL is quite sensitive
2. Solution 1: Standardize RL evaluation
3. **Challenge 2: Overfitting to benchmarks**
   a. General purpose or benchmark-specific RL methods?
   b. Motivation: RL for designing future mobility systems
   c. Task underspecification (NeurIPS 2022)
4. Solution 2: Explicitly model generalization

Motivation: seamless mobility


MBTA bus service (Boston), 2016

Transit network design


SWARCO Smart Charging, 2022

Fleet electrification

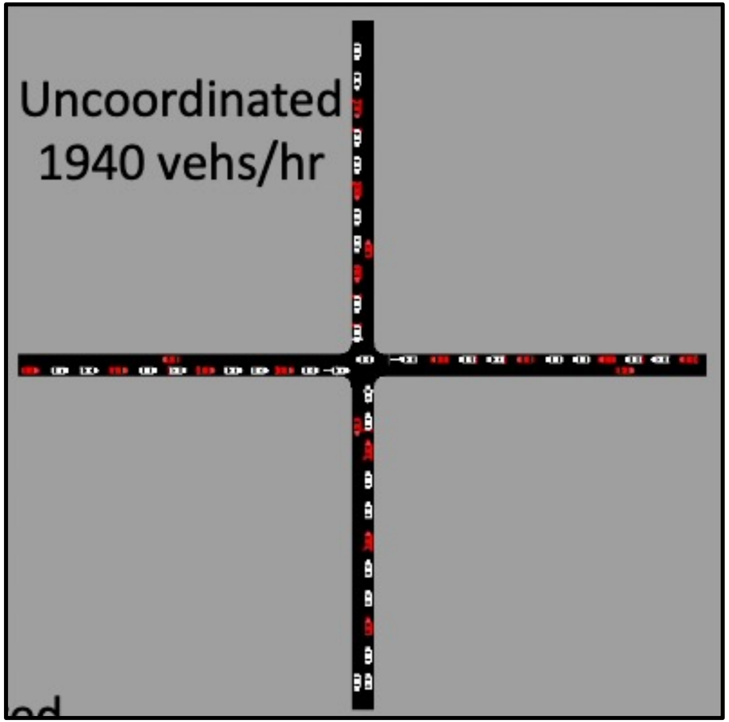

Wanis Kabbaj, 2016

Safe & efficient traffic

Challenge

Designing and operating future mobility systems requires solving **many hard optimization & control problems.**

Example: Mixed Autonomy Traffic [1-2]



- Autonomy features
  - Autonomy adoption
  - Autonomy level
- Traffic features
  - Different performance measures
  - Multi-objective optimization
  - Network topology and configuration
  - Sensing, communication, & control technology
  - Type of vehicles & road users
  - Human behavior
  - Weather conditions
  - Social & cultural norms
- …

Countless variants

Too many problems to
manually derive algorithms by hand.
Can we automatically derive algorithms?

[1] P. Varaiya, “Smart cars on smart roads: problems of control,” *IEEE Transactions on automatic control,*, 1993.
[2] P. Ioannou and Z. Xu, “Throttle and Brake Control Systems for Automatic Vehicle Following,” *I V H S Journal*, 1994.

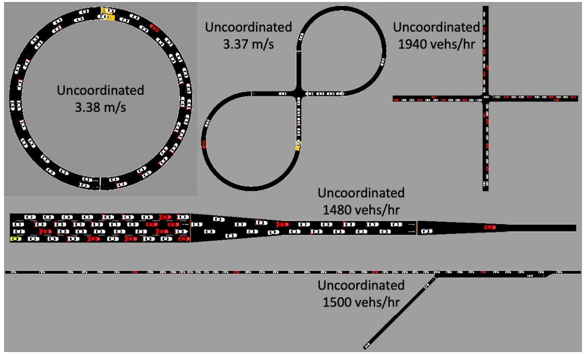
Heterogeneity in mixed autonomy traffic

Source of heterogeneity	**Examples**
Traffic phenomena	Phantom jams, capacity drop, convective instability
Basic traffic networks	Ring, multi-lane ring, figure 8, merge, bottleneck, intersection
Intersections	Topology, turn restrictions, road grade, weather, travel demand, vehicle types, age, etc.
Composite networks	Highway & urban networks, grid networks
Lane change behavior	Disabled, MOBIL, etc.
Performance measure	Congestion, safety, emissions, mixed, etc.
Traffic demand	None, low, moderate, high
Objective	Ego-centric, system-centric, mixed
Level of autonomy	Levels 0–5
Penetration of CAVs	0–100%

Roughly speaking:

$3^{10} \approx 60K$ **scenarios**

Personal journey with reinforcement learning

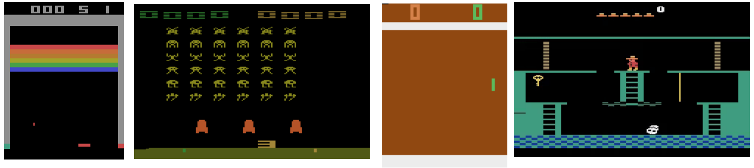


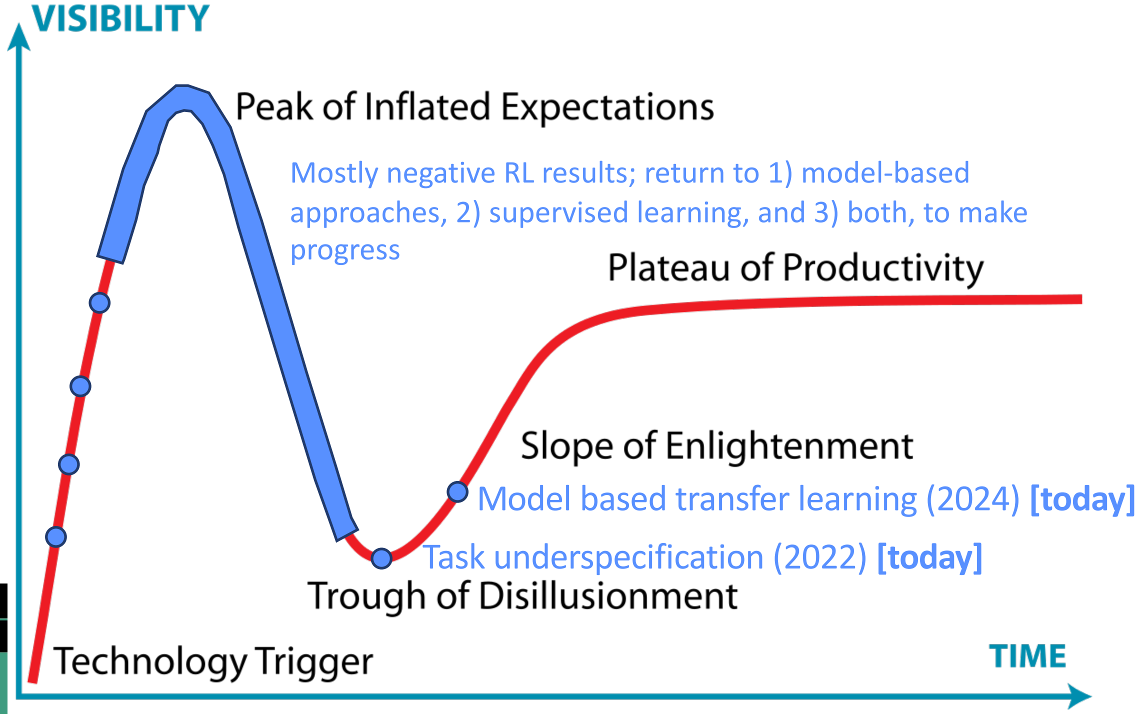
Mixed autonomy traffic (2017)

AlphaGoZero (2017)

AlphaGo (2016)

DQN (2015)

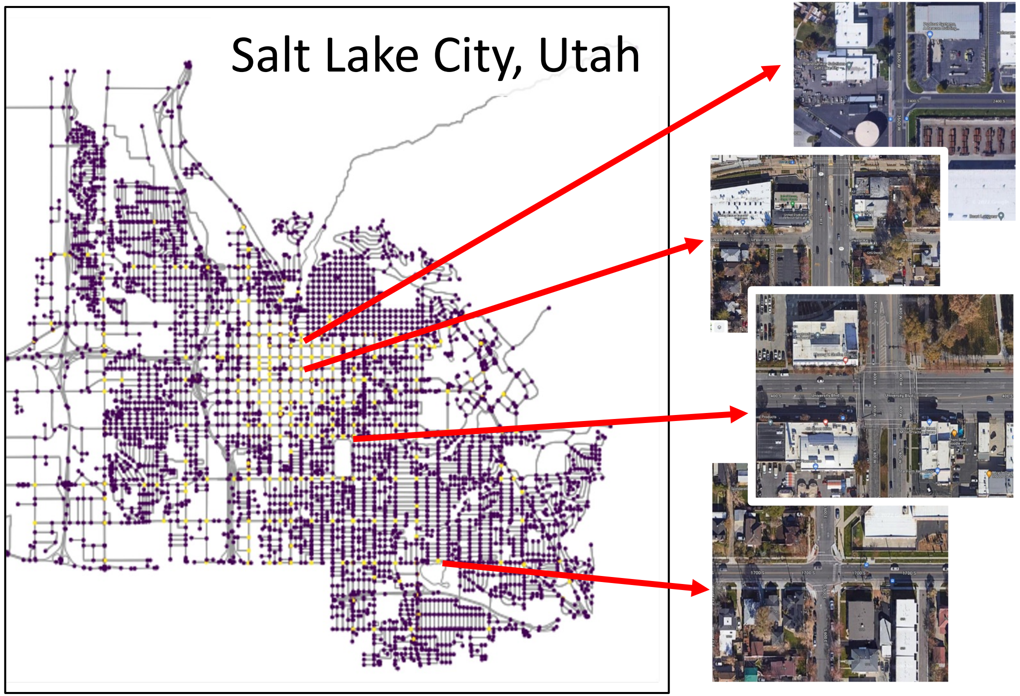




Wu, Kreidieh, Vinitsky, Bayen, “Emergent behaviors in mixed-autonomy traffic,” in *1st Annual Conference on Robot Learning (CoRL)*, PMLR, 2017.
Wu, Kreidieh, Parvate, Vinitsky, Bayen, “Flow: A modular learning framework for mixed autonomy traffic,” *IEEE Transactions on Robotics (T-RO)*, 2021.
Vinitsky, et al. **Wu**, Bayen. “Benchmarks for reinforcement learning in mixed-autonomy traffic,” in *2nd Annual Conference on Robot Learning (CoRL)*, PMLR, 2018.
Yan, Kreidieh, Vinitsky, Bayen, **Wu**, “Unified automatic control of vehicular systems with reinforcement learning,” *IEEE Transactions on Automation Science and Engineering (T-ASE)*, 2022.
Jayawardana, Tang, Li, Suo, **Wu**. “The Impact of Task Underspecification in Evaluating Deep Reinforcement Learning.” Advances in Neural Information Processing Systems (NeurIPS), 2022.


NOT SURE IF TRAFFIC IS HARD
OR DEEP RL DOESN'T WORK
imgflip.com

Task variations: signalized intersections

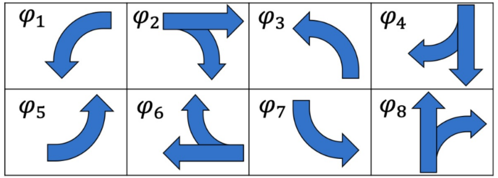


345 intersections analyzed
164 unique configurations

Feature	Units	Mean	Standard Dev
Lane Count	-	3.8	1.37
Speed	mph	32.6	5.40
Length of Lanes	meters	260.8	193
Vehicle inflow	vehicles/hour	73.5	774
Left Turns Count	-	0.229	0.496
Right Turns Count	-	0.100	0.298

Traffic signal control

Decision (action): traffic phases



Example: Typical phases, or traffic movements, in a 4-way intersection.

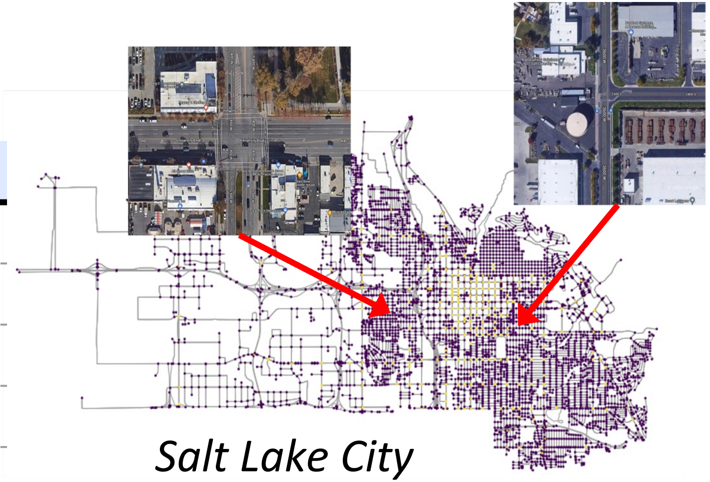
Qu*, Valiveru*, Tang, Jayawardana, Freydt, **Wu**. "What is a Typical Signalized Intersection in a City? A Pipeline for Intersection Data Imputation from OpenStreetMap." TRB, 2023.
Jayawardana, Tang, Li, Suo, **Wu**. "The Impact of Task Underspecification in Evaluating Deep Reinforcement Learning." Advances in Neural Information Processing Systems (NeurIPS), 2022.

