paper_review__I2A

February 26, 2019

1 Abstract

I2A: architecture combining model-free and model-based aspects

Existing model-based RL: prescribe how a model should be used to arrive at a policy.

- For typical model-based RL, rules for update policy or value using output of Env-Model are defined. For example, Dyna-Q uses below update rule.

$$R, S' \leftarrow Model(S, A) \tag{1}$$
$$Q(S, A) \leftarrow Q(S, A) + \alpha[R + r \max_a Q(S', a) - Q(S, A)] \tag{2}$$

I2A: learns to interpret predictions from a learned environment model to construct implicit plans in arbitrary ways.

- I2A use encoded env-model's output as additional context to train a policy or value network, so we don't know clearly how predicted trajectories affect policy or value.

I2As show improved data efficiency, performance, and robustness to model misspecification compared to several baselines.

2 1. Introduction

Model-free RL usually requires large amount of traning data and the resulting policies do not readily generalize to novel tasks.

Model-based RL aims to address these shortcomings, but suffers from model errors resulting from function approximation. These errors compound during planning, causing over-optimism and poor agent performance.

I2A shows robustness against model imperfections, which use approximate environment models by "learning to interpret" their imperfect predictions.

3 2. The I2A architecture

Imagination core

The environment model together with rollout poicy $\hat{\pi}$ constitute the imagination core module, which predicts next time step.

- $\hat{o}_{t+1}, \hat{r}_{t+1} = \text{IC}(o_t)$
- $\hat{a}_t = \text{PolicyNet}(o_t)$
- $\hat{o}_{t+1}, \hat{r}_{t+1} = \text{EnvModel}(o_t, \hat{a}_t)$
- $\hat{\pi}$: rollout policy, which is determined by PolicyNet

PolicyNet EnvModel ?

Single Imagination rollout

The imagination core is used to produce n trajectories $\hat{\mathcal{T}}_1, \ldots, \hat{\mathcal{T}}_n$. Each imagined trajectory $\hat{\mathcal{T}}$ is a sequence of features $(\hat{f}_{t+1}, \ldots, \hat{f}_{t+\tau})$, where t is the current time, τ the length of the rollout, and $\hat{f}_{t+i}$ the output of the env model.

$$\hat{f}_{t+i} = [\hat{o}_{t+1}, \hat{r}_{t+1}] = \text{IC}(\hat{o}_{t+i-1}) \tag{3}$$

$$\hat{\mathcal{T}}_i = (\hat{f}_{t+1}, \ldots, \hat{f}_{t+\tau}) \tag{4}$$

Each rollout $\hat{\mathcal{T}}_i$ is encoded as rollout embedding e_i, and then embeddings $e_1, \ldots, e_n$ are aggregated as c_{ia}

$$e_i = \mathcal{E}(\hat{\mathcal{T}}_i) c_{\text{ia}} = \mathcal{A}(e_1, \ldots, e_n)$$

Full I2A Architecture

The final componet of the I2A is the policy module, which is a network that takes the information c_{ia} from model-based predictions, as well as the output c_{mf} of a model-free path. The I2As learnings to combine information from its model-free and imagination-augmented path.

$$\pi, V = \text{FC}(c_{\text{ia}}, c_{\text{mf}})$$

4 3. Architectural choices and experimental setup

4.0.1 3.1 Roll strategy

For this experiments, we perform one rollout for each possible action in the environment. The first action in the i^{th} rollout is the i^{th} action of the action set $\mathcal{A}$, and subsequent actions for all rollouts are produced by a shared rollout policy $\hat{\pi}$.

Training rollout policy $\hat{\pi}$ - by adding to the total loss **a cross entropy auxiliary loss** between the imagination-augmented policy π and the policy $\hat{\pi}$, both for the current observation.

4.0.2 3.2 I2A components and environment models

In this experiments, the encoder is an LSTM with convolutional layers which sequentially processes a trajectory $\mathcal{T}$. The features $\hat{f}_t$ are fed to the LSTM in reverse order to mimic Bellman type backup operations. (but choice of forward, backward bi-directional seems to have little impact on the performance.)

Training environment model - pretrain and freeze (this led to faster runtime of the I2A architecture compared to training jointly) - jointly train with full network by adding l_{model} to the total loss.

pre-trained env. model

For all environments, training data for this environment model was generated from trajectories of a partially trained standard model-free agent.