# Risk-Aware Transfer in Reinforcement Learning using Successor Features

Michael Gimelfarb[1,3] André Barreto[2] Scott Sanner[1,3] Chi-Guhn Lee[1]

[1]University of Toronto
[2]DeepMind
[3]Vector Institute (Affiliate Program)

# Introduction

# Motivation

# Motivation

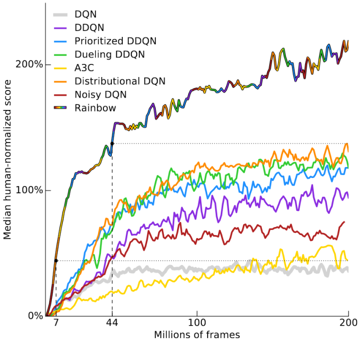



**Figure 1:** Sample Efficiency (Source: original paper on Rainbow DQN)


**Figure 2:** Risk-Awareness (Source: Wikimedia Commons)

# Motivation

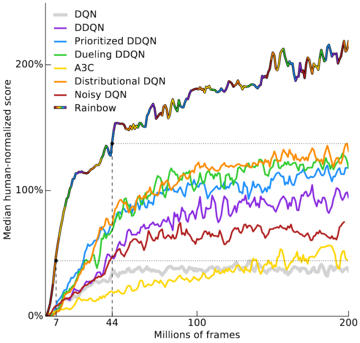



**Figure 1:** Sample Efficiency (Source: original paper on Rainbow DQN)


**Figure 2:** Risk-Awareness (Source: Wikimedia Commons)

- **transfer learning**
- replace E[ ] by non-linear **utility** $U[\,]$

# Motivation

Our goals:

# Motivation

Our goals:

- transfer between tasks with **shared dynamics and different goals**

## Motivation

Our goals:

- transfer between tasks with **shared dynamics and different goals**
- borrow **GPI/GPE** (e.g. successor features) from the risk-neutral setting[1]

[1] Barreto, Andre, et al. "Successor Features for Transfer in Reinforcement Learning." NIPS. 2017.

# Motivation

Our goals:

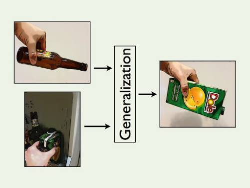
- transfer between tasks with **shared dynamics and different goals**
- borrow **GPI/GPE** (e.g. successor features) from the risk-neutral setting[1]
- provide **task generalization** by exploiting the structure of the task/reward space

[1] Barreto, Andre, et al. "Successor Features for Transfer in Reinforcement Learning." NIPS. 2017.

# Motivation

Our goals:

- transfer between tasks with **shared dynamics and different goals**
- borrow **GPI/GPE** (e.g. successor features) from the risk-neutral setting[1]
- provide **task generalization** by exploiting the structure of the task/reward space
- design a method suitable for offline RL[2]

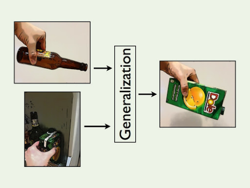



[1] Barreto, Andre, et al. "Successor Features for Transfer in Reinforcement Learning." NIPS. 2017.
[2] Levine, Sergey, et al. "O ine reinforcement learning..." arXiv. 2020.

# Motivation

Our goals:

- transfer between tasks with **shared dynamics and different goals**
- borrow **GPI/GPE** (e.g. successor features) from the risk-neutral setting[1]
- provide **task generalization** by exploiting the structure of the task/reward space
- design a method suitable for offline RL[2]




- incorporate risk explicitly by e.g. penalizing the variance of returns

[1] Barreto, Andre, et al. "Successor Features for Transfer in Reinforcement Learning." NIPS. 2017.
[2] Levine, Sergey, et al. "O ine reinforcement learning..." arXiv. 2020.

# Motivation

Introduce **Risk-Aware Successor Features** (RaSF)

# Motivation

Introduce **Risk-Aware Successor Features** (RaSF)

| | Transfers Skills | Exploits Task Structure | Risk-Sensitive |
|---|---|---|---|
| Risk-Aware RL | 7 | 7 | 3 |
| Risk-Aware Transfer | 3 | 7 | 3 |
| Successor Features | 3 | 3 | 7 |
| **RaSF (Ours)** | 3 | 3 | 3 |

# Preliminaries – Successor Features

# Policy Evaluation

## Policy Evaluation

- $\mathrm{E}[R_0 + \quad R_1 + \ldots]$ requires averaging all possible future outcomes of the world – **curse of dimensionality**