Sensitivity of RL to task variation

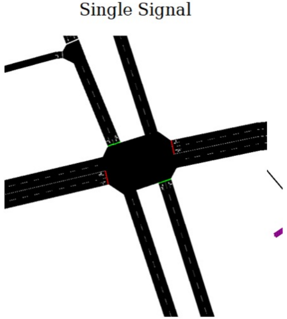
Method evaluation for traffic signal control

Rank	Benchmark [1]	Salt Lake City
1 Best	Max pressure	Fixed Time
2	IDQN	Max pressure
3	MPLight	IDQN
4	IPPO	MPLight
5	Fixed Time	MPLight*
6 Worst	MPLight*	IPPO

Methods: — Deep RL — — Classical strategy



Salt Lake City



Benchmark (RESCO [1])

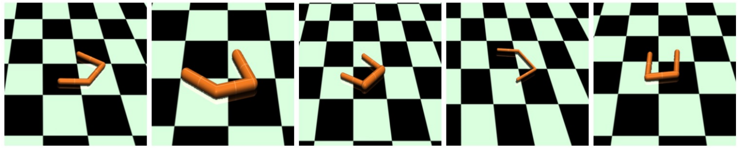
- Deep RL is **not robust** to problem variations
- Simple baseline outperforms deep RL methods!

[1] Ault, Sharon. "Reinforcement learning benchmarks for traffic signal control." Advances in Neural Information Processing Systems (NeurIPS), 2021.
Jayawardana, Tang, Li, Suo, **Wu**. "The Impact of Task Underspecification in Evaluating Deep Reinforcement Learning." Advances in Neural Information Processing Systems (NeurIPS), 2022.

Sensitivity of RL to task variation

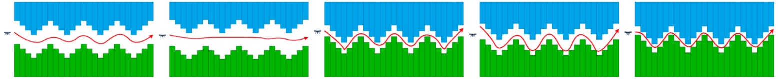
Overfitting to benchmark?

Similar findings in popular control benchmarks



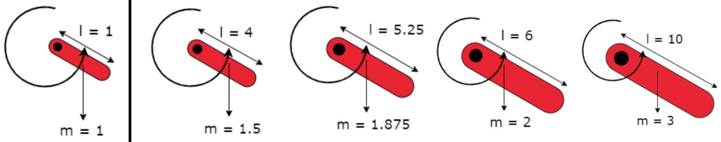
(a) MDP 1 (b) MDP 2 (c) MDP 3 (d) MDP 4 (e) MDP 5

Figure 9: Visual illustration of family of point MDPs used in the Swimmer task



(a) (u=1,l=5) (b) (u=0,l=6) (c) (u=0.5,l=3) (d) (u=0.625,l=4) (e) (u=1,l=2)

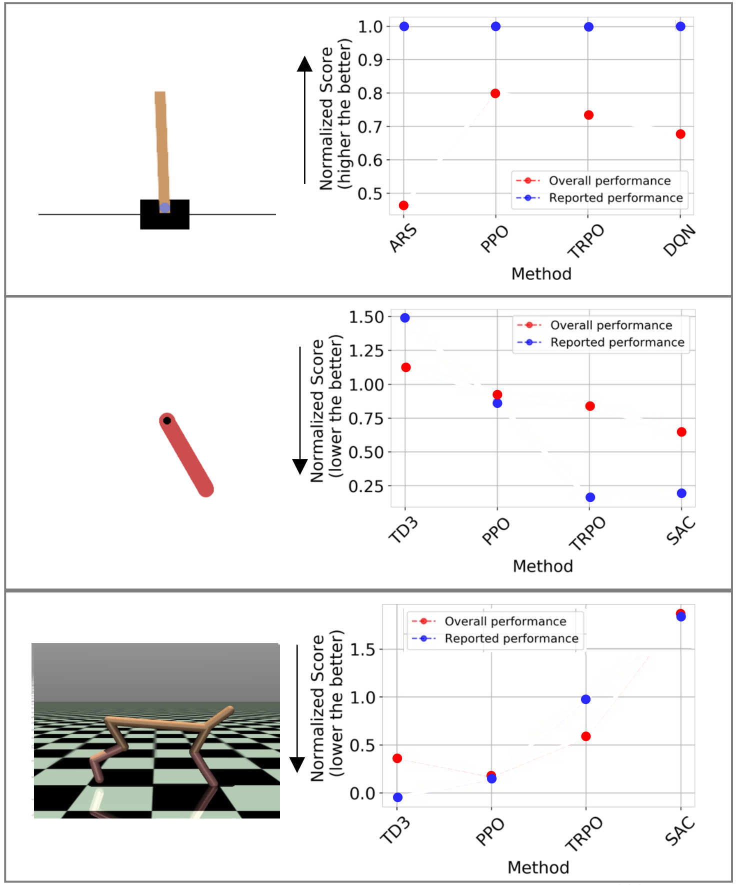
Figure 10: Visual illustration of family of point MDPs used in the Quad task



(a) (m=1,l=1) (b) (m=1.5,l=4) (c) (m=1.875,l=5.25) (d) (m=2,l=6) (e) (m=3,l=10)

Reference benchmark

Figure 11: Visual illustration of family of point MDPs used in the Pendulum task



*Reported performance is reproduced from common benchmark task specification.

Jayawardana, Tang, Li, Suo, Wu. **The Impact of Task Underspecification in Evaluating Deep Reinforcement Learning.** Advances in Neural Information Processing Systems (NeurIPS), 2022.

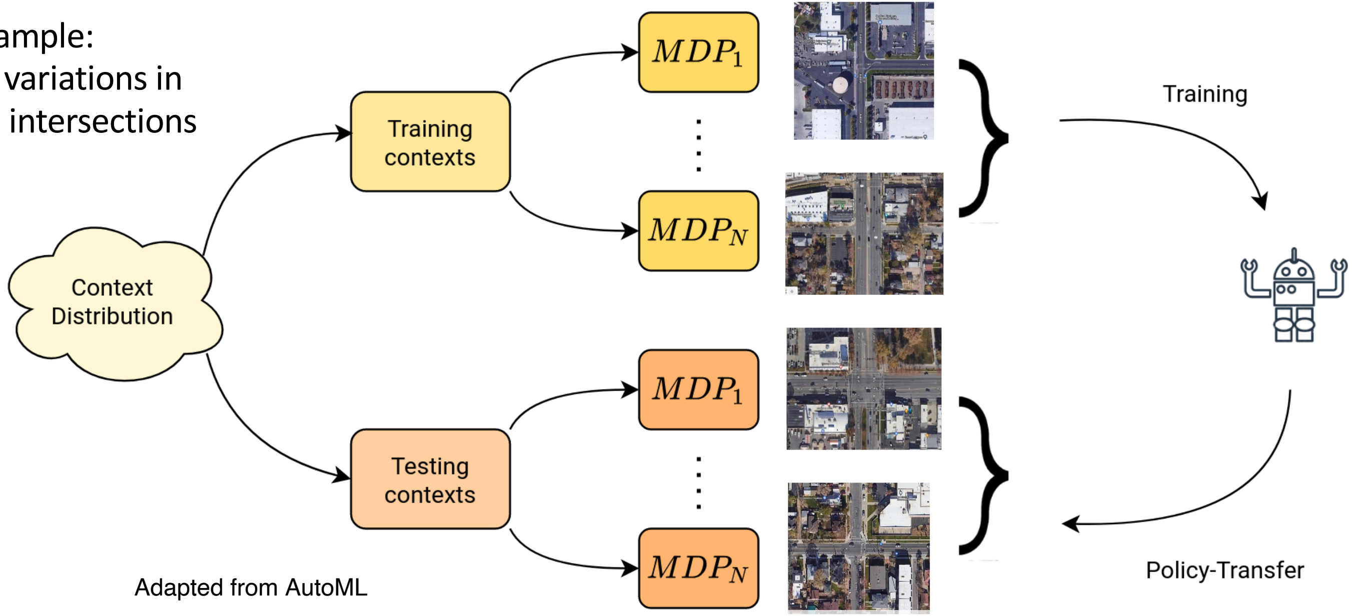
Outline

1. Challenge 1: RL is quite sensitive
2. Solution 1: Standardize RL evaluation
3. Challenge 2: Overfitting to benchmarks
4. **Solution 2: Explicitly model generalization**
    a. Contextual RL
    b. Model-based transfer learning (NeurIPS 2024)

A broader framework

- **Contextual Markov Decision Process (CMDP)**
  - A generalization of MDP that explicitly incorporates environment characteristics
  - Useful for describing families of MDPs
- **Contextual Reinforcement Learning (CRL)** studies CMDPs

Example:
Problem variations in
signalized intersections

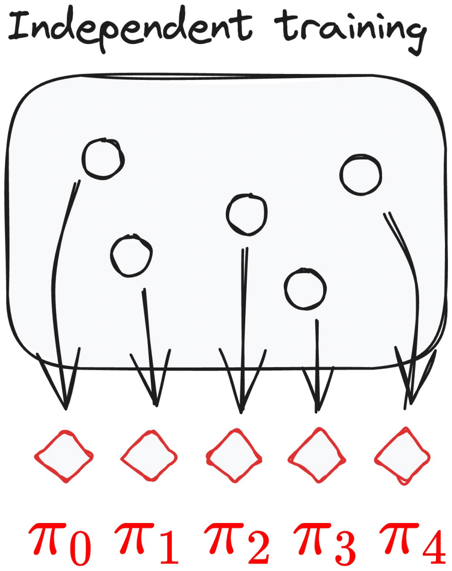


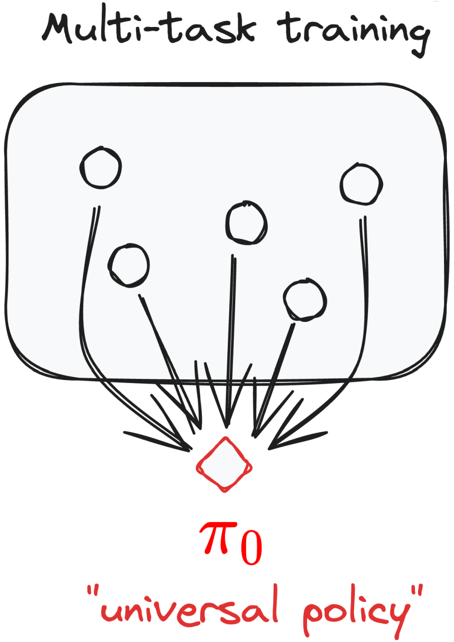
Adapted from AutoML

[1] A. Modi, N. Jiang, S. Singh, and A. Tewari, "Markov Decision Processes with Continuous Side Information," in Proceedings of Algorithmic Learning Theory, PMLR, Apr. 2018.
[2] A. Hallak, D. Di Castro, and S. Mannor, "Contextual Markov Decision Processes," Feb. 08, 2015, arXiv: arXiv:1502.02259. Available: http://arxiv.org/abs/1502.02259

