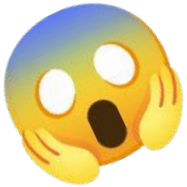
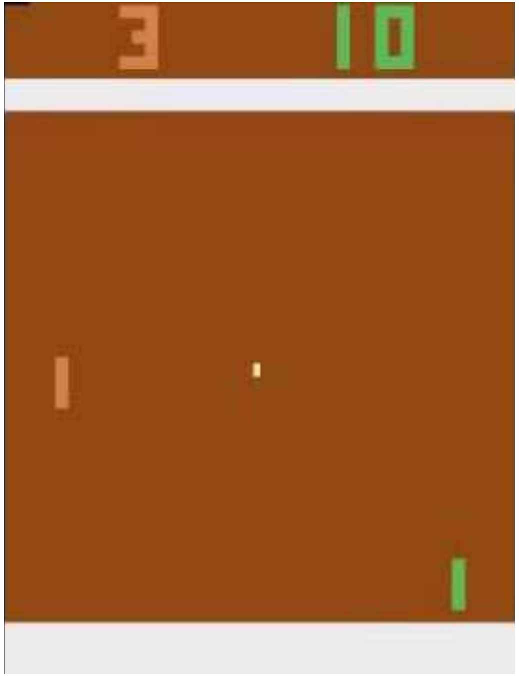
Verifiable Reinforcement Learning via Policy Extraction

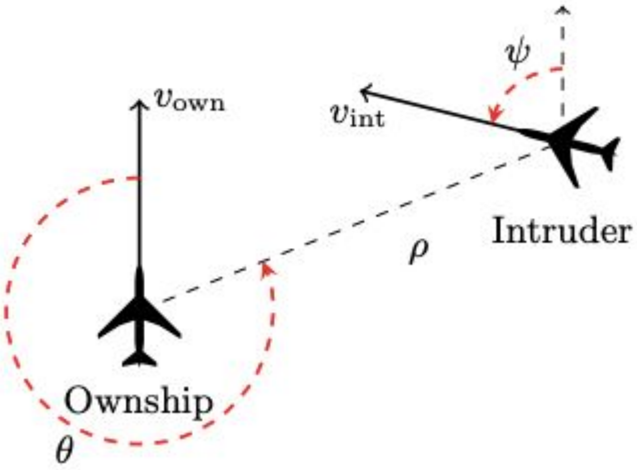
Osbert Bastani, Yewen Pu, Armando Solar-Lezama



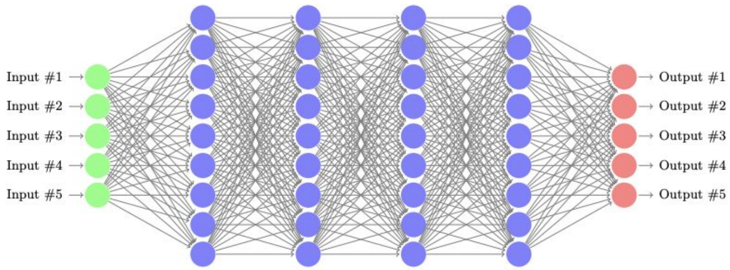
Verifiable Reinforcement Learning via Policy Extraction

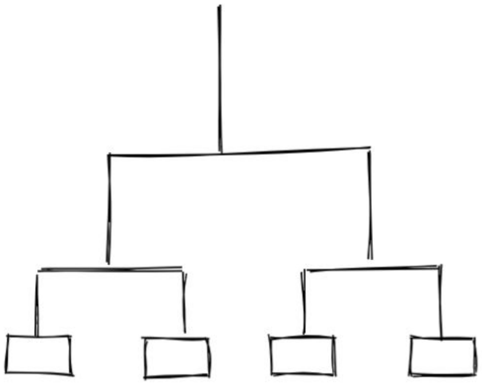
Claim: Verify that our RL agent is safe!