# Policy Evaluation

- $E[R_0 + \quad R_1 + \ldots]$ requires averaging all possible future outcomes of the world – **curse of dimensionality**
- use cached $Q(s', a')$ in each successor state to **bootstrap** the estimated Q-values in state $s$

$$Q(s, a) = E_{S' \quad P(\ |s,a)}[R_t + \quad Q(S', \quad (S'))]$$

# Policy Improvement

## Policy Improvement

Suppose an initial policy   is given:

Suppose an initial policy is given:

- compute the value of , e.g.

$$Q\ (s;a) = E_{S^0\ P(\ j s;a)}[R_t\ +\ \ Q(S^0;\ \ (S^0))] \qquad /\ \text{policy evaluation}$$

Suppose an initial policy is given:

ˆ compute the value of , e.g.

$$Q\ (s; a) = E_{S^0\ P(\ |s;a)}[R_t + \ Q(S^0;\ (S^0))] \qquad / \text{ policy evaluation}$$

ˆ construct a new policy $^0$ according to

$$^0(s)\ 2\ \arg\max_{a2A} Q\ (s; a) \qquad / \text{ policy improvement}$$

# Policy Improvement

Suppose an initial policy is given:

- compute the value of , e.g.

$$Q\ (s; a) = E_{S^0\ P(\ j s;a)}[R_t + \ Q(S^0; \ (S^0))] \qquad / \ \text{policy evaluation}$$

- construct a new policy $^0$ according to

$${}^0(s)\ 2\ \arg\max_{a2A} Q\ (s; a) \qquad / \ \text{policy improvement}$$

Policy Improvement Theorem: $^0$ is \better'' than , e.g. $Q\ ^0(s; a) \quad Q\ (s; a)$

# Generalized Policy Iteration

# Generalized Policy Iteration

Alternating between evaluation and improvement leads to an optimal policy

# Generalized Policy Iteration

Alternating between evaluation and improvement leads to an optimal policy

Key Idea: Replace with multiple source policies ${}_1 \ldots {}_n$.

# Generalized Policy Improvement 2.0

# Generalized Policy Improvement 2.0

Suppose $_1; \ldots \; _n$ are given[1]:

[1] Barreto, Andre, et al. "Successor Features for Transfer in Reinforcement Learning." NeurIPS 2017.

# Generalized Policy Improvement 2.0

Suppose ${}_1; \ldots\; {}_n$ are given[1]:

- compute the values

$$Q^{1}(s;a); \ldots Q^{n}(s;a) \qquad / \quad \text{generalized policy evaluation (GPE)}$$

[1] Barreto, Andre, et al. "Successor Features for Transfer in Reinforcement Learning." NeurIPS 2017.

# Generalized Policy Improvement 2.0

Suppose $_1, \ldots, _n$ are given[1]:

- compute the values

$$Q^{\;1}(s,a), \ldots, Q^{\;n}(s,a) \qquad / \quad \text{generalized policy evaluation (GPE)}$$

- pick the policy with the best return

$$i \in \arg\max_i Q^{\;i}(s,a)$$

[1] Barreto, Andre, et al. "Successor Features for Transfer in Reinforcement Learning." NeurIPS 2017.

# Generalized Policy Improvement 2.0

Suppose $\pi_1, \ldots, \pi_n$ are given[1]:

- compute the values

$$Q^{\pi_1}(s,a), \ldots, Q^{\pi_n}(s,a) \qquad \leftarrow \text{ generalized policy evaluation (GPE)}$$

- pick the policy with the best return

$$i \in \arg\max_i Q^{\pi_i}(s,a)$$

- construct $\pi'$ as usual but w.r.t. the "best" policy $\pi_i$

$$\pi'(s) \in \arg\max_{a \in A} Q^{\pi_i}(s,a) = \arg\max_{a \in A} \max_{i=1 \ldots n} Q^{\pi_i}(s,a)$$

$$\leftarrow \text{ generalized policy improvement (GPI)}$$

[1] Barreto, Andre, et al. "Successor Features for Transfer in Reinforcement Learning." NeurIPS 2017.

# Generalized Policy Improvement 2.0

Suppose $_1, \ldots\ _n$ are given[1]:

- compute the values

$$Q^{\ 1}(s,a), \ldots, Q^{\ n}(s,a)$$

/ generalized policy evaluation (GPE)

- pick the policy with the best return

$$i \ 2 \ \arg\max_i Q^{\ i}(s,a)$$

- construct $^0$ as usual but w.r.t. the \best" policy $_i$

$$^0(s) \ 2 \ \arg\max_{a2A} Q^{\ i}(s,a) = \arg\max_{a2A} \max_{i=1:::n} Q^{\ i}(s,a)$$

/ generalized policy improvement (GPI)

Key result: $^0$ is better than $_i$ .