Want: Fast, reliable training for CMDPs

- Typical approaches





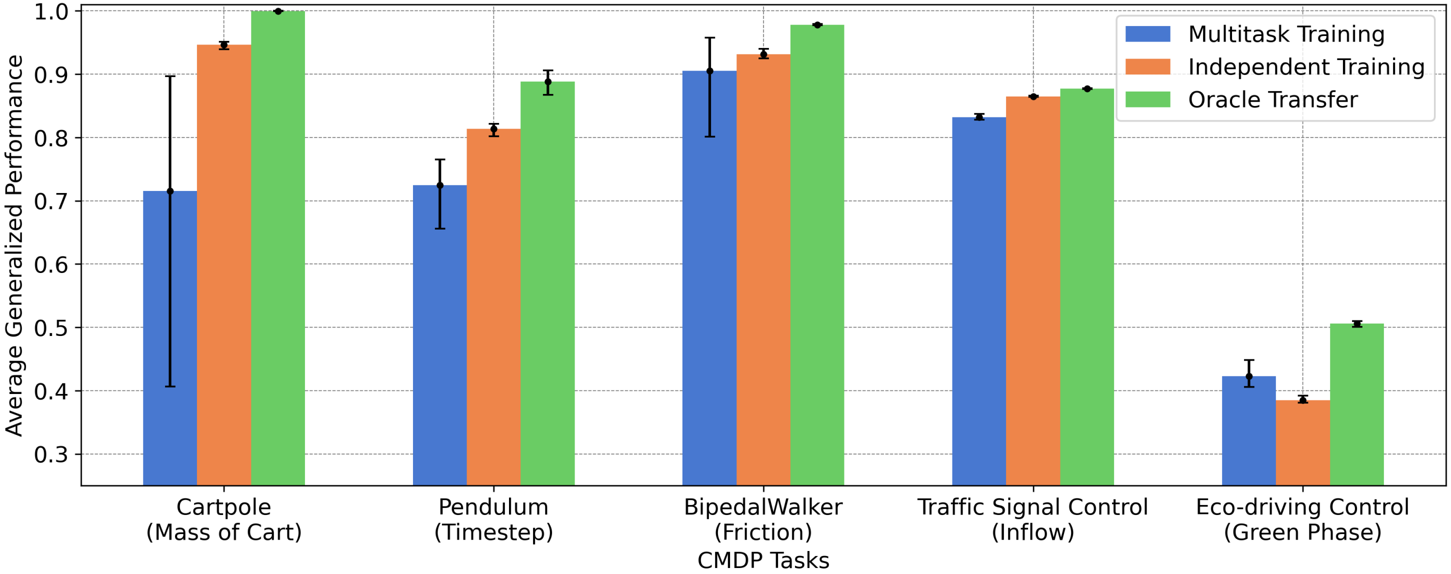
❌ Expensive to train all tasks
✅ Specializes policies to different tasks

✅ More sample efficient to train
❌ Requires high model capacity [1]
❌ Cross-task training instability

[1] A. A. Taıga, R. Agarwal, J. Farebrother, A. Courville, and M. G. Bellemare, “Investigating Multi-task Pretraining and Generalization in Reinforcement Learning,” in The Eleventh International Conference on Learning Representations, 2023.

Unreasonable effectiveness of zero-shot transfer

- **Neither is sufficient.**
- Observation: **Zero-shot transfer** is cheap & works remarkably well



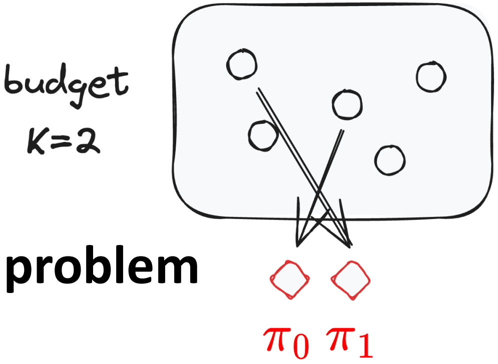
Note: all methods given same training budget

- Algorithmic question
  - How to select which tasks to train? → **Source task selection problem**

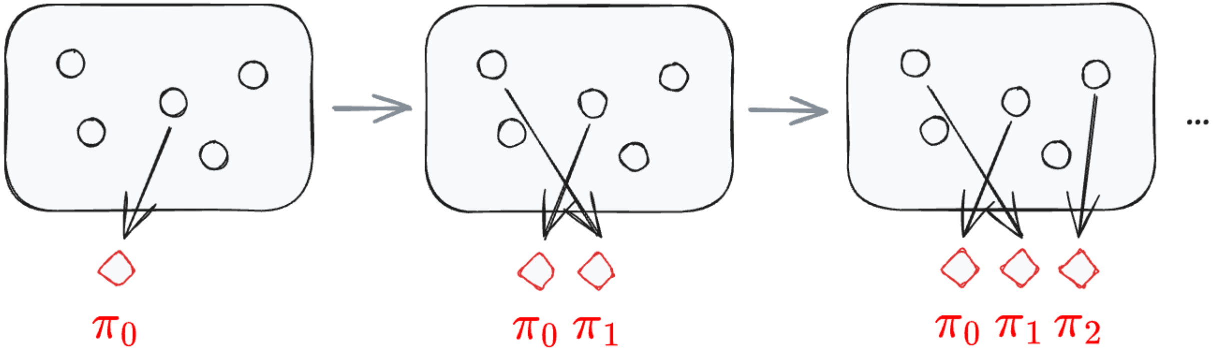
J.-H. Cho, V. Jayawardana, S. Li, and **C. Wu**, "Model-Based Transfer Learning for Contextual Reinforcement Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, Dec. 2024.

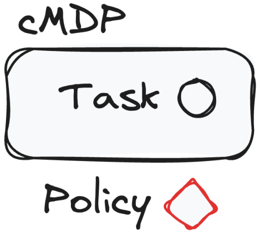
Problem formulation

- Aim: Choose source tasks that maximize target tasks performance
- **Option 1: Source task selection (STS) problem**
  - Choose training tasks all at once



- **Option 2: <u>Sequential</u> source task selection (SSTS) problem**
  - Choose training tasks sequentially



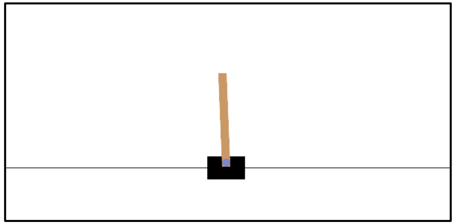


[1] J.-H. Cho, S. Li, J. Kim, and **C. Wu**, "Temporal Transfer Learning for Traffic Optimization with Coarse-grained Advisory Autonomy." Under review.
[2] J.-H. Cho, V. Jayawardana, S. Li, and **C. Wu**, "Model-Based Transfer Learning for Contextual Reinforcement Learning." Under review.

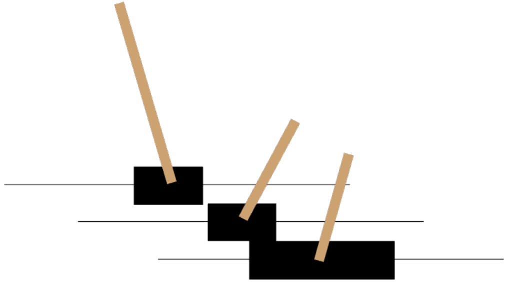
Proposed approach: Model-based transfer learning

- **Strategically** select training tasks
- **How?** **Explicit modeling of generalization performance ("model-based")**

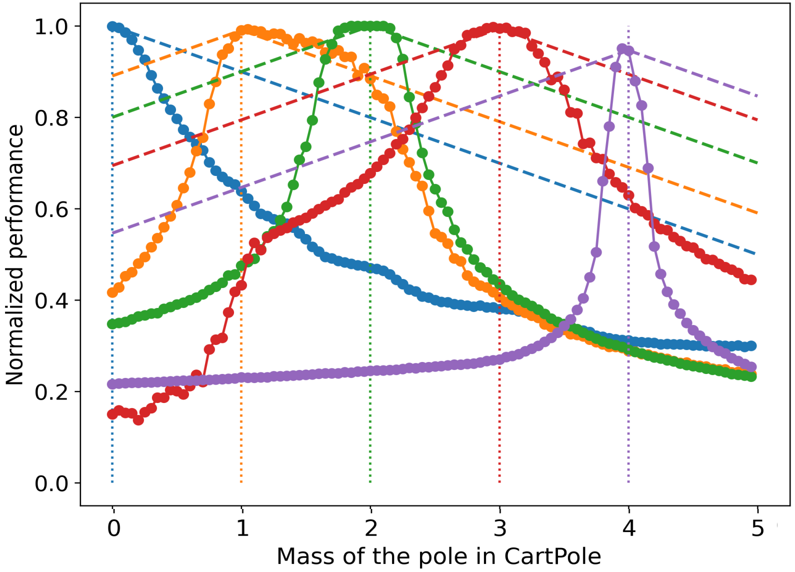
Example: Cartpole benchmark task



Cartpole CMDP



Zero-shot Generalization



Modeling assumption:

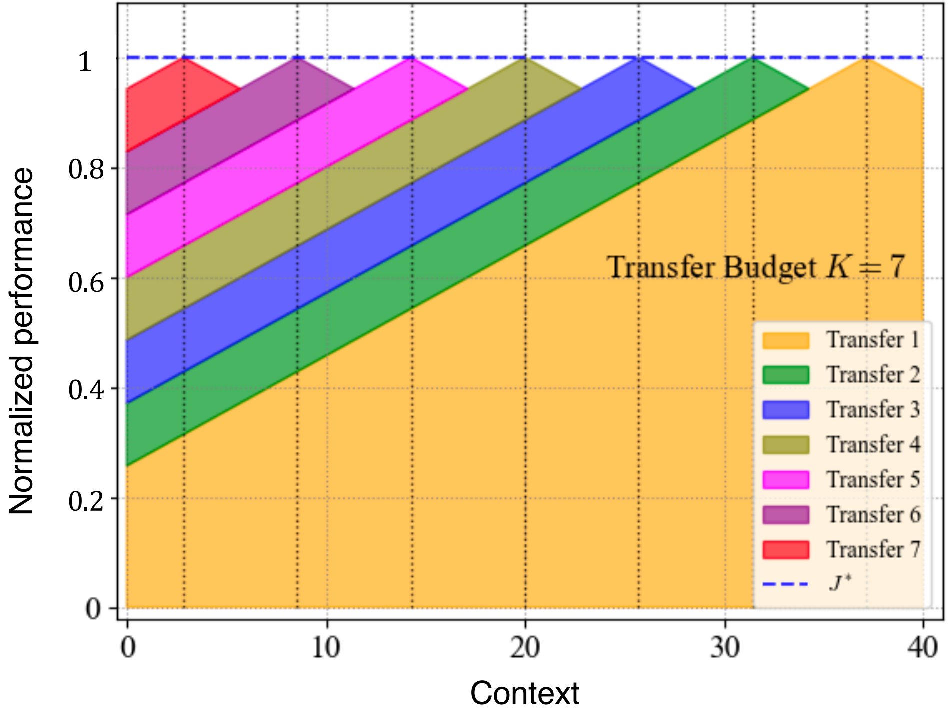
Linear generalization gap

J.-H. Cho, V. Jayawardana, S. Li, and **C. Wu**, "Model-Based Transfer Learning for Contextual Reinforcement Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, Dec. 2024.

Warm-up 1

- Assume: Constant performance across contexts
- Assume: Fixed training budget

Theorem 1:
Equidistant selection is optimal

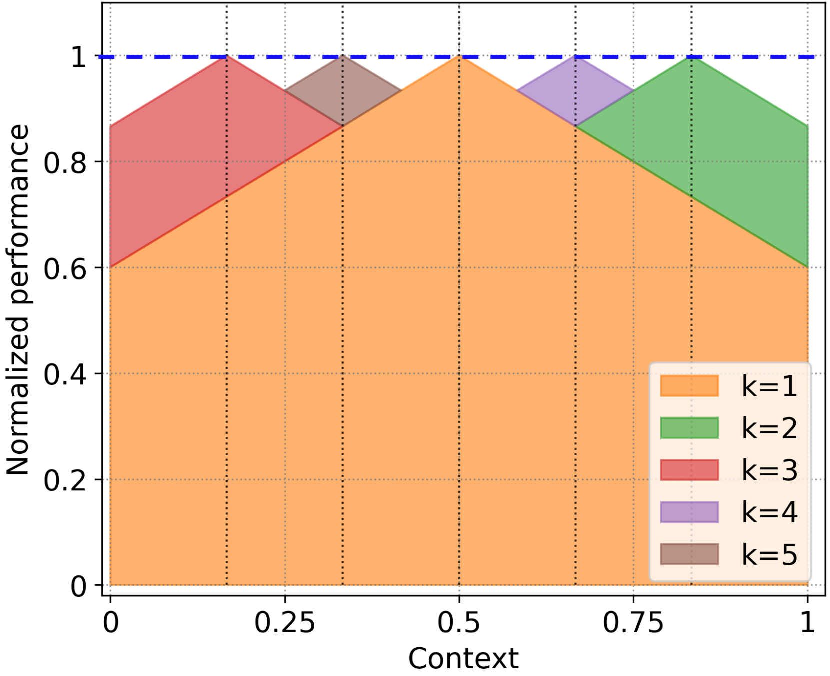


J.-H. Cho, S. Li, J. Kim, and **C. Wu**, "Temporal Transfer Learning for Traffic Optimization with Coarse-grained Advisory Autonomy," arXiv, 27 Nov. 2023, doi:10.48550/arXiv.2312.09436.