Katz et al. (2017)





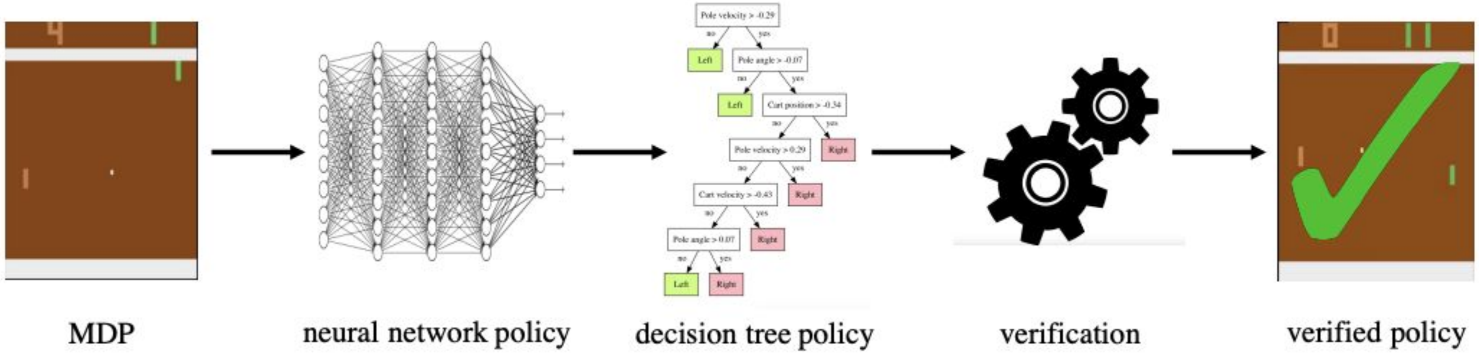
DNN agent:

- Easy to train
- Hard to verify

Tree agent:

- Hard to train
- Easy to verify

How to get a verifiable RL policy?

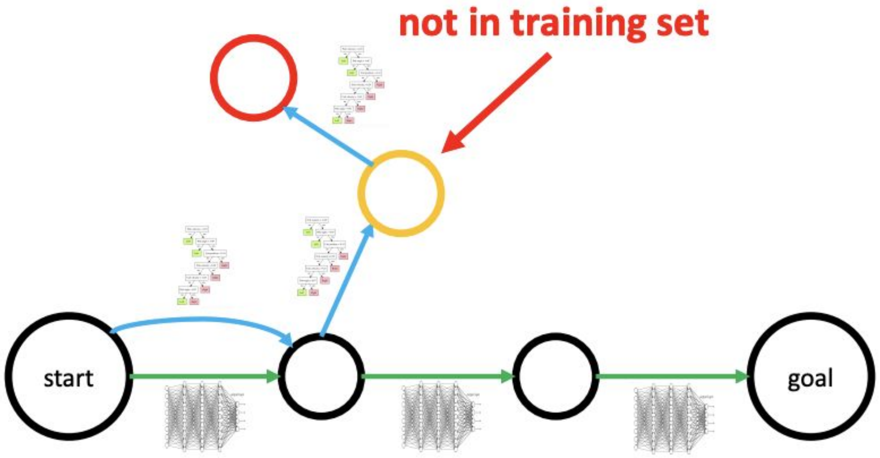


Bastani et al. (2019)

Structure of The Talk

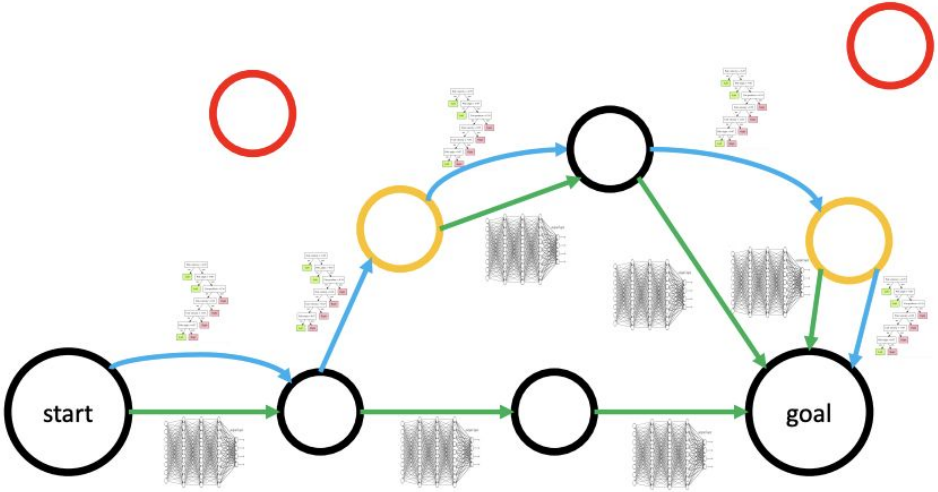
1. Policy Extraction
2. Policy Verification
    a. Correctness
    b. Robustness
    c. Stability
3. Evaluation
4. Discussion

