Intelligent Decision-Making: Reinforcement Learning Applications in the Defense Environment

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Joseph Banks

Spring, 2022

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Rosanne Vrugtman, Department of Computer Science

Intelligent Decision-Making: Reinforcement Learning Applications in the Defense Environment

Abstract

Heron Systems is a government defense contractor focused on machine learning, working on problems such as AI dogfighting. Reinforcement learning is a form of machine learning that focuses on intelligent decision making in order to maximize cumulative reward. My work with Heron primarily involved improving Heron's open-source reinforcement learning library, adeptRL.The work also included designing, implementing, and training reinforcement learning models using adeptRL. Changes I made to the data-preprocessing pipeline of adeptRL increased training speed by 400%. All Heron projects using adeptRL switched over to my improved version allowing for faster model development and lower costs. AdeptRL and reinforcement learning more generally can be applied to many problems such as resource management, driving, etc.

1. Introduction

Reinforcement learning is one of the most challenging areas of machine learning for computer scientists to research or use because of the difficulty of transferring it to the real world (Dulac-Arnold, et al., 2019). Another challenge of reinforcement learning is that the best-performing models from cutting-edge research require prohibitive costs and frameworks to manage a large number of distributed environments simultaneously. For example, the Rain model required 34,200 GPU hours or 1425 days, thus making it unfeasible for smaller research groups (Castro, 2021). Interest in reinforcement learning has gained steam since the emergence of deep reinforcement learning, deep learning in reinforcement learning, and distributed training. These introductions to the field have led to newsworthy breakthrough events such as an RL agent, AlphaGO, defeating a professional Go player (Chan, 2017).

2. Related Works

"Playing atari with deep reinforcement learning" is the seminal paper on using deep learning with reinforcement learning, and all my work at Heron Systems is dependent on deep reinforcement learning's emergence and development (Mnih, et al., 2013). Another paper that introduced crucial techniques that informed my work is "Massively parallel methods for deep reinforcement learning" (Nair, et al., 2015). It provided the basis for all distributed agent reinforcement learning models.

The open-source library I spent a large portion of my internship working on provides a framework for distributed reinforcement learning. I worked directly on components related to managing distributed environments and sending data between them and the model. At Heron Systems, I worked on reinforcement learning self-play, which is having reinforcement learning agents learn by playing each other. "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play" highlights the power of using self-play to achieve superhuman results (Silver et al., 2018).

3. Process Design

At Heron Systems I worked on designing, implementing, and training reinforcement learning models and improving their open-source reinforcement learning library, adeptRL (Adept).

3.1 Data Pipeline Processing

In Adept, I worked on refactoring and updating the data preprocessing pipeline. Originally environment's data tensors,

n-dimensional matrices, were serialized using Python object pickling and sent to the model. I changed the serialization method to MSGPACK-Numpy so that less time when training would be spent waiting for the data from environments. I separated data preprocessing operations performed on the CPU and GPU to allow for easier development. I created abstract methods for operations so that engineers can easily extend them for their data preprocessing needs without needing to understand the framework itself. I also reimplemented all the preprocessing functions they had created before my changes.

3.2 Self-play League

I refactored Heron's self-play league for reinforcement learning agents. The league saves the agent at points during training, and the frequency of saves depends on the set parameters for training. Different versions of the agent training, versions of other agents, and human-designed intelligent agents are selected to play against the training model. Models in the league are ranked based on their performance using the Elo rating system or TrueSkill. The rankings and scores of agents can provide objective measures of agent skill.

3.3 Reward Schema and Model Creation

Reward schemas determine agent rewards from actions when playing a game. I created multiple reward schema for agents learning to play StarCraft. For example, two reward schemas created are one that only rewards the agent for killing an enemy troop and a reward schema that gives agents negative rewards when they lose one of their troops. At Heron using Adept, I implemented the model architecture of Google's AlphaStar model from the paper "Grandmaster level in StarCraft II using multi-agent reinforcement learning" (Vinyals, et al., 2019).

4. Results

My work at Heron Systems brought significant and positive results to their reinforcement learning framework, Adept. The changes to data serialization and the data preprocessing pipeline resulted in a 400% increase in training speed. All Heron projects using Adept switched over to my improved version allowing for faster model development and lower costs. Engineers at Heron continue to use my data preprocessing pipeline and implement their operations by extending my methods. Until the end of the contract associated with StarCraft ended, my reward schemas were used and updated.

5. Conclusion

During my time at Heron Systems, I was able to get hands-on experience with reinforcement learning and make significant contributions to the company. My work focused on reinforcement learning. I refactored parts of their framework, implemented models and rewards schema, and worked on a self-play league for agents. My refactors to Heron's reinforcement learning framework are in use across the company, and the work on the data preprocessing pipeline resulted in a 400% increase in training speed. The work I completed provided a meaningful learning experience and helped the company.

7. Future Work

Reinforcement learning is broadly applicable to problems or environments involving intelligent decision-making. Adept, the open-source reinforcement learning framework created by Heron Systems, can be used to train and develop agents for many scenarios such as resource management, driving, or famously dogfighting. My refactors to Adept will continue to provide an easy development experience for engineers creating data

preprocessing operations and using the framework in the future. Adept has areas that can continue to be improved, such as updating it to be compatible with more environments.

8. UVA Evaluation

The University of Virginia provided me with the core computer science skills needed to hit the ground running at Heron Systems. However, without my familiarity with Numpy or PyTorch, I would have had to spend a much longer time becoming comfortable with the technologies before being able to make an impact. Knowledge and comfortability with linear algebra and its concepts helped tremendously. I believe that linear algebra (APMA 3080 or an equivalent) should be required for computer science majors rather than just being one of the possible math course choices for the major.

9. Acknowledgements

Special thanks to all my coworkers at Heron Systems, and specifically my manager and the original creator of Adept, Joe Tatsuko.

References

Castro, P. S. (2021, July 13). Reducing the Computational Cost of Deep Reinforcement Learning Research [web log]. Retrieved from https://ai.googleblog.com/2021/07/reducing-computational-cost-of-deep.html.

Chan, D. (2017, October 20). The AI that has nothing to learn from humans. The Atlantic. Retrieved from https://www.theatlantic.com/technology/archive/2017/10/alphago-zero-the-ai-that-taught-itself-go/543450/

Dulac-Arnold, G., Mankowitz, D., & Hester, T. (2019). Challenges of Real-World Reinforcement Learning. ArXiv:1904.12901 [Cs, Stat]. http://arxiv.org/abs/1904.12901

Mnih, V., Kavukcuoglu, K., Silver, D., et al. 2013. Playing Atari with Deep Reinforcement Learning. http://arxiv.org/abs/1312.5602.

Nair, A., Srinivasan, P., Blackwell, S., et al. 2015. Massively Parallel Methods for Deep Reinforcement Learning. http://arxiv.org/abs/1507.04296.

Silver, D., Hubert, T., Schrittwieser, J., et al. 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science 362, 6419, 1140--1144.

Vinyals, O., Babuschkin, I., Czarnecki, W.M., et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nat. 575, 7782, 350–354.