EXPLORATION POLICY/BONUS NOTES

NIKO YASUI 1452815

Questions

(1) ϕ-exploration-bonus uses a pseudocount that is a lower bound of the Hamming Similarity. What's the point of trying to minimize a lower bound?

Reward-Bonus Exploration

Şimşek, Ö., & Barto, A. G. (2006). An intrinsic reward mechanism for efficient exploration. Proceedings of the 23rd International Conference on Machine Learning

The goal is to explore to develop an optimal behavior policy π_{s_i} that will be executed at some later time.

Approach

- Uses state with two parts: s_e describes the external state of the environment (the traditional "state"); s_i, the internal state, describes π_{s_i} and other information that may cause it to change in the future.
- Intrinsic reward is defined as $r_i(t) = p + \sum_{s \in \mathcal{S}} max_{T \leq t} V_T(s) - max_{T \leq t} V_T(s)$, where the max is used for smoothing; a moving average is also suggested.
- Pessimistic initialization is important because the approach relies on increasing the value estimates over time.

Empirical Results

- Learned a good policy much more quickly and with less RMS error than baseline action selection methods in a maze task.
- Learned a good option policy after a shorter exploration period than baseline methods. Unlike baseline methods, performance was at least as good as an intra-option Q-learning agent that only used primitive actions.

Singh, S., Barto, A. G., & Chentanez, N. (2004). Intrinsically motivated reinforcement learning. 18th Annual Conference on Neural Information Processing Systems (NIPS), 17, 1281âĂŞ1288.

Creates options based on pre-defined salient events, and builds option transition and reward models. Upon reaching a state with a salient event e, the probability of that state being reached under the current option o_e is used to create an internal reward.

Using option models to determine intrinsic reward significantly speeds up learning compared with no intrinsic reward.

Brafman, R. I., & Tennenholtz, M. (2003). R-max - a general polynomial time algorithm for near-optimal reinforcement learning.

Simple algorithm that attains near-optimal average reward in polynomial time. Builds a model, which is initialized with all states returning the maximum possible reward.

Main Idea

- Following the optimal policy with respect to the optimistically initialized model always behaves optimally or leads to efficient learning.

Theoretical Results

- R-max produces near-optimal average reward in polynomial time.
- The polynomial guarantee will probably not be practical.

Strehl, A. L., & Littman, M. L. (2008). An analysis of model-based Interval Estimation for Markov Decision Processes.

Introduces Model-Based Interval Estimation (MBIE), and shows that it is PAC for finite MDPs. Also proposes a simpler algorithm (MBIE-EB), which has the same theoretical bounds as MBIE.

Proposition

- PAC-MDP means that the algorithm executes a near-optimal strategy most of the time. It can only be suboptimal for a small polynomial number of time-steps.
- Instantaneous Loss at time t is $il(t) = V^*(s_t) - sum_{i=t}^{H}\gamma^{i-t}r_i$, where H is the total number of time steps in the trial.
- Average loss is $\frac{1}{H}\sum_{t=1}^{H} il(t)$.
- An algorithm is PAC-MDP in the average loss setting if for any ϵ, δ, we can choose H and a polynomial such that the average loss of the agent on a trial of H steps is guaranteed to be less than ϵ with probability at least $1-\delta$.
- There is a model size limit m. After m experiences of (s, a), no future data is considered for (s, a).
- Action-values are initialized to $1/(1-\gamma)$.
- MBIE builds the upper tail of the CI on V^* of M by considering a confidence interval on the space of MDPs.
- MBIE-EB's confidence intervals are similar to those of Action Elimination. It uses the equation $\tilde{Q}(s, a) = \hat{R}(s, a) + \gamma \sum_{s'} \hat{T}(s' \mid s, a) max_{a'} \tilde{Q}(s', a') + \frac{\beta}{\sqrt{n(s,a)}}$.

Theoretical Results

- By building CIs for the reward and transition probabilities, the authors define a Bellman equation for Q that maximizes over the CIs. They show that their proposed update leads to a contraction mapping in the max-norm.
- The polynomial bound on time-steps that are not PAC is proved.
- "Optimism in the face of uncertainty": supposing the confidence intervals computed contain the mean, then $\tilde{Q}(s, a) \geq Q^*(s, a)$ for every iteration of MBIE and MBIE-EB.
- The bounds on sample complexity for some MDPs are better for MBIE than R-max, but R-max is more computationally efficient.
- A PAC algorithm with respect to sample complexity is also PAC with respect to average loss.

Empirical Results

- Both MBIE variants outperformed R-Max and E-3 significantly on River-Swim and SixArms with respect to cumulative reward, but performance was only "clinically significant" for SixArms.