Policy Extraction via Imitation Learning



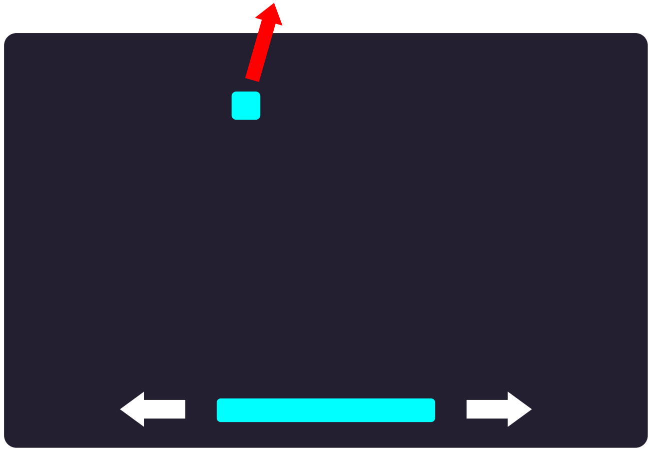
Ross and Bagnell (2011)

Policy Extraction via 🗡️ DAgger



Ross and Bagnell (2011)

Idea: The imitator should focus on critical states



$$V_t^{(\pi^*)}(s) \approx \min_a Q_t^{(\pi^*)}(s,a)$$

$$V_t^{(\pi^*)}(s) \gg \min_a Q_t^{(\pi^*)}(s,a)$$

VIPER: Sampling

Verifiability via **I**terative **P**olicy **E**xt**R**action

Define this measure of “criticalness” of a state

$$\tilde{\ell}_t(s) = V_t^{(\pi^*)}(s) - \min_{a \in A} Q_t^{(\pi^*)}(s, a)$$

And use it to re-sample from our trace data:

$$(s, a) \sim p((s, a)) \propto \tilde{\ell}_t \mathbb{I}[(s, a) \in D]$$

🐍 VIPER: Algorithm

Verifiability via **I**terative **P**olicy **E**xt**R**action

Algorithm 1 Decision tree policy extraction.

procedure VIPER($(S, A, P, R), \pi^*, Q^*, M, N$)
  Initialize dataset $\mathcal{D} \leftarrow \varnothing$
  Initialize policy $\hat{\pi}_0 \leftarrow \pi^*$
  for $i = 1$ **to** N **do**
    Sample M trajectories $\mathcal{D}_i \leftarrow \{(s, \pi^*(s)) \sim d^{(\hat{\pi}_{i-1})}\}$
    Aggregate dataset $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_i$
    Resample dataset $\mathcal{D}' \leftarrow \{(s, a) \sim p((s, a)) \propto \tilde{\ell}(s)\mathbb{I}[(s, a) \in \mathcal{D}]\}$
    Train decision tree $\hat{\pi}_i \leftarrow \text{TrainDecisionTree}(\mathcal{D}')$
  end for
  return Best policy $\hat{\pi} \in \{\hat{\pi}_1, ..., \hat{\pi}_N\}$ on cross validation
end procedure

Bastani et al. (2019)