[1] Barreto, Andr e, et al. "Successor Features for Transfer in Reinforcement Learning." NeurIPS 2017.

# Preliminaries { Risk-Aversion in MDPs using Entropic Utility Functions

# Time-Consistency

# Time-Consistency

Optimizing risk measures in sequential problems is hard[2]

[2] Chow, Yin-Lam, and Marco Pavone. "A framework for time-consistent, risk-averse model predictive control: Theory and algorithms." 2014 American Control Conference. IEEE, 2014.

# Time-Consistency

Optimizing risk measures in sequential problems is hard[2]:

- considering optimizing a nal cost $Z$:

[2] Chow, Yin-Lam, and Marco Pavone. "A framework for time-consistent, risk-averse model predictive control: Theory and algorithms." 2014 American Control Conference. IEEE, 2014.

Optimizing risk measures in sequential problems is hard[2]:

- considering optimizing a  nal cost $Z$:

- consider the dynamic risk measure

$$
_{k;N}(Z) = \max_{p \in \{0.4;0.6\}} E_p[Z | F_k];
$$

for $k = 0; 1; 2$

[2] Chow, Yin-Lam, and Marco Pavone. "A framework for time-consistent, risk-averse model predictive control: Theory and algorithms." 2014 American Control Conference. IEEE, 2014.

Optimizing risk measures in sequential problems is hard[2]:

- considering optimizing a final cost $Z$:
  - it is easy to check that ${}_1(Z)(\omega) = 60$ for all $\omega$, so $Z$ is riskier than a deterministic cash flow of $W = 50$ at time 1
- consider the dynamic risk measure

$$ {}_{k;N}(Z) = \max_{p \in \{0.4; 0.6\}} E_p[Z \mid F_k]; $$

  for $k = 0; 1; 2$

[2] Chow, Yin-Lam, and Marco Pavone. "A framework for time-consistent, risk-averse model predictive control: Theory and algorithms." 2014 American Control Conference. IEEE, 2014.

# Time-Consistency

Optimizing risk measures in sequential problems is hard[2]:

- considering optimizing a nal cost $Z$:
- consider the dynamic risk measure

$$_{k;N}(Z) = \max_{p \in \{0.4, 0.6\}} E_p[Z | F_k];$$

for $k = 0; 1; 2$

- it is easy to check that $_1(Z)(!) = 60$ for all $!$, so $Z$ is riskier than a deterministic cash ow of $W = 50$ at time 1
- yet, $_0(Z)(!) = 40$ and so $Z$ is less risky than $W$ at time 0!

[2]Chow, Yin-Lam, and Marco Pavone. "A framework for time-consistent, risk-averse model predictive control: Theory and algorithms." 2014 American Control Conference. IEEE, 2014.

Optimizing risk measures in sequential problems is hard[2]:

- considering optimizing a final cost $Z$:
  - it is easy to check that $\rho_1(Z)(\omega) = 60$ for all $\omega$, so $Z$ is riskier than a deterministic cash flow of $W = 50$ at time 1
  - yet, $\rho_0(Z)(\omega) = 40$ and so $Z$ is less risky than $W$ at time 0!
- consider the dynamic risk measure

$$\rho_{k;N}(Z) = \max_{p \in \{0.4, 0.6\}} E_p[Z \mid \mathcal{F}_k];$$

for $k = 0, 1, 2$

Key idea: $Z$ has become riskier just because time has passed!

[2] Chow, Yin-Lam, and Marco Pavone. "A framework for time-consistent, risk-averse model predictive control: Theory and algorithms." 2014 American Control Conference. IEEE, 2014.