Warmup 2

- Assume: Constant performance across contexts
- ~~Assume: Fixed training budget~~ → ϵ-optimal
- Anytime algorithm

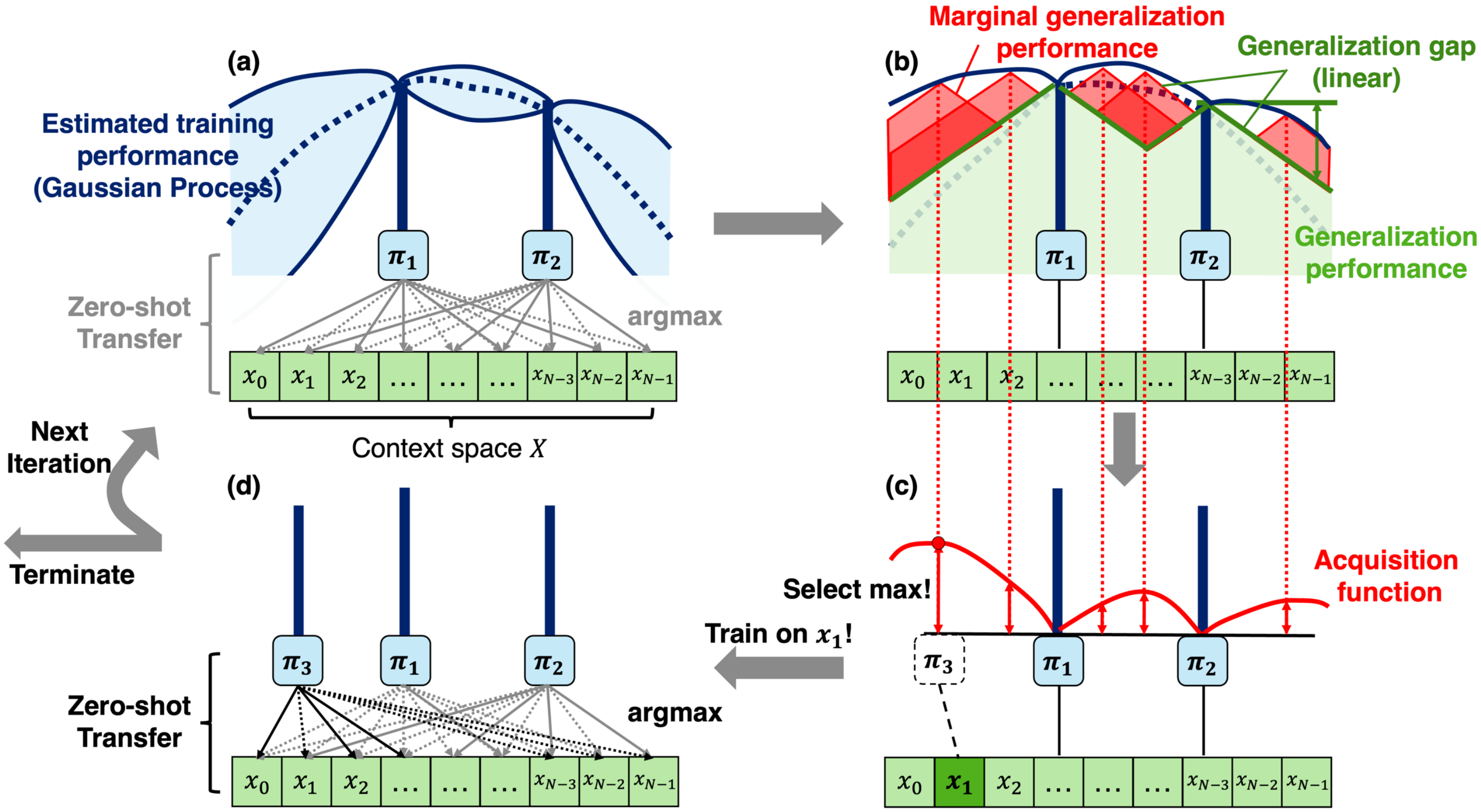
Theorem 2:
Greedy selection is bounded sub-optimal



J.-H. Cho, S. Li, J. Kim, and **C. Wu**, "Temporal Transfer Learning for Traffic Optimization with Coarse-grained Advisory Autonomy," arXiv, 27 Nov. 2023, doi:10.48550/arXiv.2312.09436.

Model-based transfer learning

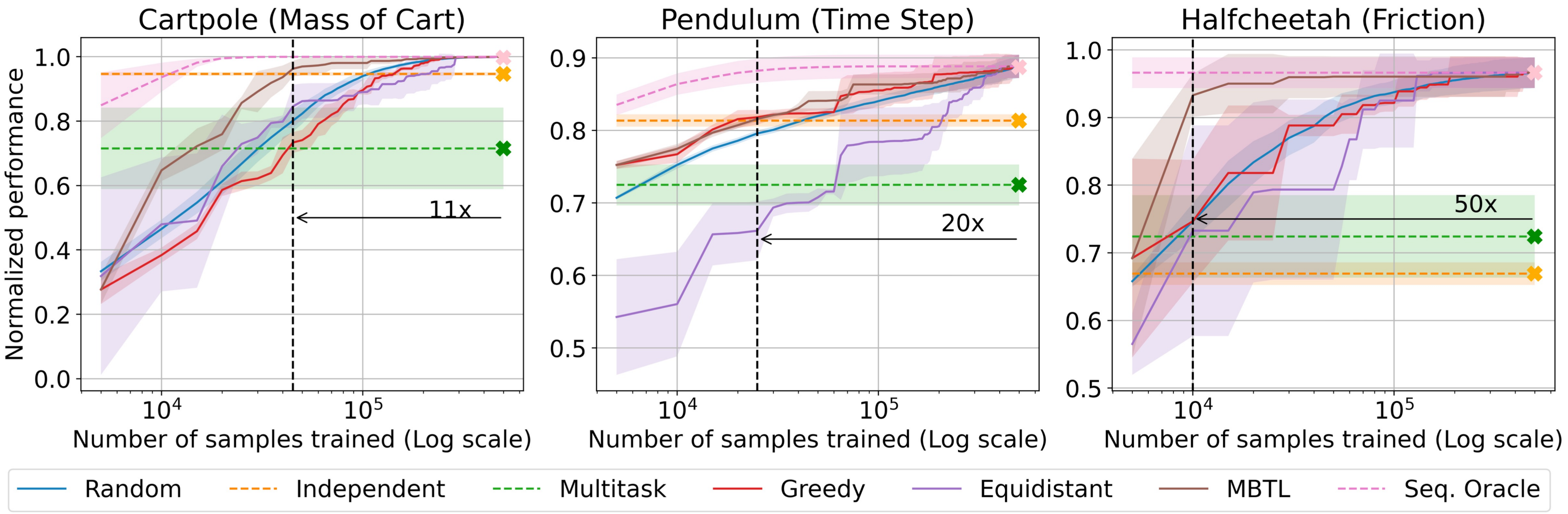
~~Assume: Constant performance across contexts~~
~~Assume: Fixed training budget~~

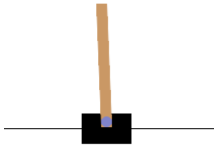


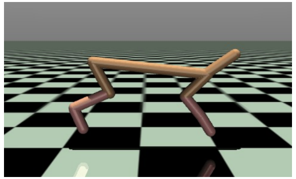
J.-H. Cho, V. Jayawardana, S. Li, and **C. Wu**, "Model-Based Transfer Learning for Contextual Reinforcement Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, Dec. 2024.

MBTL: Control experiments

MBTL achieves **up to 50x improved sample efficiency** over independent and multi-task training.



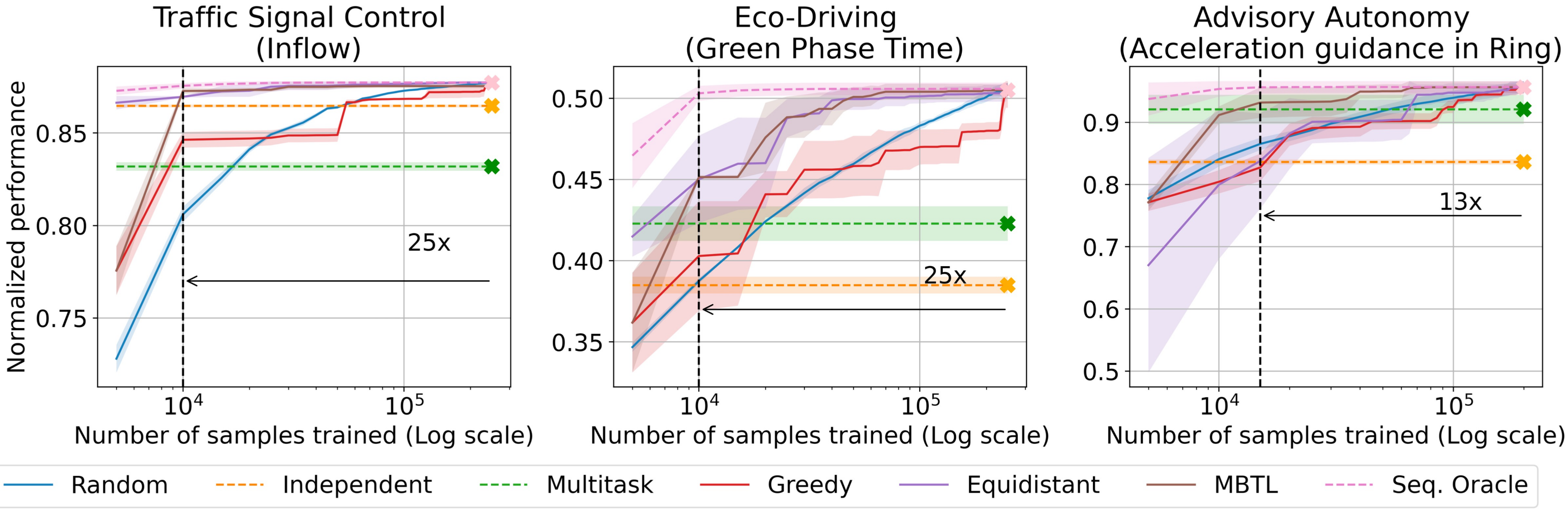


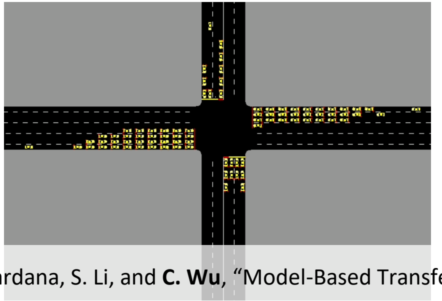


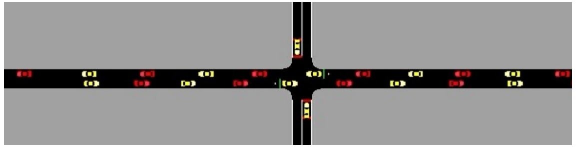
J.-H. Cho, V. Jayawardana, S. Li, and **C. Wu**, "Model-Based Transfer Learning for Contextual Reinforcement Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, Dec. 2024.

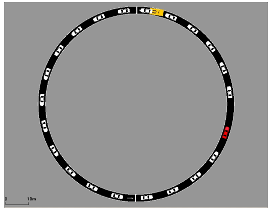
MBTL: Traffic experiments

MBTL achieves **up to 25x improved sample efficiency** over independent and multi-task training.









J.-H. Cho, V. Jayawardana, S. Li, and **C. Wu**, "Model-Based Transfer Learning for Contextual Reinforcement Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, Dec. 2024.