🐍 VIPER: Theoretical guarantees

Theorem 2.2. *For any $\delta > 0$, there exists a policy $\hat{\pi} \in \{\hat{\pi}_1, ..., \hat{\pi}_N\}$ such that*

$$J(\hat{\pi}) \leq J(\pi^*) + T\varepsilon_N + \tilde{O}(1)$$

with probability at least $1 - \delta$, as long as $N = \tilde{\Theta}(\ell_{max}^2 T^2 \log(1/\delta))$.

$$\tilde{\ell}_t(s, \pi) = \tilde{\ell}_t(s)\tilde{g}(s, \pi)$$

Implies that we can achieve the same training loss via re-sampling:

$$\mathbb{E}_{(s,a)\sim p((s,a))}[\tilde{g}(s, \pi)] = \mathbb{E}_{(s,a)\sim \mathcal{D}}[\tilde{\ell}(s, \pi)]$$

1. Policy Extraction
2. **Verifying the Decision Tree Policy**
    a. Correctness
    b. Robustness
    c. Stability
3. Evaluation
4. Discussion

Correctness for Toy Pong

$$f_\pi(s) = f_i(s) = \beta_i^T s$$

$$\psi = \left(\bigwedge_{t=1}^{t_{\max}} \phi_t \right) \wedge \psi_0 \Rightarrow \bigvee_{t=1}^{t_{\max}} \psi_t$$

$\psi_t = (s_t \in Y_0)$

ϕ_t : Inductive controller invariant

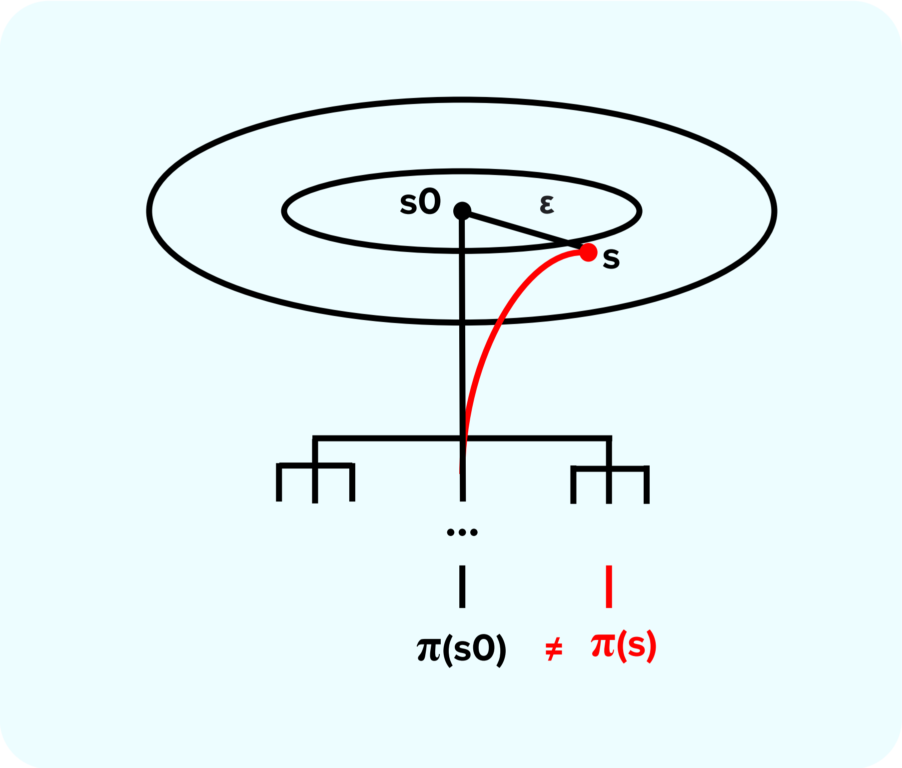
Controller is correct when $\neg\psi$ cannot be satisfied

Correctness for Toy Pong

- 30 Decision tree nodes vs 700 NN neurons
- SMT solved in <3 seconds
- Finds policy error!

