Paper summary: Never give up: learning directed exploration strategies

March 23, 2022

1 Idea in few sentances

2 Explanation of the central concept

3 Methodology

4 Initial rambly notes

4.1 Abstract

A RL designed to solve hard exploration problems is proposed. The idea is to learn a range of directed exploratory policies. An episodic memory-based intrinsic reward is constructed using k-nearest neighbors over the agent's recent experience. That reward is used to train the directed exploratory policies, thereby ecouraging the agent to repeatedly revisit all states in its environment. A self-supervised inverse dynamics model is used to train the embedding of the nearest neighbor lookup so that the novelty is biased toward what the agent can control. Universal value function approximators (UVFA) are used to simultaneously learn the different directed exploratory policies within the same neural network. Using the same network enables transfer between predominantly exploratory policies to effective exploitative policies. Works in parallel environments, achieves good results on *Pitfall!*.

4.2 Introduction

To have effective exploration, you want to ensure you reach all state-action pairs infinitelly often. While ϵ-greedy or Boltzmann exploration works, it's inefficient and steps needed grow exponentially with state space size. Still they work well in dense reward settings. Novelty based exploration achieved nice results, but fails once whatever measure of novelty has been satisfied (it does not go further downstream). Using prediction error from predictive forward models is expensive and error-prone.

The main idea of the proposed approach is to jointly learn separate exploration and exploitation policies derived from the same network so that we get the best of each. In fact, a family of policies with various degrees of exploratory behavior is learned.

An intrinsic reward that combines episodic and life-long novelty is proposed. The episodic novelty encourages periodic revisits of familiar states over several episodes (but not the same episode). Every observation potentially changes the per-episode novelty significantly. Life-long novelty downmodulates states that become familiar over many episodes. It is driven by random network distillation error and it is slow.

4.2.1 Short note on UFVA

The universal in universal function approximators is that they generalize not only over states $V(s;\theta)$, but also over goals $V(s,g;\theta)$. There are "general value functions" that tell you about progressions in an environment. The paper which introduced this used a special network architecture where the goals and states produce different embeddings and can be trained in 2 ways. Check out the paper again if needed.

4.2.2 Contributions

1. defining an exploration bonus which includes life-long and episodic novelty which is able to maintain exploration throughout the training process (*never give*)
2. learning a family of policies that separate exploration and exploitation using a conditional architecture with shared weights
3. experimental evidence

4.3 Method

TODO: crop their fantastic image showing their architecture and put it here. The introduced intrinsic reward r_i^t satisfies 3 properties:

1. it rapidly discourages revisiting the same state within the same episode
2. it slowly discourages visits to states visited many times across episodes
3. the notion of a state ignores aspects of the environment that are not influenced by the agent's action

It is composed of 2 blocks: an *episodic novelty module* and an *life-long novelty module*. The episodic novelty model consists of an episodic memory M and an embedding function f which maps from the current observation to a learned representation that is refered to as a *controllable states*. At every timestep r_t^{episodic} is computed, the controllable state is appended to the current obsevation to memory M.

The life-long module modulates the exploration bonus r_t^{episodic} with a life-long curiosity factor α_t. This vanishes over time. The formula is:

$$r_t^i = r_t^{\text{episodic}} \cdot \min\left\{\max\left\{\alpha_t, 1\right\}, L\right\} \tag{1}$$

where L is a chosen maximum reward scaling.

Embedding network $f : \mathcal{O} \to \mathbb{R}^p$ maps the current observation to a p-dimensional vector corresponding to its controllable state. To avoid meaningless exploration, given two consecutive observations a Siamese network f is trained to predict the action taken by the agent to go from one obsevation to the next. More formally, given $\{x_t, a_t, x_{t+1}\}$ where x_t are observations taken by the agent a_t, a conditional likelihood is parametrized:

$$p(a|x_t, x_{t+1}) = h(f(x_t), f(x_{t+1})) \tag{2}$$

where h is one hidden layer MLP followed by a softmax. Parameters of both h and f are trained by maximum likelihood.

The intrinsic reward definition is inspired by state-counting:

$$r_t^{\text{episodic}} = \frac{1}{\sqrt{n(f(x_t))}} \approx \frac{1}{\sqrt{\sum_{f_i \in N_k} K(f(x_t), f_i)} + c} \tag{3}$$

where $n(f(x_t))$ are the counts for the visits to the abstact state $f(x_t)$. These counts are approximate as the sum of similarities given by a kernel function $K : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ over the content of M. In practie, pseudo-counts are computed using the k-nearest neighbors of $f(x_t)$ in memory M, denoted by $N_k = \{f_i\}_{i=1}^k$ The constant c guarantees a minimum amount of pseudo-counts.

$$K(x, y) = \frac{\epsilon}{\frac{d^2(x,y)}{d_m^2} + \epsilon} \tag{4}$$

where $\epsilon = 10^{-3}$ is used in all experiments.

4.3.1 Integrating life-long curiosity

Could be any long-term novelty estimator, but RND is used. I've done that paper already, check details there and in this paper for implementation.

4.3.2 The never give up agent

The basis is recurrent replay distributed DQN (R2D2). chekc that paper later.

4.4 Other stuff