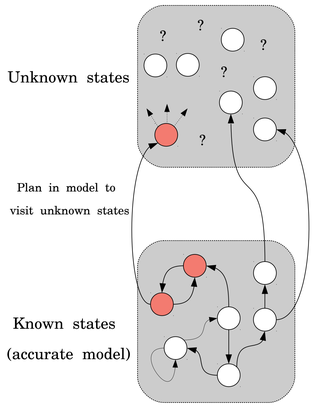
Explicit Explore-Exploit Algorithms in Continuous State Spaces

Mikael Henaff

Microsoft Research NYC

Introduction

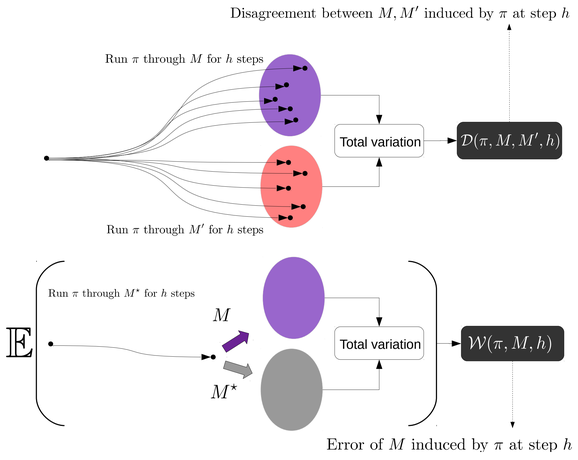
- Many RL algorithms have high sample complexity
- Explicit Explore-Exploit (E^3): first provably sample-efficient RL algorithm
- Sample complexity: $\texttt{poly}(|\mathcal{S}|, |\mathcal{A}|, H)$



- Assumes small discrete $\mathcal{S}$
- **Goal: handle large/infinite $\mathcal{S}$**

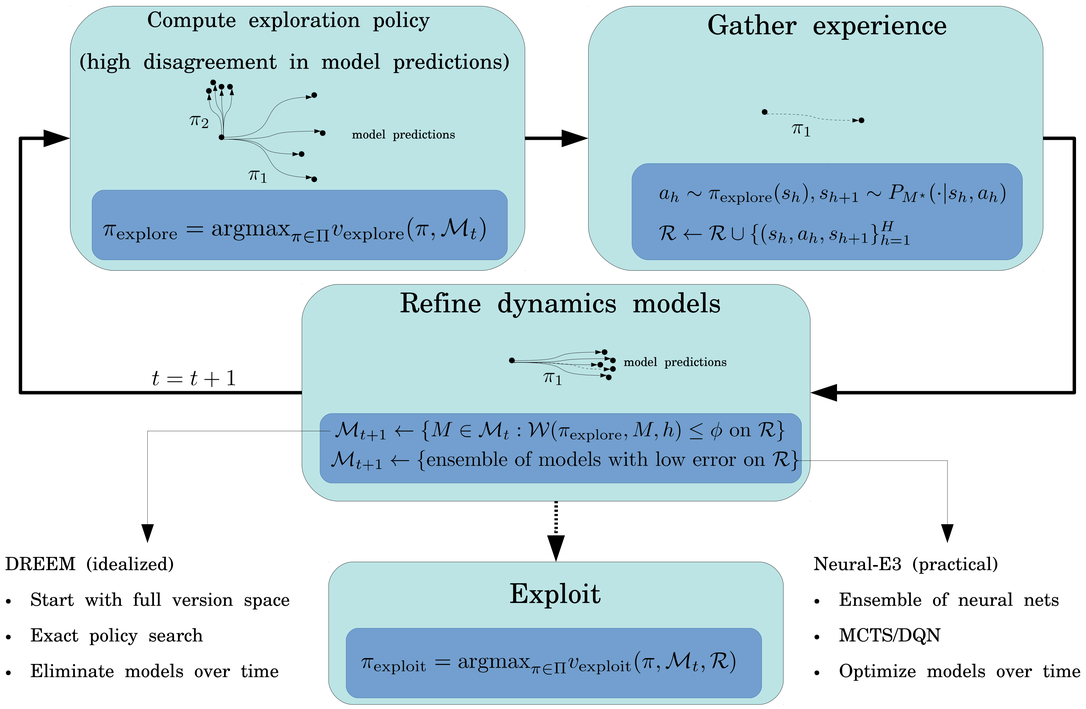
Setup

Model class $\mathcal{M}$, $M : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$
Policy class Π, horizon H