Martin, J., Sasikumar, S. N., Everitt, T., & Hutter, M. (2017). Count-Based Exploration in Feature Space for Reinforcement Learning. Introduces a ϕ-Exploration-Bonus algorithm, which uses a feature-based density model for generalized visit-counts. The joint feature visit-density is a product of independent distributions for each feature. Proposed estimators for these independent distributions all assume discreteness. The naive ϕ-pseudocount is a lower bound on the Hamming Similarity, but the ϕ-pseudocount is used in the algorithm.

Proposition

- The joint feature visit density $\rho_t(\phi) = \prod_{i=1}^{m} \rho_t^i(\phi_i)$ is a product of independent feature-wise visit densities.
- Naive ϕ-pseudocount: $\tilde{N}_t^{\phi}(s) = t\rho_t(\phi(s))$
- ϕ-pseudocount: $\hat{N}_t^{\phi}(s) = \frac{\rho_t(\phi)(1-\rho_t'(\phi)))}{\rho_t'(\phi)-\rho_t(\phi)}$
- Augment reward with $R_t = \frac{\beta}{\hat{N}_t^{\phi}(s)}$

Theoretical Results

- If $\rho_t^i(\phi_I) = \frac{1}{t}N_t(\phi_i)$, then for binary features, $\rho_t^i(\phi_i) = \frac{1}{t}\sum_{k=1}^{t} 1 - |\phi_i - \phi_{i,k}|$
- For binary features, $\rho_t(\phi) \leq \frac{1}{t}\sum_{k=1}^{t} \mathrm{Sim}(\phi, \phi_k)$, where Sim is the Hamming Similarity.
- Lower bound: $\tilde{N}_t^{\phi}(s) \leq \sum_{k=1}^{t} \mathrm{Sim}(\phi, \phi_k)$

Empirical Results Tests were conducted in the ALE on sparse reward games (Montezuma's Revenge, Venture, Freeway), and dense reward games (Frostbite, Q*bert). The reward bonus tested using SARSA(λ). Baseline was ϵ-greedy SARSA with unknown ϵ. Parameter $\beta = 0.05$ was chosen once on a rough sweep for all games.

- Sarsa-ϕ-EB outperformed Sarsa-ϵ on all games except Freeway, where the agent is content to watch the cars go across the screen. Setting β to 0.035 corrected this issue.
- Sarsa-ϕ-EB is second best in Montezuma after Double DQN with Pseudocount, and trains for half the frames.
- Sarsa-ϕ-EB is competitive with state-of-the-art deep network methods.

Value-Bonus Exploration

Gehring, C., & Precup, D. Smart exploration in reinforcement learning using absolute temporal difference errors. Proposes a controlability function that approximates the predictability of action effects that is linear in the number of features. $C^{\pi}(s,a) = -boldsymbol E_{\pi}[|\delta_t| \mid s_t = s, a_t = a]$. Pick actions greedily wrt $Q(s_t, a_t) + \omega C(s_t, a_t)$. Update using $\boldsymbol{w}_{a_t} \leftarrow \boldsymbol{w}_{a_t} - \alpha\prime(|\delta_t| + \boldsymbol{w}_{a_t}^{\top}\phi_{s_t})\phi_{s_t}$.

Algorithm Properties

- Controlability is a bonus, so the traditional value function is always available.
- By sticking to controlable regions, the algorithm avoids high-variance regions of the state space.

Theoretical Results

- Huber [3] showed that mean abs devation is a lower bound on standard deviation.
- The given updates, with function approximation, converge to optimal values using Theorem 2.2 of [1]. This requires a slowly, smoothly changing policy and annealing ω, but is "not practically useful".

Emperical Results

- Speeds up learning and decreases standard deviation of return in 18x18 gridworld, but in the given plot both methods look similar.
- In function approximation case, controlability can push the agent towards more observable states.
- Improves performance in the helicopter task for parameters optimized for $\omega = 0$. The plot suggests that annealing ω could be very effective.

Szita, I., & Lorincz, A. (2008). The many faces of optimism.

This paper presents an algorithm that combines ideas from several exploration papers to create the Optimistic Initial Model (OIM) algorithm.