# Variance Reduction in MDPs with Entropic Utilities

# Variance Reduction in MDPs with Entropic Utilities

We use the entropic utility to measure risk:

# Variance Reduction in MDPs with Entropic Utilities

We use the **entropic utility** to measure risk:

- de ned in terms of the moment-generating function

$$U\ [R] = \frac{1}{\ } \log \mathbb{E}\left[e^{\ R}\right]$$

# Variance Reduction in MDPs with Entropic Utilities

We use the entropic utility to measure risk:

- defined in terms of the moment-generating function

$$U\,[R] = \frac{1}{\ } \log E\left[e^{\ R}\right]$$

- has the Taylor expansion

$$U\,[R] = E[R] + \frac{\ }{2} V[R] + O(\ ^2)$$

/ is a level of risk-aversion

# Variance Reduction in MDPs with Entropic Utilities

We use the entropic utility to measure risk:

- defined in terms of the moment-generating function

$$U_\beta[R] = \frac{1}{\beta}\log \mathbb{E}\left[e^{\beta R}\right]$$

- has the Taylor expansion

$$U_\beta[R] = \mathbb{E}[R] + \frac{\beta}{2}\mathbb{V}[R] + O(\beta^2)$$ / $\beta$ is a level of risk-aversion

- connected to the mean-variance optimization in MDPs[1]

[1]Mannor, Shie, and John N. Tsitsiklis. "Mean-variance optimization in Markov decision processes." ICML. 2011.

We incorporate entropic utility into MDPs:

# Variance Reduction in MDPs with Entropic Utilities

We incorporate entropic utility into MDPs:

- dynamic programming: has a Bellman equation formulation

# Variance Reduction in MDPs with Entropic Utilities

We incorporate entropic utility into MDPs:

- dynamic programming: has a Bellman equation formulation

- recursive property: behaves similar to expectation in total-reward episodic MDPs[1] (and discounted MDPs with simple modi cations)

[1] Osogami, Takayuki. "Robustness and risk-sensitivity in Markov decision processes." NeurIPS 2012.

# Variance Reduction in MDPs with Entropic Utilities

We incorporate entropic utility into MDPs:

- dynamic programming: has a Bellman equation formulation
- recursive property: behaves similar to expectation in total-reward episodic MDPs[1] (and discounted MDPs with simple modi cations)
- convex/concave: satis es properties that can be seen as rational decision making

[1] Osogami, Takayuki. "Robustness and risk-sensitivity in Markov decision processes." NeurIPS 2012.

# Variance Reduction in MDPs with Entropic Utilities

We incorporate entropic utility into MDPs:

- dynamic programming: has a Bellman equation formulation
- recursive property: behaves similar to expectation in total-reward episodic MDPs[1] (and discounted MDPs with simple modi cations)
- convex/concave: satis es properties that can be seen as rational decision making
- time consistency: can focus on Markov policies

[1]Osogami, Takayuki. "Robustness and risk-sensitivity in Markov decision processes." NeurIPS 2012.

# Roadmap

# Theory

# Motivating Example

# Motivating Example

Why is the problem non-trival?

# Motivating Example

Why is the problem non-trival?

# Motivating Example

1. De ne a family of tasks:

   ˆ two source tasks:

   low failure cost + high failure cost

   ˆ one target task:

   only X has high failure cost

## Motivating Example

1. De ne a family of tasks:
   - two source tasks:
     low failure cost + high failure cost
   - one target task:
     only X has high failure cost

2. Solve them with VI for xed

# Motivating Example

1. De ne a family of tasks:
   - two source tasks:
     low failure cost + high failure cost
   - one target task:
     only X has high failure cost

2. Solve them with VI for xed

3. Apply risk-aware and risk-neutral GPI:

## Motivating Example

1. De ne a family of tasks:
   - two source tasks:
     low failure cost + high failure cost
   - one target task:
     only X has high failure cost
2. Solve them with VI for xed
3. Apply risk-aware and risk-neutral GPI:

Conclusion: only risk-aware GPI results in the correct target policy

# Key Theoretical Results

# Key Theoretical Results

Armed with this knowledge, we prove that risk-aware GPI:

## Key Theoretical Results

Armed with this knowledge, we prove that risk-aware GPI:

- is a strict policy improvement operator

# Key Theoretical Results

Armed with this knowledge, we prove that risk-aware GPI:

- is a strict policy improvement operator

- is optimal up to an irreducible task discrepancy gap

# Generalized Policy Evaluation

Assume linear reward:

$$r(s; a; s') = \quad (s; a; s')^{\top} \, w$$

# Generalized Policy Evaluation

Assume linear reward:

$$r(s,a,s') = \phi(s,a,s')^\top \mathbf{w}$$

Now:

$$
\begin{aligned}
Q_{\mathbf{w}}(s,a) &= \mathbb{E}\left[\sum_t \gamma^t R_t \,\middle|\, S_0 = s, A_0 = a, A_t \sim \pi(S_t)\right] \\
&= \mathbb{E}\left[\sum_t \gamma^t \phi_t^\top \mathbf{w} \,\middle|\, S_0 = s, A_0 = a, A_t \sim \pi(S_t)\right] \\
&= \underbrace{\mathbb{E}\left[\sum_t \gamma^t \phi_t \,\middle|\, S_0 = s, A_0 = a, A_t \sim \pi(S_t)\right]^\top}_{\psi(s,a)} \mathbf{w}
\end{aligned}
$$