MBTL: Sensitivity analysis

- Generalization gap (slope): Did not tune
- Bayesian optimization acquisition function: Not that sensitive

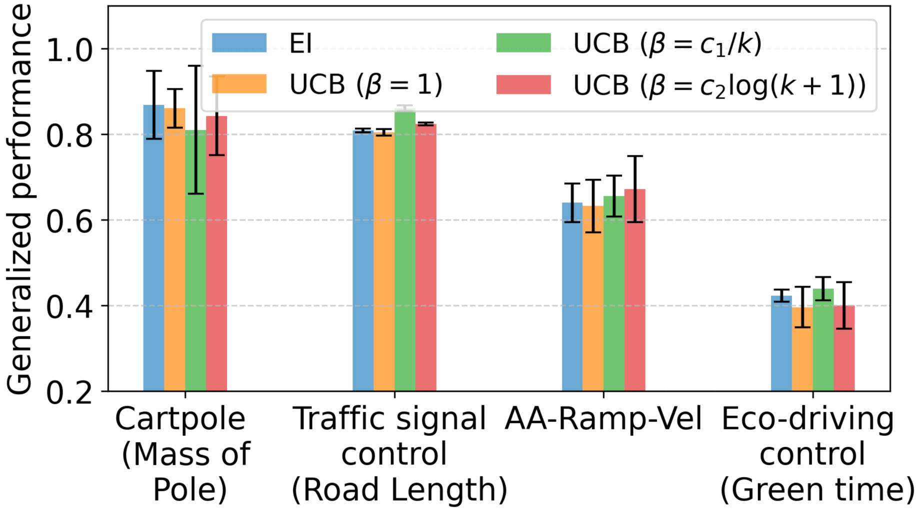
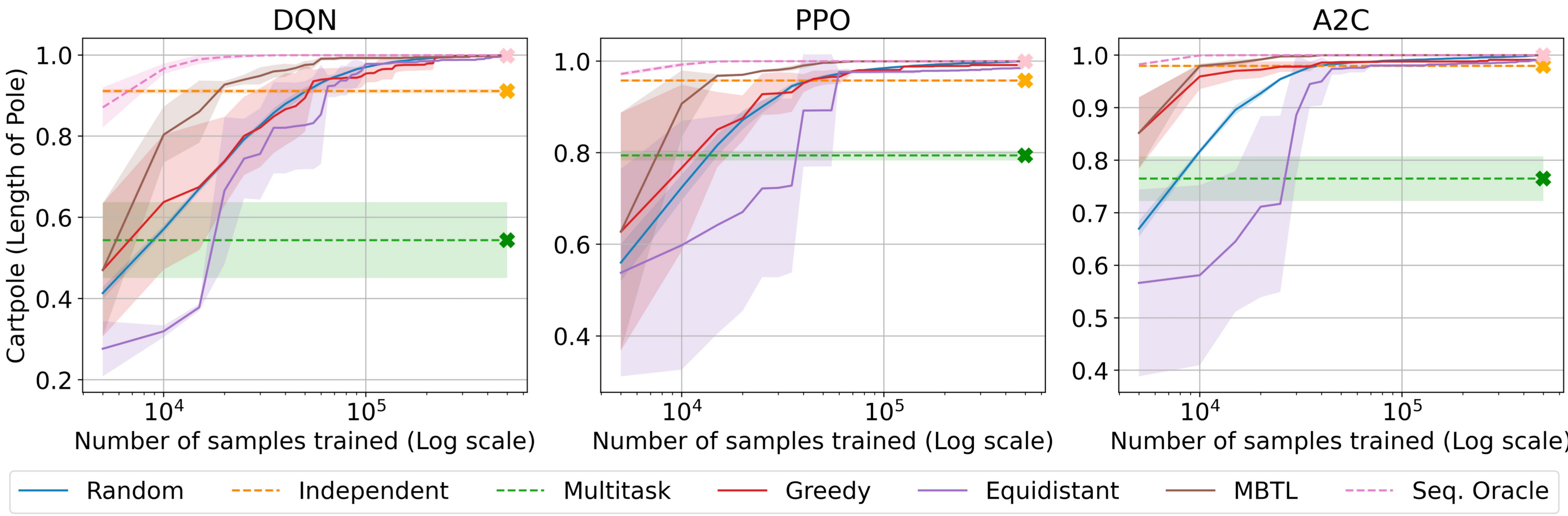


Figure 8: Sensitivity analysis on acquisition functions.

J.-H. Cho, V. Jayawardana, S. Li, and **C. Wu**, "Model-Based Transfer Learning for Contextual Reinforcement Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, Dec. 2024.

MBTL: Sensitivity analysis

- Underlying RL method: Remains effective vs baselines



J.-H. Cho, V. Jayawardana, S. Li, and **C. Wu**, "Model-Based Transfer Learning for Contextual Reinforcement Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, Dec. 2024.

Summary

- Modern RL methods are powerful but highly highly sensitive.
- Factors include: random seeds, hyperparameters, implementation details, MDP specifications, task variation
- To effectively evaluate methods, control as many factors as possible, use common benchmarks & codebases, and apply standard statistical techniques. At the very least, run multiple trials (≈10-20).
- However, overreliance on benchmarks can lead to methods that overfit the benchmark, including issues of task underspecification (a.k.a. limited external validity).
- Explicit modeling of generalization performance can enable reliable RL methods (with greater external validity).
- The design of RL methods remains an art as well as a science. See Lecture Appendix B for RL method implementation best practices.

Further reading

1. Alex Irpan. Deep Reinforcement Learning Doesn't Work Yet. Sorta Insightful Blog, 2018. https://www.alexirpan.com/2018/02/14/rl-hard.html

2. Amid Fish. Lessons Learned Reproducing a Deep Reinforcement Learning Paper. Blog, 2018. http://amid.fish/reproducing-deep-rl

3. A. Patterson, S. Neumann, M. White, and A. White, “Empirical Design in Reinforcement Learning.” arXiv, Apr. 03, 2023. doi: 10.48550/arXiv.2304.01315.

4. For general principles on experimental design: D. C. Montgomery, *Design and analysis of experiments*, Tenth edition. Wiley, 2020.
   - Chapter 1: Introduction
   - Chapter 2: Simple Comparative Experiments

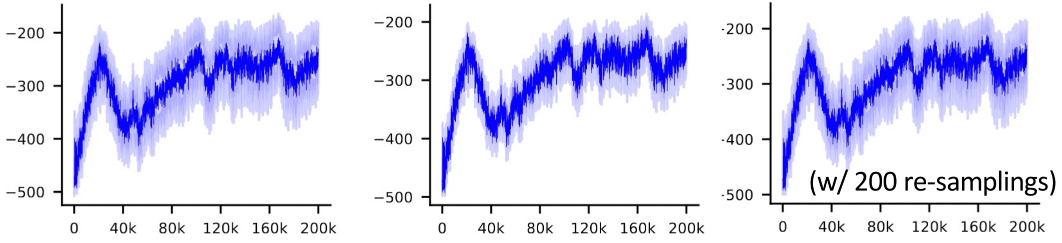
Appendix A

Confidence intervals, p-values, testing under non-standard distributions

Confidence intervals

- Capture confidence in estimation of the mean performance
- A common choice is the Student t-distribution (Gaussian assumption)
  - Consider n samples: $M_1, M_2, \dots, M_n$
  - The confidence interval is of the form $\left[\bar{M} - t_{\alpha,n}\frac{\hat{\sigma}}{\sqrt{n}}, \bar{M} + t_{\alpha,n}\frac{\hat{\sigma}}{\sqrt{n}}\right]$
    - where $\bar{M} \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} M_i$, $\bar{\sigma} \stackrel{\text{def}}{=} \frac{1}{n-1}\sum_{i=1}^{n}(M_i - \bar{M})^2$
    - The t-test statistic $t_{\alpha,n}$ depends on the significance level (typically $\alpha = 0.05$ or 0.01) and the number of samples [see lookup table].

Confidence intervals for DQN on Mountain Car (30 runs)



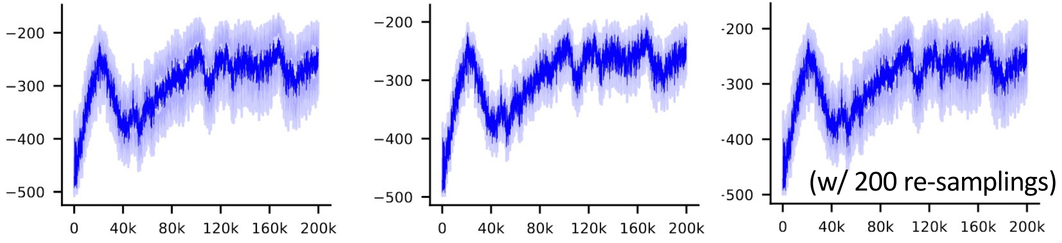
(a) $\alpha = 0.05$ with Student's t (b) $\alpha = 0.3$ with Student's t (c) $\alpha = 0.05$ with bootstrap

A. Patterson, S. Neumann, M. White, and A. White, "Empirical Design in Reinforcement Learning." arXiv, 2023. doi: 10.48550/arXiv.2304.01315.

Confidence intervals

- Percentile bootstrap is more expensive, but requires few assumptions
  - Method: Obtain n samples, then re-sample n values *with replacement*. Repeat for m re-samplings, usually large ($m \approx 10{,}000$).
  - A 95% confidence interval is generated using the $[0.025, 0.975]^{th}$ percentiles of the m re-sampled means.

Confidence intervals for DQN on Mountain Car (30 runs)



(a) $\alpha = 0.05$ with Student's t (b) $\alpha = 0.3$ with Student's t (c) $\alpha = 0.05$ with bootstrap

A. Patterson, S. Neumann, M. White, and A. White, "Empirical Design in Reinforcement Learning." arXiv, 2023. doi: [10.48550/arXiv.2304.01315](https://doi.org/10.48550/arXiv.2304.01315).

Reminder on p-values

- p-value (α): the risk of wrongly rejecting the null hypothesis
- p-values are NOT the error rate of the statistical test (β)

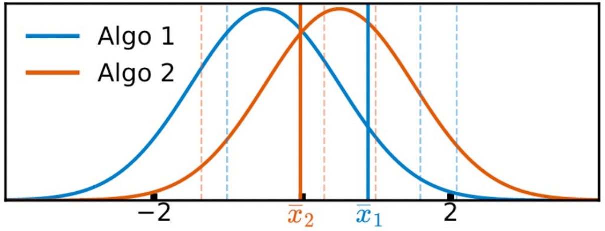


Figure 1: **Two normal distributions representing the performances of two algorithms.** Dashed lines: performance measures (realizations). Plain lines: empirical means of the two samples ($N = 3$).

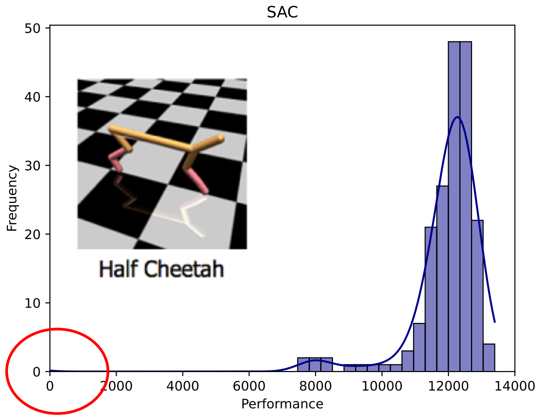
Table 1: Hypothesis testing

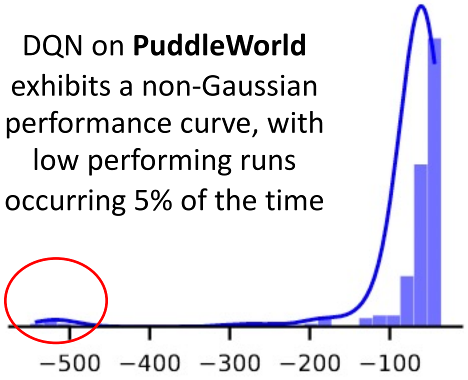
	True $\mathcal{H}_0$	True $\mathcal{H}_a$
Pred. $\mathcal{H}_0$	True neg. $1-\alpha^*$	False neg. β^*
Pred. $\mathcal{H}_a$	False pos. α^*	True pos. $1-\beta^*$

Testing under non-standard distributions

- It’s typical for RL methods to exhibit **non-standard distributions**, i.e. non-Gaussian.
- There may even be **long-tailed** performance distributions
  - Ex (PuddleWorld). 30 runs was found to be sufficient to estimate the mean.
- How bad is the Gaussian assumption then?

192 runs of SAC





DQN on **PuddleWorld** exhibits a non-Gaussian performance curve, with low performing runs occurring 5% of the time

A. Patterson, S. Neumann, M. White, and A. White, “Empirical Design in Reinforcement Learning.” arXiv, 2023. doi: [10.48550/arXiv.2304.01315](https://doi.org/10.48550/arXiv.2304.01315).

Testing under standard distributions

- How sensitive are statistical tests to distributional assumptions? To RL methods?

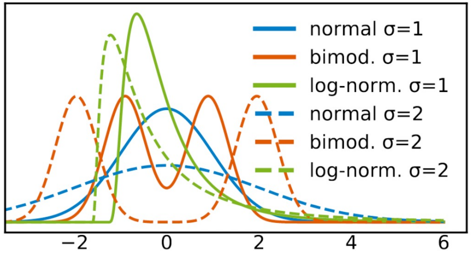
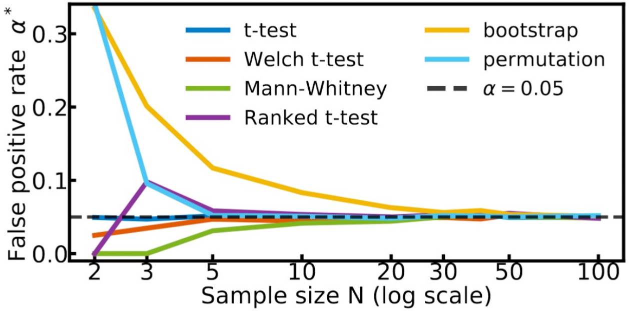


Figure 2: **Candidate distributions to represent algorithm performances**.

