Reinforcement Learning
Final Project
Progress Report

Lawrence Kurowski 2019280743 / Jan Henrik 2019

30 May 2020

Summary

In this project we examine the implementation augmentations suggested in [1] to the Proximal Policy Optimization algorithm[2] applied to train an agent playing the game of football.[3]

As a starting point we select the PPO (proximal policy optimization) model created by OpenAI[2]. PPO is a RL model widely used due to its relative simplicity and efficiency compared to other approaches.[2] While the PPO framework is widely used, it is also notoriously difficult to train, especially in an environment as complex as Google Football. Engstrom et. al. [1] analyzed how fine-tuning the PPO code can aid making this framework more efficient to train. They tested their approach on a number of tasks, including continuous-space tasks such as humanoid running, as well as discrete-space tasks such as the game of Atari.

In this work, we will consider the considerably more challenging and relatively new environment Google Football, with two teams of 11 players, where the PPO optimizes a single agent (a single player). The environment is difficult to train due to action space size as well as multi-player complexity.

The baseline paper [1] provides examples trained on 3 random seeds which had been previously pointed out to be a potential weak point of this work.[1] We will thus aim to train the model for 10 seeds on a selection of code augmentations proposed in this paper.

Environment introduction

We will implement the model on the Google Football environment [3]. The environment contains a number of different gameplay scenarios, such as "empty goal", "3 vs 1 with keeper" etc. which simulate possible scenarios playing out during a game of football. In this paper we will focus our attention on the "11 vs 11" scenario which simulates the entire game.

The state space in this game corresponds to a tuple representing positions on the field of all the players, as well as that f the ball. The "goal" results in score +1.

[1] See the reviewers' comments available at `https://openreview.net/forum?id=r1etN1rtPB`

The action space corresponds to agents' movements ("up", "right", "up-right", etc.)

Model summary

We will base our PPO implementation on OpenAI's PPO2 baseline [4].

Proximal Policy Optimization replaces second-order optimization with first-order gradient descent-style optimization algorithm by optimizing a surrogate objective function which then defines the bounds for the actual objective.

The implementation "trick" examined in [1] concerns implementation parameter tuning independent of the algorithm itself. The implementation details examined include e.g. reward normalization, Adam annealing and network initialization. The paper also examine the effect of clipping effect (clipping refers to clipping of the probability ratio in the objective expectation that we are optimizing at the heart of the PPO algorithm.)

References

[1] Engstrom L., et.al., "Implementation Matters in Deep Policy Gradients: A Case Study on PPO and TRPO" (2019) ICLR

[2] Schulman et.al., "Proximal Policy Optimization Algorithms" (2017), arXiv:1707.06347

[3] Kurach et.al., "Introducing Google Research Football: A Novel Reinforcement Learning Environment" (2019) Google AI Blog, available at `ai.googleblog.com/2019/06/introducing-google-research-football.html` (accessed 2020-5-20)

[4] PPO2 baseline by OpenAI, repository available at: `github.com/openai/baselines/blob/master/baselines/ppo2`