# Risk-Aware GPE

# Risk-Aware GPE

Can generalize GPE to distributions of return:

$$Q_{h,\pi}(s,a) = U\left[\sum_{t=h}^{T} r(s_t, \pi_t(s_t), s_{t+1})\right] = U\left[\psi_h(s,a)^\intercal w\right]$$

Can generalize GPE to distributions of return :

$$Q_{h;}(s;a) = U\left[\sum_{t=h}^{T} r(s_t;\ _t(s_t);s_{t+1})\right] = U[\ _h(s;a)^\intercal w]$$

One simple trick is to Taylor expand to the second moment:

$$U[\ _h(s;a)^\intercal w] = E_P[\ _h(s;a)^\intercal w] + \frac{}{2}\mathrm{Var}_P[\ _h(s;a)^\intercal w] + O(\ ^2)$$

$$\ _h(s;a)^\intercal w + \frac{}{2} w^\intercal \mathrm{Var}_P[\ _h(s;a)]w = \tilde{Q}_{h;}(s;a)$$

# Risk-Aware GPE

Can generalize GPE to distributions of return :

$$Q_{h;}(s;a) = U\left[\sum_{t=h}^{T} r(s_t;\ {}_t(s_t); s_{t+1})\right] = U[\ {}_h(s;a)^{\intercal} w]$$

One simple trick is to Taylor expand to the second moment:

$$U[\ {}_h(s;a)^{\intercal} w] = E_P[\ {}_h(s;a)^{\intercal} w] + \frac{\ }{2}\mathrm{Var}_P[\ {}_h(s;a)^{\intercal} w] + O(\ ^2)$$

$$\ {}_h(s;a)^{\intercal} w + \frac{\ }{2} w^{\intercal}\,\mathrm{Var}_P[\ {}_h(s;a)] w = \tilde{Q}_{h;}(s;a)$$

Reduces to a (simpler) problem of estimating su cient statistics of the feature occupancy

Experiments

# Domains

# Domains

Two domains from Barreto et al., 2017:

# Domains

Two domains from Barreto et al., 2017:

Introduce reward volatility :

- traps X for four-room
- action noise + danger zones for reacher

# Four-Room

# Four-Room

Train on a sequence of 128 random task instances, for 2,000 steps each

# Four-Room

Train on a sequence of 128 random task instances, for 20,000 steps each

## Four-Room

Sensitivity to parameter:

# Four-Room

Sensitivity to parameter:

# Reacher

## Reacher

Train on four source tasks, test periodically on 8 unseen test tasks:

# Reacher

Train on four source tasks, test periodically on 8 unseen test tasks:

# Reacher

Does the agent learn risk-sensitive behavior?

# Reacher

Does the agent learn risk-sensitive behavior?

# Reacher

How sensitive is the agent to ? Does the C51 method help in learning SFs?

# Reacher

How sensitive is the agent to  ? Does the C51 method help in learning SFs?

# Conclusion

# Conclusion

In conclusion:

# Conclusion

In conclusion:

- we presented Risk-aware Successor Features (RaSFs) for realizing policy transfer in domains where tasks have different goals

# Conclusion

In conclusion:

- we presented Risk-aware Successor Features (RaSFs) for realizing policy transfer in domains where tasks have different goals
- we extended generalized policy improvement to the risk-aware setting with entropic utilities

# Conclusion

In conclusion:

- we presented Risk-aware Successor Features (RaSFs) for realizing policy transfer in domains where tasks have different goals
- we extended generalized policy improvement to the risk-aware setting with entropic utilities
- we then extended the notion of generalized policy evaluation via the Taylor expansion of the entropic utility

# Conclusion

In conclusion:

- we presented Risk-aware Successor Features (RaSFs) for realizing policy transfer in domains where tasks have different goals
- we extended generalized policy improvement to the risk-aware setting with entropic utilities
- we then extended the notion of generalized policy evaluation via the Taylor expansion of the entropic utility
- together, risk-aware GPI and GPE are shown to inherit the superior task generalization abilities of successor features, while also learning to avoid risky situations

**Thank you.**