- Consider the standard normal
  - Number of samples to achieve false positive rate of 0.05



B. L. Welch, "The generalization of student's' problem when several different population variances are involved," Biometrika, vol. 34, no. 1/2, pp. 28–35, 1947.
C. Colas, O. Sigaud, and P.-Y. Oudeyer, "A Hitchhiker's Guide to Statistical Comparisons of Reinforcement Learning Algorithms." arXiv, 2022. doi: 10.48550/arXiv.1904.06979.

Testing under non-standard distributions

- **Gaussian assumption is fine!**
- Welsh t-test (and standard t-test) is most robust to candidate distributions
  - T-test: Assumes equal variances.
  - Welch's t-test: Different variances.

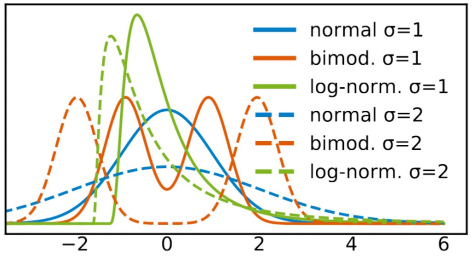


Figure 2: **Candidate distributions to represent algorithm performances**.

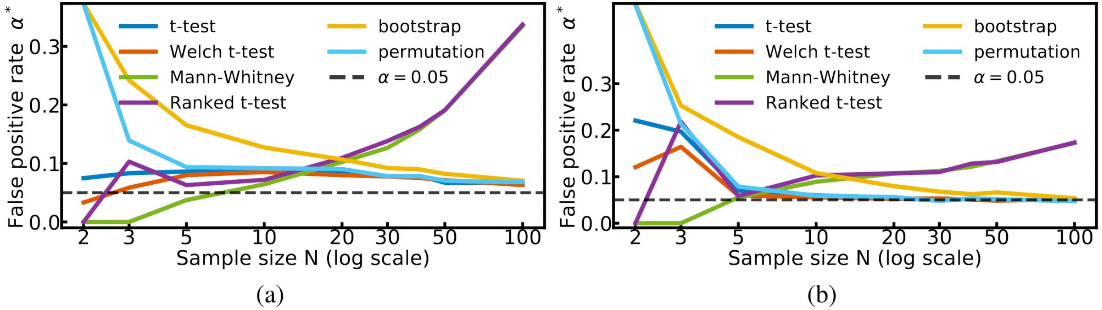


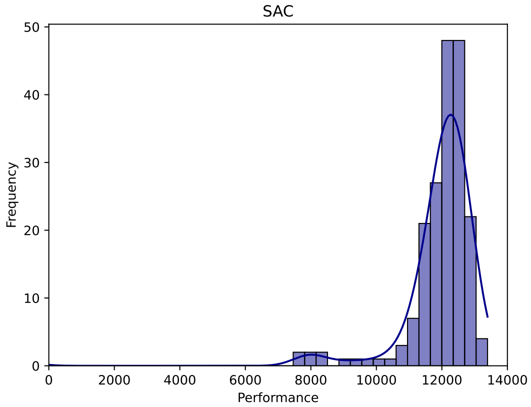
Figure 6: **False positive rates for different distributions, different standard deviations.** x_1 and x_2 are drawn from two different distributions, centered in 0 (mean or median), with $\sigma_1 = 1$ and $\sigma_2 = 2$. (**a**): normal and log-normal distributions. (**b**): bimodal and log-normal distributions.

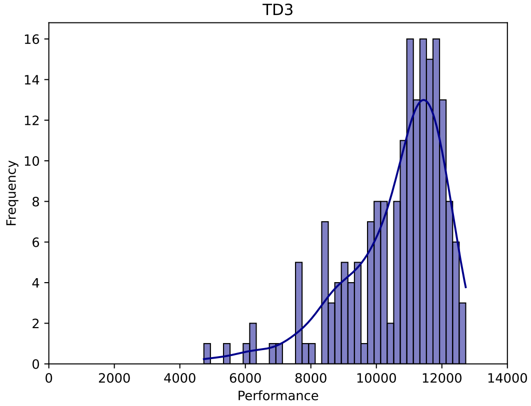
B. L. Welch, "The generalization of student's' problem when several different population variances are involved," Biometrika, vol. 34, no. 1/2, pp. 28–35, 1947.

C. Colas, O. Sigaud, and P.-Y. Oudeyer, "A Hitchhiker's Guide to Statistical Comparisons of Reinforcement Learning Algorithms." arXiv, 2022. doi: [10.48550/arXiv.1904.06979](https://doi.org/10.48550/arXiv.1904.06979).

Example: performance distributions for SAC vs TD3

Visualization of 192 runs





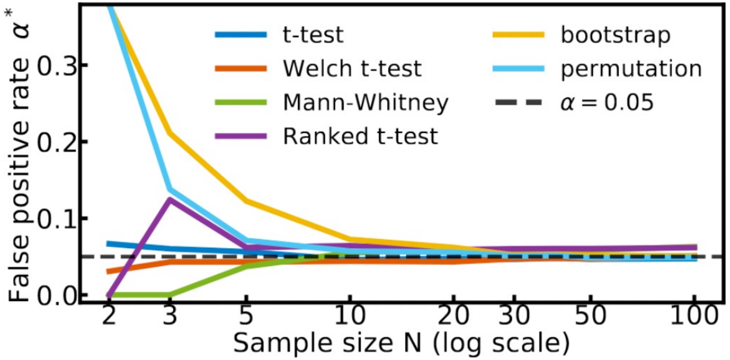


Figure 7: **False positive rates when comparing SAC and TD3.** x_1 is drawn from SAC performances, x_2 from TD3 performances. Both are centered in 0 (mean or median), with $\sigma_1 = 1.313$ and $\sigma_2 = 1.508$.

Half Cheetah

Relative effect size ϵ = 0.93 (mean)

→ 10-15 tests sufficient

C. Colas, O. Sigaud, and P.-Y. Oudeyer, "A Hitchhiker's Guide to Statistical Comparisons of Reinforcement Learning Algorithms." arXiv, 2022. doi: 10.48550/arXiv.1904.06979.

Appendix B

Implementation best practices

For:

- Designing RL methods
- Applying RL methods

Implementation best practices

Randomization

- Control the random seeds for the environment. This allows for more precise “blocking designs” to cancel out nuisance factors, e.g., unlucky initializations, by permitting applying different algorithms or configurations on the same data (environment instance) [1]
- In practice, this means using separate random seeds for the agent and the environment [2].

[1] D. C. Montgomery, “Design and analysis of experiments,” Tenth edition. Hoboken, NJ: Wiley, 2020.
[2] A. Patterson, S. Neumann, M. White, and A. White, “Empirical Design in Reinforcement Learning.” arXiv, Apr. 03, 2023. doi: 10.48550/arXiv.2304.01315.

Implementation best practices

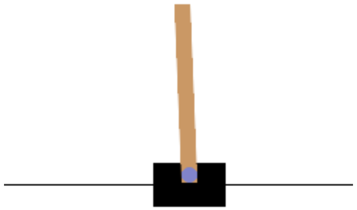
Baselines

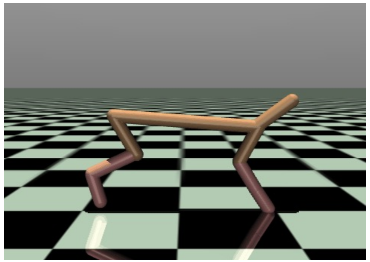
- **Consider simple methods**, such as random search or a constant policy, to indicate if the RL method is learning anything interesting. These can be surprisingly strong baselines.
- **Consider SotA methods for the problem of interest**, to indicate if the RL method is learning something important. These may include non-learning methods, especially for specific applications!
- **Design an oracle method** that has privileged access to information, to indicate the suboptimality gap of the RL method. For example, full state information (vs partially observed) or oracle access to certain training labels.
- **Select comparable hyperparameter sets for all methods**, for a fair comparison. For example, consider the same number of hyperparameter settings for each method.

Implementation best practices

Environments

- Make use of existing benchmarks. Because benchmarks are prone to errors, misspecification, reward hacking, etc. Popular benchmarks are more likely to have these issues worked out (but no guarantee, of course).
- Be wary of perverse (worst-case) examples or random MDPs. These problems, while commonly used for (worst-case) theoretical analysis, may not be representative of problems you ultimately care about [1].
- Most importantly, use tasks that test the claims of the approach.



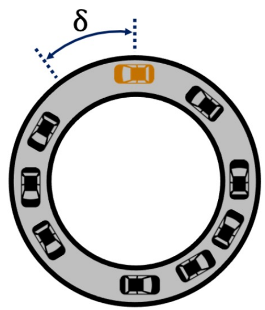


[1] A. Patterson, S. Neumann, M. White, and A. White, “Empirical Design in Reinforcement Learning.” arXiv, Apr. 03, 2023. doi: 10.48550/arXiv.2304.01315.

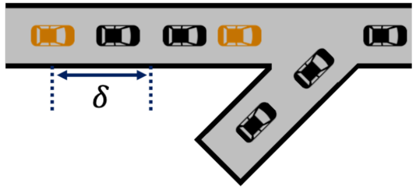
Implementation best practices

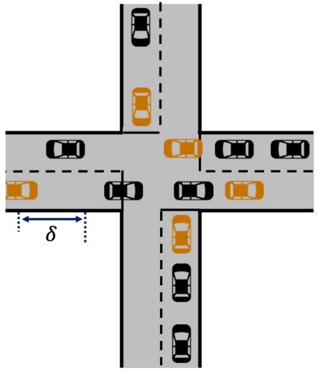
Environments

- Applications of RL - your mileage may vary (YMMV)
  - If benchmarks do not exist for your domain, start with creating simple versions of your application.
  - Consider using the closest available benchmarks to your application.
- Start simple and gradually increase complexity – both for the environment and the agent.
  - Avoid working on image-based environments unless you are specifically interested in pixel-based learning (e.g., sensorimotor control). That extra complexity is often orthogonal to other challenges.



[1]



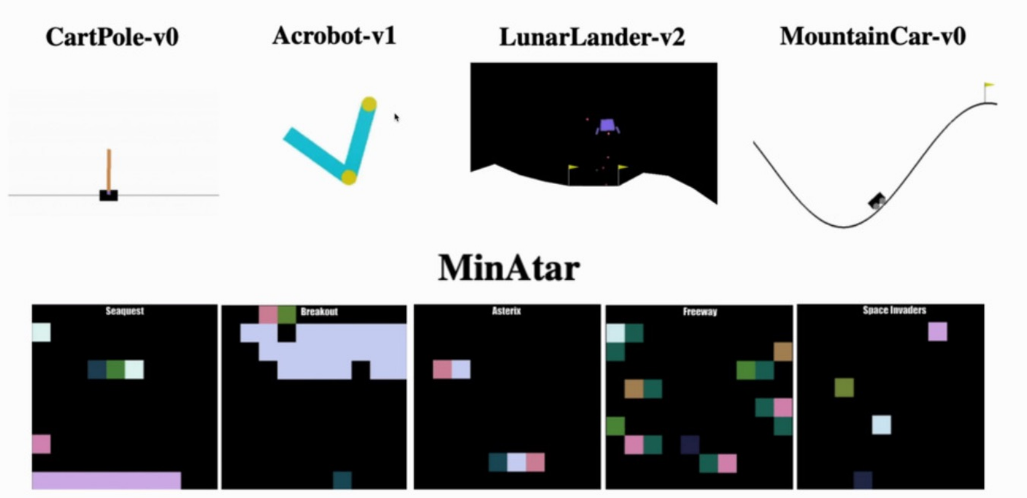


[1] J.-H. Cho, S. Li, J. Kim, and **C. Wu**, "Temporal Transfer Learning for Traffic Optimization with Coarse-grained Advisory Autonomy," arXiv, 27 Nov. 2023, doi:10.48550/arXiv.2312.09436.

Implementation best practices

Environments

- More tasks is not necessarily better. Training is expensive!
  - Example [1]: Similar conclusions to Rainbow DQN using 9 carefully selected small-scale tasks (vs 57 medium-scale games)!
  - 438x less compute to get the same results



[1] Obando-Ceron J. S., Castro P. S. Revisiting Rainbow: Promoting more insightful and inclusive deep reinforcement learning research. ICML, 2021.