Brief survey of exploration methods:

- e-greedy and boltzmann converge to optimality but time may scale exponentially in the number of states
- optimistic initial values (OIV) converge to near-optimality but might take a long time if initial values are too high
- bayesian methods are computationally expensive and cannot calculate the optimal policy exactly
- confidence interval estimation assumes state values are drawn from a distribution, and choose the action with the highest upper confidence bound. Polynomial time convergence bounds exist for one algorithm, and another has logarithmic regret in the number of steps taken.
- bonus methods:
  - could explore by adding a bonus b to the reward as $r+\kappa*b(x_t, a_t, x_{t+1}$. Since the bonus can change quickly, the algorithm must be able to "spread the changes effectively". κ must also be annealed over time.
  - could learn two Q functions; Q^r based on reward, and Q^e based on exploration, with $Q_t = Q_t^r + \kappa Q_t^e$. Then, if we set κ to 0, we immediately have the reward based Q^r, which may converge even if Q^e does not.
- E^3 and R-max are the first algorithms with poly-time bounds to finding near-optimal policies. R-max keeps a model of the environment that assumes all actions in all states lead to a hypothetical max-reward absorbing state. The model is updated each time a "high-precision" estimate of transition-reward probabilities is known.

The algorithm:

- Greedy action selection
- Uses two Q values: $Q(x, a) = Q^r(x, a) + Q^e(x, a)$.
- Absorbing "garden of Eden" state x_E at which the agent recieves $R_{\max}$ reward at each timestep.
- Use sample averages to approximate probability of a transition and the expected reward of a transition. Exploration rewards are $R_{\max}$ inside x_E and 0 otherwise.
- The initial model assumes x_E has been reached once in each state-action pair, so that $Q_0(x, a) = \frac{R\max}{1-\gamma} = V_{\max}$

- Value functions are updated using a sum over all states for each step using the following equations:

$$Q^r_{t+1}(x,a) \coloneqq \sum_{y\in X} \hat{P}_t(x,a,y)\left(\hat{R}_t(x,a,y) + \gamma Q^r_t(y,a_y)\right)$$

$$Q^e_{t+1}(x,ya) \coloneqq \gamma \sum_{y\in X} \hat{P}_t(x,a,y) Q^e_t(y,a_y) + \hat{P}_t(x,a,y) V_{\max}.$$

- Can be used online in a neighborhood of the agent's current state. Authors use improved prioritized sweeping.

Relationship to other methods:

- model based extension of OIV. DP updates do not lower the exploration boost, but model updates (experiencing more transitions) do.
- R-max updates the model after a transition's estimates are precise. OIM updates the model after each transition as soon as it is available – bootstrapping?
- The state x_E can be seen as implementing an exploration bonus $b_t(x,a) = \frac{1}{N_t(x,a)}\left(V_{\max} - Q_t(x,a)\right)$, where $N_t(x,a)$ is the number of times action a was selected in state x before time t.
- Proof of poly-time convergence is similar to model-based interval exploration.

Experiments:

- RiverSwim and SixArms: Selected optimal parameters for existing agents and a rough sweep for $R_{\max}$ in OIM. Used value iteration rather than prioritized sweeping. Point estimates are superior, but 95% CIs overlap with MBIE.
- MazeWithSubgoals: OIM learns near-optimal policies much faster than e-greedy, bonus based, and MBIE methods.
- Chain: Higher accumulated reward than competition.
- Loop: Higher accumulated reward than competition.
- FlagMaze: Not as good as Bayesian DP, which was given the list of successor states. Plateaued around the same cumulative reward as e-greedy, but twice as fast. About 60% of optimal.

Unprocessed Papers

Understanding least-squares methods for control in reinforcement learning

- Least-squares methods are sample-efficient
- Least-squares methods suffer from forgetting AND have outdated samples?
- Variability comes from partial observability rather than noise in the reward distributions

White, M., & White, A. (2010). Interval estimation for reinforcement-learning algorithms in continuous-state domains. Robust confidences for continuous MDPs. Computes CIs online under a changing policy.

Auer, P. (2003). Using confidence bounds for exploitation–exploration trade-offs. Proves regret bounds based on using an upper-confidence bound tradeoff.

Grande, R., Walsh, T., & How, J. (2014). Sample Efficient Reinforcement Learning with Gaussian Processes. Introduces DGPQ, an GP algorithm that is sample efficient for model-free RL.

Kakade, S., Kearns, M., & Langford, J. (2003). Exploration in metric state spaces. Introduces metric-E3, which finds a near-optimal policy using locally accurate models when there is a metric on the state space.

Osband, I., Van Roy, B., & Wen, Z. (2016). Generalization and Exploration via Randomized Value Functions. Introduces RLSVI; efficient exploration and effective generalization using randomized least squares value iteration.

Russo, D., & Van Roy, B. (2013). Eluder Dimension and the Sample Complexity of Optimistic Exploration. Regret bound for UCB and posterier sampling algorithms that measure the degree of dependence for linear rewards.

Strehl, A. L., Li, L., Wiewiora, E., Langford, J., & Littman, M. L. (2006). PAC model-free reinforcement learning. Introduces Delayed Q-Learning; Model-free efficient PAC RL.