Bastani et al (2019)

Robustness for Toy Pong



VIPER:

- Completes in seconds
- Accurate to ε within 10-5

Reluplex:

- Huge variance of completion times
- One timeout even
- Accurate to ε within 0.1

Stability for cartpole

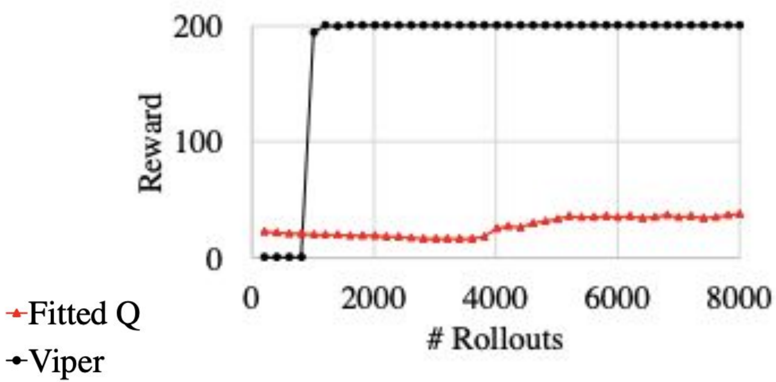
- Uses an iLQR oracle
- Achieves perfect reward on Cartpole
- Three node tree with linear regressors

Evaluation:

- VIPER is verified at stability region with Linf norm ≤ 0.03 in 4 seconds
- NN requires enumeration which takes 10 min. and verifies area 10^-15 of stability region

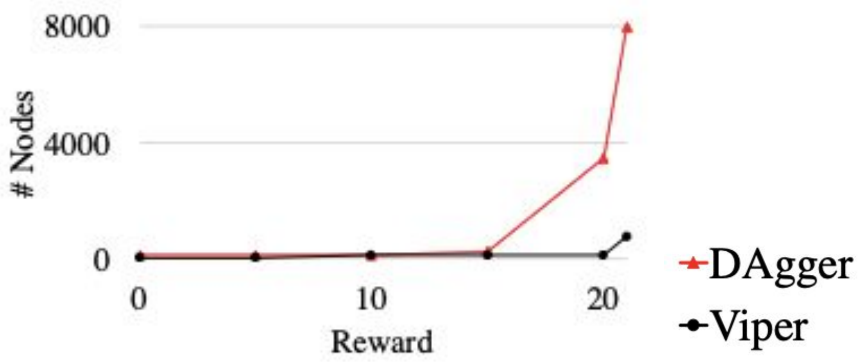
Comparing VIPER to other methods

Cartpole



Vs FittedQ

Toy Pong



DAgger

Discussion

- Policy Extraction also useful for explainable AI.
- Extracted policies need manual fixing.
- Verification process requires many approximations.
- What is the limit to decision tree extraction?

Things to take away if nothing else

- You can efficiently distill a trained DNN agent into a decision tree and have theoretical upper bounds on its training reward.
- The key idea of VIPER is sampling the Oracle in such a way that critical states are given more important weight (Q_opt >> Q_worst).
- Using a decision tree policy you can efficiently verify attributes such as correctness, stability, robustness.

Sources

- Bastani et al (2019)
- Ross et Bagnell (2011)
- https://trustml.github.io/docs/viper-presentation.pdf
- Katz et al. (2017)

Backup slides

How do you obtain the loss for continuous actions, i.e. when you cannot find Qmin?

Instead, we used an approach inspired by guided policy search [21]. We trained another decision tree using a different oracle, namely, an iterative linear quadratic regulator (iLQR), which comes with stability guarantees (at least with respect to the linear approximation of the dynamics, which are a very good near the origin). Note that we require a model to use an iLQR oracle, but we anyway need the true model to verify stability. We use iLQR with a time horizon of $T = 50$ steps and $n = 3$ iterations. To extract a policy, we use $Q(s, a) = -J_T(s)$, where $J_T(s) = s^T P_T s$ is the cost-to-go for the final iLQR step. Because iLQR can be slow, we compute the LQR controller for the linear approximation of the dynamics around the origin, and use it when $\|s\|_\infty \leq 0.05$. We now use continuous actions $A = [-a_{\text{max}}, a_{\text{max}}]$, so we extract a (3 node) decision tree policy π with linear regressors at the leaves (internal branches are axis-aligned); π achieves a reward of 200.0.

