Paper summary: Agent 57: Outperforming the human Atari benchmark

March 20, 2022

1 Idea in few sentances

2 Explanation of the central concept

3 Methodology

4 Initial rambly notes

4.1 Abstract

The paper proposes Agent57, the first deep RL agent to outperform the standard human benchmark on all 57 Atari games. It achieves this by training a neural network which parametrizes a family of policies ranging from very exploratory to purely exploitative. The authors propose an adaptive mechanism to choose which policy to prioritize throughout the training process. Additionally, a novel parametrization of the architecture allows for more consistent and stable training.

4.2 Introduction

The reasons we benchmark algorithms on the Atari 2600 games are:

- the games are varied enough to claim generality
- each game is interesting enough to be representative of a setting that may be faced in practise
- each game is created by an independent party and thus free of the experimenter's bias

Human baseline scores (HNS) are deemed to be "reasonable performances". DQN was the first to algorithm to achieve human level performace in a large number of games. MuZero, a state of the art model-based RL algorithm suprasses 100% HNS on 51 games, and R2D2, a state of the art model-free RL algorithm

surpasses 100% HNS on 52 games. However, when these algorithms fail to surpass HSN, they fail completely.

The games they fail in showcase the problem of *long-term credit assignment*, e.x. when the reward structures are peculiar (games such as *Skiing* and *Solaris*) which makes it unclear which decisions are most deserving of positive or negative credit for the outcomes that follow. The second problem is efficient *exploration* (games like *Private Eye, Montezuma's Reverge, Pitfall!, Venture*), i.e. when the rewards are extra sparse. In this case the agent needs to explore without seeing any extrinsic reward.

There are 3 types of exploration algorithms in deep RL:

- randomized value functions
- unsupervised policy learning
- intrinsic motivation

An algorithm which recently showed significat progress on hard exploration games is Never Given Up (NGU). It achieves this by augmenting the reward signal with an internally generated intrinsic reward that is sensitive to novelty on 2 levels: the short-term novelty within an episode and long-term novelty across episodes. It then learns a family of policies for exploring and exploiting (sharing the same parameters), with the end goal of obtaining the highest score on the exploitative policy. However, NGU performs strongly on a few games (those on which MuZero and R2D2 perform poorly). The problem is that some games require significantly different amounts of exploration.

This method augments NGU to adapt its exploration strategy to the particular game it's playing. Furthermore, authors tackle the long-term credit assignment problem by improving the overall training stability, dynamicaly adjusting the discount factor and increasing the backprop though time window (what?). The hyperparameters are adjusted dynamically in the following way. A simple non-stationary multi-armed bandit directly controls the exploration rate and the discount factor to maximize the episode return and then provides this information to the value network of the agent as input. The bandit also controls a family of state-action value functions that back up the effects of exploration and longer discounts.

In total, the paper's contributions are the following:

1. a new parametrization of the state-action value function that decomposes the contributions of the intrinsic and extrinsic rewards, resulting in increased training stability over a large range of intrinsic reward scales.
2. a *meta-controller*: an adaptive mechanism to select which of the policies to prioritize throughout the training process. the policies are parametrized by exploration rate and discount factors.
3. a demostration of an agent which outperfoms HNS on all 57 Atari games.

4.3 Method

at this point i really need to read the never give up paper as the algorithm here is literally just an improvement over that one.

4.4 Other stuff