$v_{\text{explore}}(\pi, \mathcal{M}) = \max_{M, M' \in \mathcal{M}} \sum_{h=1}^{H} \mathcal{D}(\pi, M, M', h)$
$v_{\text{exploit}}(\pi, M) = \sum_{h=1}^{H} \sum_{s_h} P_M^{\pi,h}(s_h) R^\star(s_h)$

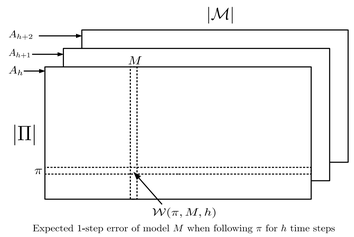
Algorithm



Theorem

Assume that $M^\star \in \mathcal{M}$. With probability at least $1 - \delta$, DREEM outputs an ϵ-optimal exploitation policy after collecting at most $\tilde{O}\left(\frac{H^5 d^2 |\mathcal{A}|^4}{\epsilon^2} \log\left(\frac{T|\mathcal{M}||\Pi|}{\delta}\right)\right)$ samples, where d is the max rank of the misfit matrices.

Use ranks of **model misfit matrices** as complexity measure: $d = \max_h \text{rank}(A_h)$

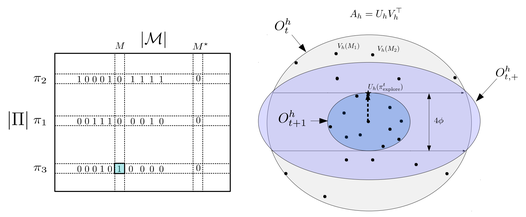


Expected 1-step error of model M when following π for h time steps

- Bounded by $|\mathcal{S}|$
- Bounded by rank of transition matrix
- Bounded by # parameters in factored MDPs

Proof sketch (simplified, errors are $0/1$):

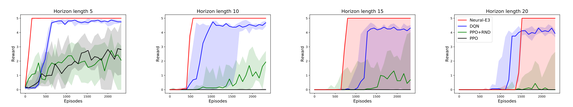
- High disagreement between $M, M' \implies$ at least one must have high error
- At iteration t, there is a model in $\mathcal{M}_t$ with high error or all models give a good exploitation policy
- Row π_t of A_h is linearly independent of rows of previous $\pi_t \implies$ at most $\text{rank}(A_h) \leq d$ iterations

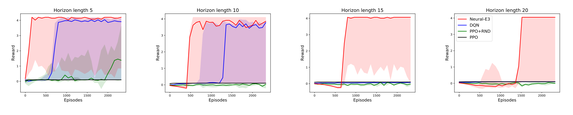


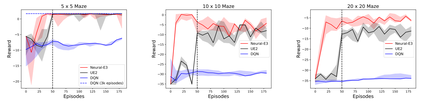
Experiments

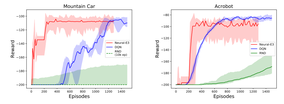
- Small ensemble of NN models to approximate version space (4-8 models)
- MCTS/BFS for planning during exploration
- DQN on replay buffer for exploitation


(a) Stochastic Combination Lock (b) Mazes (c) Classic Control


(d) Stochastic Combination Lock


(e) Stochastic Combination Lock (antishaped rewards)


(f) Maze


(g) Classic Control

Related Work

E^3: Kearns and Singh, 2002
Error matrices: Jiang et al, 2017; Sun et al, 2019
Practical algorithm: Shyam et al, Pathak et al 2019

Links

- Paper: `https:/arxiv.org/abs/1911.00617`
- Code: `https://github.com/mbhenaff/neural-e3`
- Contact: `mihenaff@microsoft.com`