Implementation best practices

Comparisons

- Use the Welch's t-test or t-test [1,2]
  - 10-20 trials typically good enough
- For sample efficiency comparison, report environment steps
  - **Not wall clock or CPU time**, which depend on hardware and/or other running processes
  - # samples is a more reproducible measure

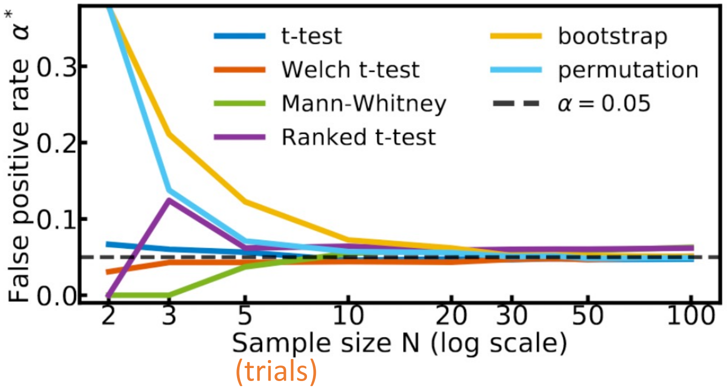


Figure 7: **False positive rates when comparing SAC and TD3.** x_1 is drawn from SAC performances, x_2 from TD3 performances. Both are centered in 0 (mean or median), with $\sigma_1 = 1.313$ and $\sigma_2 = 1.508$.

[1] B. L. Welch, "The generalization of student's' problem when several different population variances are involved," Biometrika, vol. 34, no. 1/2, pp. 28–35, 1947.
[2] C. Colas, O. Sigaud, and P.-Y. Oudeyer, "A Hitchhiker's Guide to Statistical Comparisons of Reinforcement Learning Algorithms." arXiv, 2022. doi: 10.48550/arXiv.1904.06979.

Implementation best practices

Guidance on hyperparameters

- Learning rates
  - A (small-enough) constant learning rate typically suffices
  - Beyond that, annealing the learning rate can help (continually decay the learning rate)
  - Nonstationary environments tend to benefit from smaller learning rates
- Exploration parameters (e.g., ϵ-greedy)
  - Start with higher exploration, gradually decay it to some lower limit, e.g. $\epsilon = 0.05$.
  - Ensure that there is enough training time near the lower limit, for training stability

Implementation best practices

Guidance on hyperparameters

- For standard Mujoco benchmarks, see comprehensive hyperparameter study [1]
  - Caveat 1: **YMMV for other benchmarks / tasks**.
  - Caveat 2: YMMV even for Mujoco benchmarks, because the hyperparameter sweeps are conducted **locally** wrt PPOv2.
  - Recommendations (see paper for theories behind the recommendations):
    - Normalize normalize normalize – reward scaling, input normalization, value function normalization
    - Use separate networks for the value fn & policy
    - Limit exploration at the start. In practice, this means setting a low initial std for the policy network
    - tanh activation $\gg$ ReLU
    - MSE $\gg$ Huber loss for the value function loss
    - …

[1] Andrychowicz, *et al.*, "What matters in on-policy reinforcement learning? A large-scale empirical study," ICLR, 2021.

Implementation best practices

Replay buffer

- Needs considerable capacity to perform well
  - Typical sizes are 10K–1M.
- Take care at the start of training: Collect enough initial "experience" by allowing the replay buffer to fill up part way, before starting to update the value function

Implementation best practices

Optimization

- Take multiple passes through collected data
- Use mini-batch gradient descent, rather than SGD or batch gradient descent
  - Typical mini-batch sizes range from 32 to 1024
- Optimizers: Adam (RMSprop + momentum) and RMSprop are both sensible
  - RMSprop is preferred for value-based methods (more stable, less sensitive to hyperparameters)

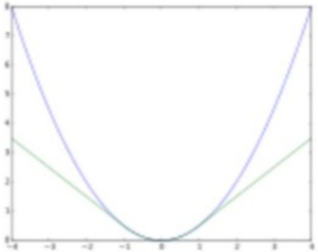
More practical tips (via Eugene Vinitsky, 2024)

- Use fast RL environments rather than slow environments
  - Collecting trajectories is often the bottleneck in RL training
  - Fast environment implementations can easily collect trajectories 100x faster
  - Examples: pufferlib and vmas
  - Can then iterate much faster on RL algorithms
- Use a hyperparameter tuner instead of trying to grid search.
  - Now even directly built into weights and biases: https://docs.wandb.ai/tutorials/sweeps/
- There is growing evidence that classification losses outperform regression losses: https://arxiv.org/pdf/2403.03950

More practical tips (via John Schulman circa 2018)

- DQN is more reliable on some Atari tasks than others. Pong is a reliable task: if it doesn't achieve good scores, something is wrong
- Large replay buffers improve robustness of DQN, and memory efficiency is key
- Use uint8 images, don't duplicate data
- Be patient. DQN converges slowly—for ATARI it's often necessary to wait for 10-40M frames (couple of hours to a day of training on GPU) to see results significantly better than random policy
- Try Huber loss on Bellman error

$$L(x) = \begin{cases} \frac{x^2}{2} & \text{if } |x| \leq \delta \\ \delta|x| - \frac{\delta^2}{2} & \text{otherwise} \end{cases}$$



More practical tips (via John Schulman circa 2018)

- Try Double DQN—significant improvement from small code change
- To test out your data pre-processing, try your own skills at navigating the environment based on processed frames
- Always run at least two different seeds when experimenting
- Learning rate scheduling is beneficial. Try high learning rates in initial exploration period
- Try non-standard exploration schedules

A final note

- If you noticed some inconsistencies in the tips, it's because RL is still an art as well as a science. Your experience may vary too.

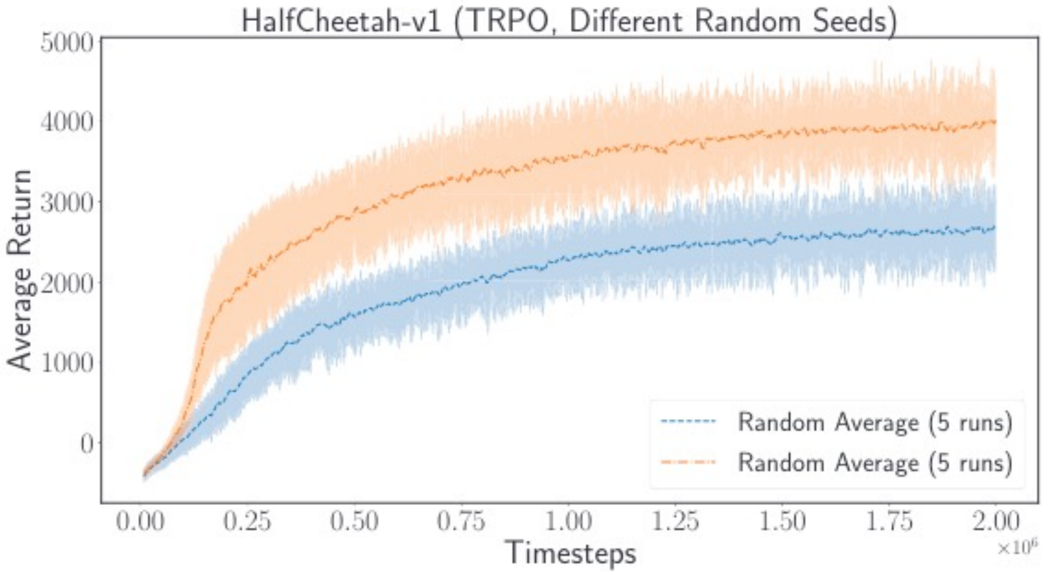
- And feel free to get in touch—including in the years to come—if you have suggestions on implementation best practices.

Appendix C

Additional examples of RL sensitivity

Example: Sensitivity of RL to random seeds

Half Cheetah



Difference is "statistically significant"

Henderson, Islam, Bachman, Pineau, Precup, Meger, **Deep Reinforcement Learning That Matters**, *AAAI Conference on Artificial Intelligence*, 2018.

Sensitivity of RL to hyperparameters

- Example hyperparameter sweep for RL methods

	Hyperparameter	Full Space	Small Space	LR Only
PPO	leaning_rate	log(interval(1e-6, 0.1))	log(interval(1e-6, 0.1))	log(interval(1e-6, 0.1))
	ent_coef	interval(0.0, 0.5)	interval(0.0, 0.5)	
	n_epochs	range[5,20]	range[5,20]	
	batch_size	{16, 32, 64, 128}		
	n_steps	{256, 512, 1024, 2048, 4096}		
	gae_lambda	interval(0.8, 0.9999)		
	clip_range	interval(0.0, 0.5)		
	clip_range_vf	interval(0.0, 0.5)		
	normalize_advantage	{True, False}		
	vf_coef	interval(0.0, 1.0)		
	max_grad_norm	interval(0.0, 1.0)		
SAC	leaning_rate	log(interval(1e-6, 0.1))	log(interval(1e-6, 0.1))	log(interval(1e-6, 0.1))
	train_freq	range[1,1e3]	range[1,1e3]	
	tau	interval(0.01, 1.0)	interval(0.01, 1.0)	
	batch_size	{64, 128, 256, 512}		
	learning_starts	range[0,1e4]		
	buffer_size	range[5e3,5e7]		
	gradient_steps	range[1,10]		
DQN	learning_rate	log(interval(1e-6, 0.1))	log(interval(1e-6, 0.1))	log(interval(1e-6, 0.1))
	batch_size	{4, 8, 16, 32}	{4, 8, 16, 32}	
	exploration_fraction	interval(0.005, 0.5)	interval(0.005, 0.5)	
	learning_starts	range[0,1e4]		
	train_freq	range[1,1e3]		
	gradient_steps	range[1,10]		
	exploration_initial_eps	interval(0.5, 1.0)		
	exploration_final_eps	interval(0.001, 0.2)		
	buffer_size	range[5e3,5e7]		

Table 4: StableBaselines search spaces.

T. Eimer, M. Lindauer, and R. Raileanu, “Hyperparameters in Reinforcement Learning and How To Tune Them.” arXiv, 2023. doi: 10.48550/arXiv.2306.01324. Wu

Example: Sensitivity of RL to hyperparameters

- Same algorithm has different sensitivities in different environments

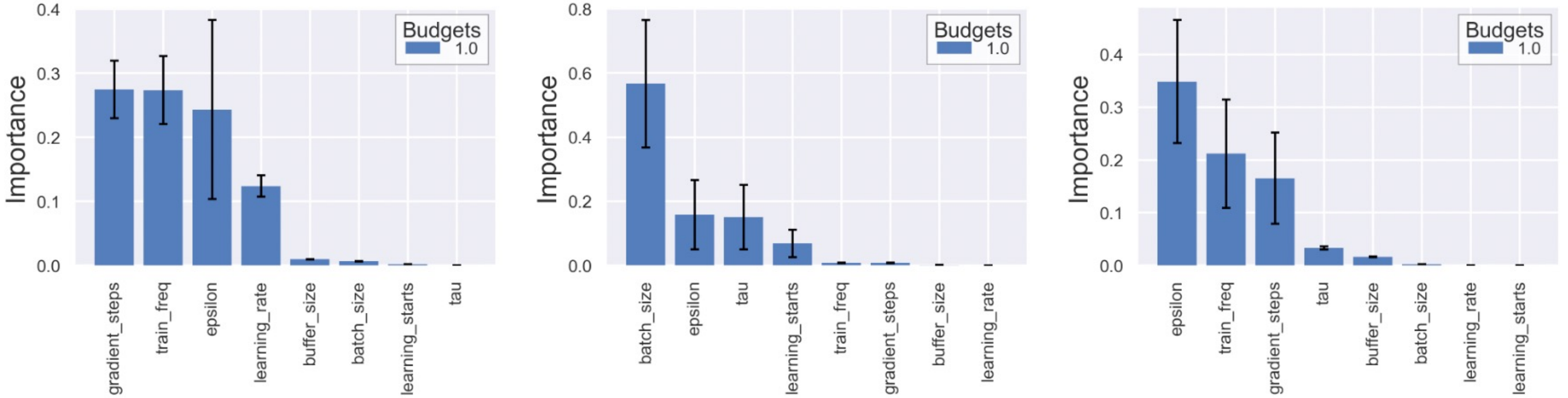
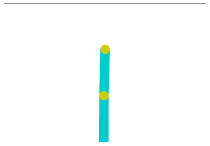
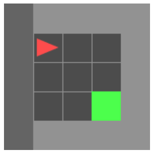


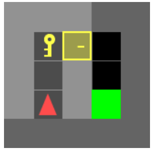
Figure 28: DQN Hyperparameter Importances on Acrobot (left), MiniGrid Empty (middle) and MiniGrid DoorKey (right).