Problem formulation

Problem formulation. Let (S, A, P, R) be a finite-horizon (T-step) MDP with states S, actions A, transition probabilities $P : S \times A \times S \to [0, 1]$ (i.e., $P(s, a, s') = p(s' \mid s, a)$), and rewards $R : S \to \mathbb{R}$. Given a policy $\pi : S \to A$, for $t \in \{0, ..., T-1\}$, let

$$V_t^{(\pi)}(s) = R(s) + \sum_{s' \in S} P(s, \pi(s), s') V_{t+1}^{(\pi)}(s')$$

$$Q_t^{(\pi)}(s, a) = R(s) + \sum_{s' \in S} P(s, a, s') V_{t+1}^{(\pi)}(s')$$

be its value function and Q-function for $t \in \{0, ..., T-1\}$, where $V_T^{(\pi)}(s) = 0$. Without loss of generality, we assume that there is a single initial state $s_0 \in S$. Then, let

$$\begin{aligned} d_0^{(\pi)}(s) &= \mathbb{I}[s = s_0] \\ d_t^{(\pi)}(s) &= \sum_{s' \in S} P(s', \pi(s'), s) d_{t-1}^{(\pi)}(s') \qquad (\text{for } t > 0) \end{aligned}$$

be the distribution over states at time t, where $\mathbb{I}$ is the indicator function, and let $d^{(\pi)}(s) = T^{-1} \sum_{t=0}^{T-1} d_t^{(\pi)}(s)$. Let $J(\pi) = -V_0^{(\pi)}(s_0)$ be the cost-to-go of π from s_0. Our goal is to learn the best policy in a given class Π, leveraging an *oracle* $\pi^* : S \to A$ and its Q-function $Q_t^{(\pi^*)}(s, a)$.

Reward bound

Theorem 2.2. *For any $\delta > 0$, there exists a policy $\hat{\pi} \in \{\hat{\pi}_1, ..., \hat{\pi}_N\}$ such that*

$$J(\hat{\pi}) \leq J(\pi^*) + T\varepsilon_N + \tilde{O}(1)$$

with probability at least $1 - \delta$, as long as $N = \tilde{\Theta}(\ell_{max}^2 T^2 \log(1/\delta))$.

In contrast, the bound $J(\hat{\pi}) \leq J(\pi^*) + uT\varepsilon_N + \tilde{O}(1)$ in [25] includes the value u that upper bounds $Q_t^{(\pi^*)}(s, a) - Q_t^{(\pi^*)}(s, \pi^*(s))$ for all $a \in A$, $s \in S$, and $t \in \{0, ..., T-1\}$. In general, u may be $O(T)$, e.g., if there are *critical states* s such that failing to take the action $\pi^*(s)$ in s results in forfeiting all subsequent rewards. For example, in cart-pole [5], we may consider the system to have failed if the pole hit the ground; in this case, all future reward is forfeited, so $u = O(T)$.

An analog of u appears implicitly in ε_N, since our loss $\tilde{\ell}_t(s, \pi)$ includes an extra multiplicative factor $\tilde{\ell}_t(s) = V_t^{(\pi^*)}(s) - \min_{a \in A} Q_t^{(\pi^*)}(s, a)$. However, our bound is $O(T)$ as long as $\hat{\pi}$ achieves high accuracy on critical states, whereas the bound in [25] is $O(T^2)$ regardless of how well $\hat{\pi}$ performs.

Correctness for Toy Pong

We partition the state so that for every partition S_i we can get a Beta_i from the policy.

$$f_\pi(s) = f_i(s) = \beta_i^T s$$

Either we were not in this state previously or our current state is a result from the dynamics

$$\phi_t = \bigvee_{i=1}^{k} \left(s_{t-1} \in S_i \Rightarrow s_t = \beta_i^T s_{t-1} \right) \quad \forall t \in \{1, \ldots, t_{\max}\}$$

$$\psi = \left(\bigwedge_{t=1}^{t_{\max}} \phi_t \right) \wedge \psi_0 \Rightarrow \bigvee_{t=1}^{t_{\max}} \psi_t$$