OpenAI's Acrobot



MiniGrid Empty



MiniGrid DoorKey

T. Eimer, M. Lindauer, and R. Raileanu, "Hyperparameters in Reinforcement Learning and How To Tune Them." arXiv, 2023. doi: 10.48550/arXiv.2306.01324. Wu

Example: Sensitivity of RL to hyperparameters

- Same algorithm has different sensitivities in different environments

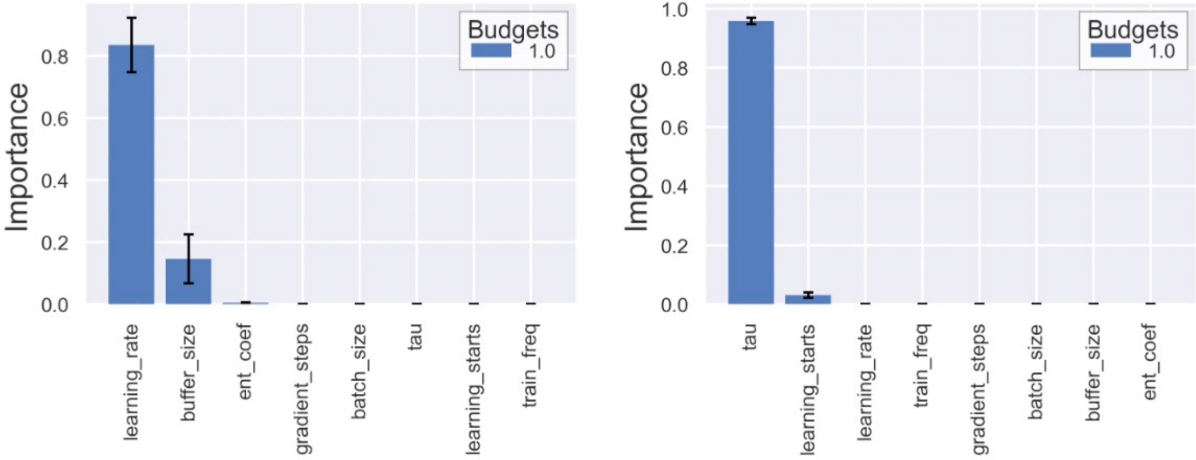
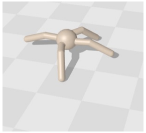


Figure 29: SAC Hyperparameter Importances on Pendulum (left) and Brax Ant(right).

OpenAI's pendulum



Brax Ant

(fully differentiable)

T. Eimer, M. Lindauer, and R. Raileanu, "Hyperparameters in Reinforcement Learning and How To Tune Them." arXiv, 2023. doi: 10.48550/arXiv.2306.01324. Wu

Example: Sensitivity of RL to discount factor

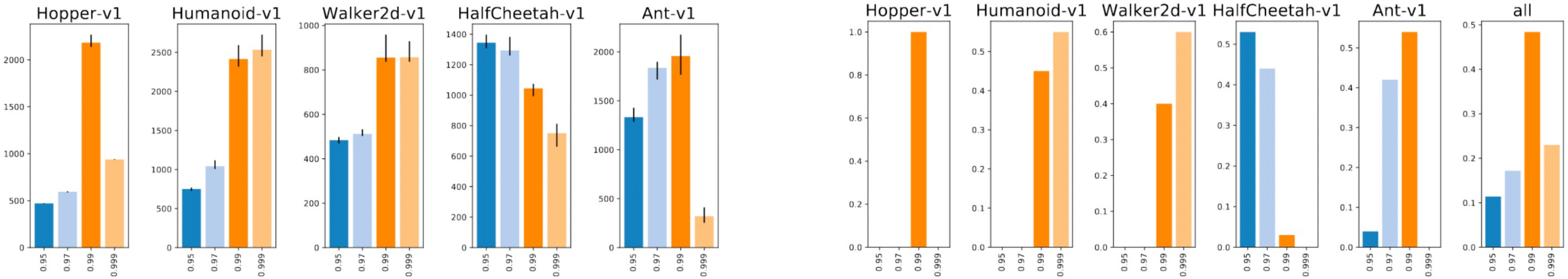
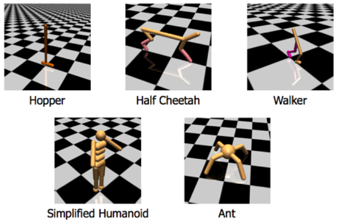
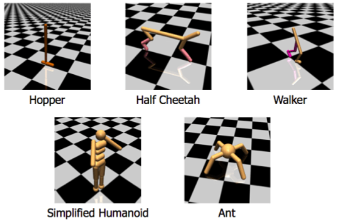


Figure 60: Analysis of choice `Discount factor` γ (C20): 95th percentile of performance scores conditioned on choice (left) and distribution of choices in top 5% of configurations (right).



Andrychowicz, *et al.*, "What matters in on-policy reinforcement learning? A large-scale empirical study," International conference on learning representations (ICLR), 2021.

Example: *Ins*ensitivity of RL to "frame skip"



- Persist action for k frames (time steps). Popularly used for Atari [1].
- *Not so sensitive* in these Mujoco tasks for $k \in \{1,2,5\}$. [2]

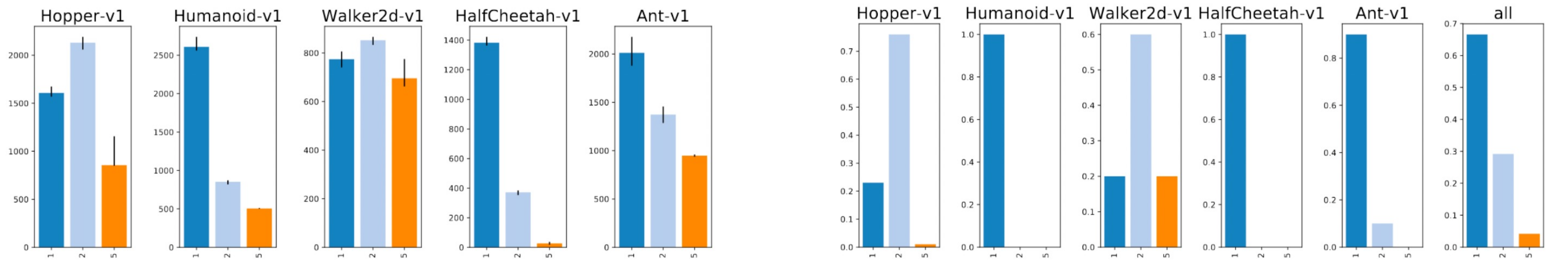
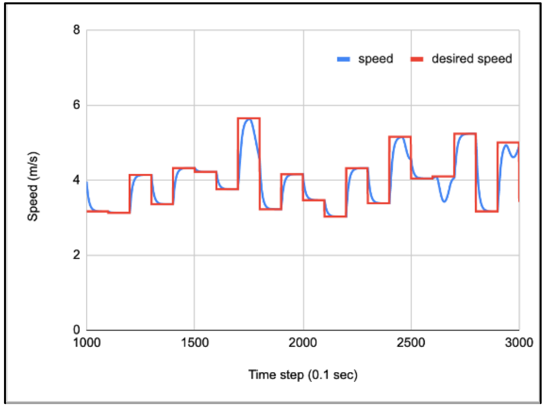


Figure 61: Analysis of choice `Frame skip` (C21): 95th percentile of performance scores conditioned on choice (left) and distribution of choices in top 5% of configurations (right).

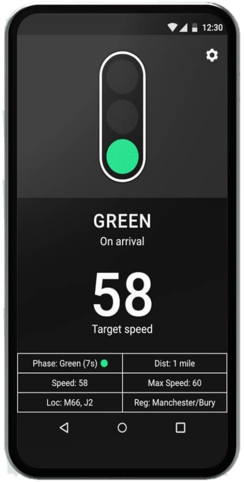
[1] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The Arcade Learning Environment: An Evaluation Platform for General Agents," *JAIR*, 2013, doi: 10.1613/jair.3912.
[2] Andrychowicz, *et al.*, "What matters in on-policy reinforcement learning? A large-scale empirical study," International conference on learning representations (ICLR), 2021.

Example: Sensitivity of RL to "frame skip"

- Caution: With each of these factors, your mileage may vary.
- Example: Consider control frequency in a driving application [1].
  - Equivalent to frame skip $k \in \{1, \ldots, 400\}$



Velocity-based advisories



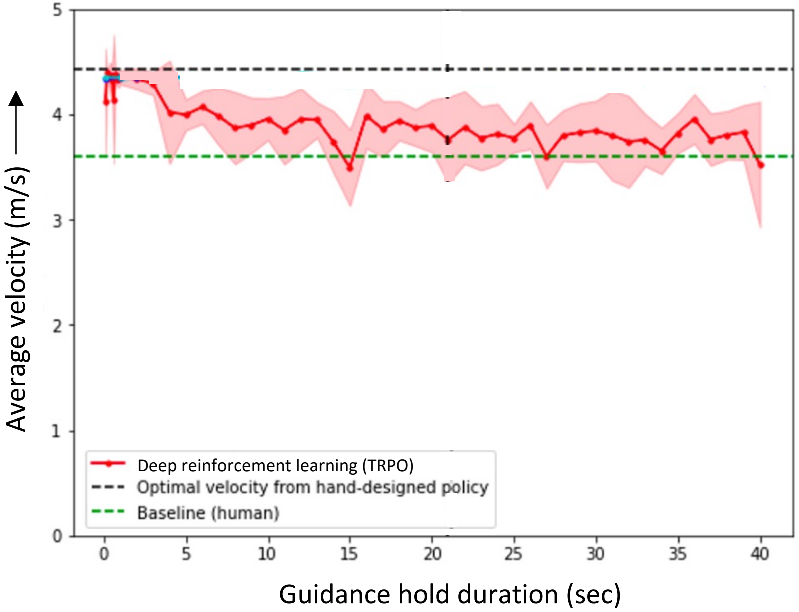
GLOSA Demo app [2]

<u>Terminology for similar ideas</u>:
Frame skip (video game development)
Zero-order hold (control theory)
Action persistence (deep RL)
Action repetition (deep RL)
Control frequency (control theory)

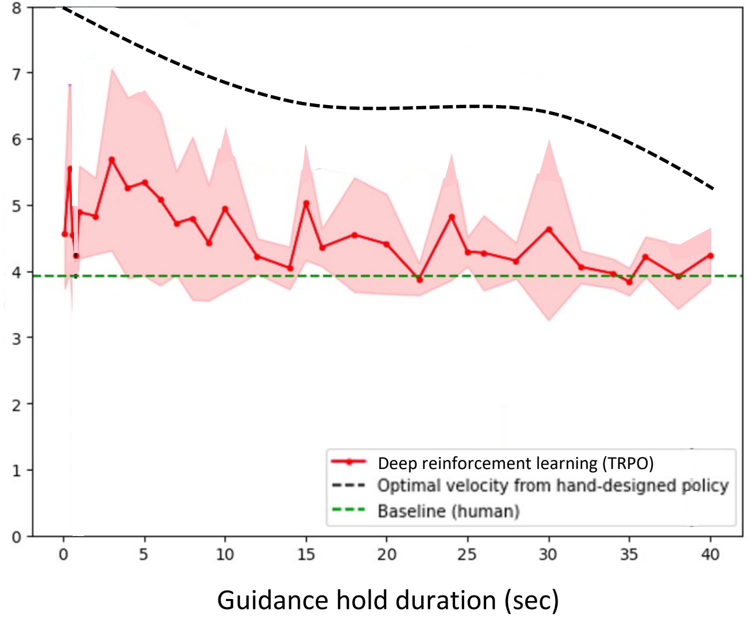
[1] Cho, Li, Kim, <u>Wu</u>. "Temporal Transfer Learning for Traffic Optimization with Coarse-grained Advisory Autonomy." In review, T-RO.
[2] Reducing fuel emissions with innovative tech, 2018. https://www.eastpoint.co.uk/case-studies/glosa/

Example: Sensitivity of RL to "frame skip"

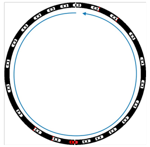
Model highway (acceleration guidance)

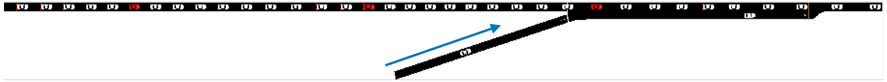


Highway ramp (velocity guidance)



Tasks: Equivalent to frame skip $k \in \{1, \ldots, 400\}$
Method: TRPO
Train: 5 trials per hold duration
Test: 50 test epochs using the best trained policy





Cho, Li, Kim, Wu. "Temporal Transfer Learning for Traffic Optimization with Coarse-grained Advisory Autonomy." In review, T-RO.