Yuhao Peng

fhw9hh

Big Data

Report of A3

This report presents an experimental analysis aimed at understanding the impact of tuning three key hyper-parameters—num_agents, batch_size, and num_epochs_agents_train—on the performance of a distributed reinforcement learning setup. The experiments were designed to evaluate how these parameters affect key training metrics such as training time and training quality. The findings provide insights into the complex interplay between computational resources, network overhead, and training efficiency in a distributed learning environment.

Experimental Setup

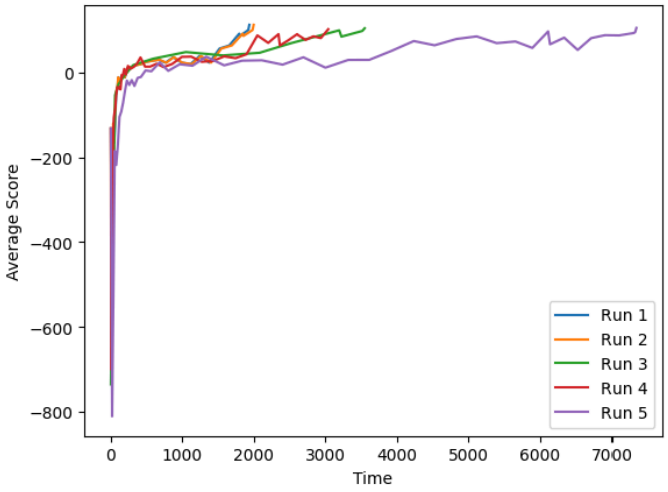
The experiments were conducted on a five-node cluster, each equipped with 2 CPUs, running distributed reinforcement learning tasks. The primary variables adjusted across experiments included the number of agent actors (num_agents), the batch size of states retrieved from the ReplayBuffer (batch_size), and the number of epochs an agent trains locally before updating the global model (num_epochs_agents_train). The default configuration was altered in the following ways across five experiments:

- Default settings for a baseline comparison.
- Reduced CPU allocation per actor from 2 to 1, without changing the number of agents.
- Increased num_epochs_agents_train from 1 to 3.
- Doubled the batch_size to 128.
- Increased num_agents from 4 to 9 while reducing CPU allocation per actor to 1.

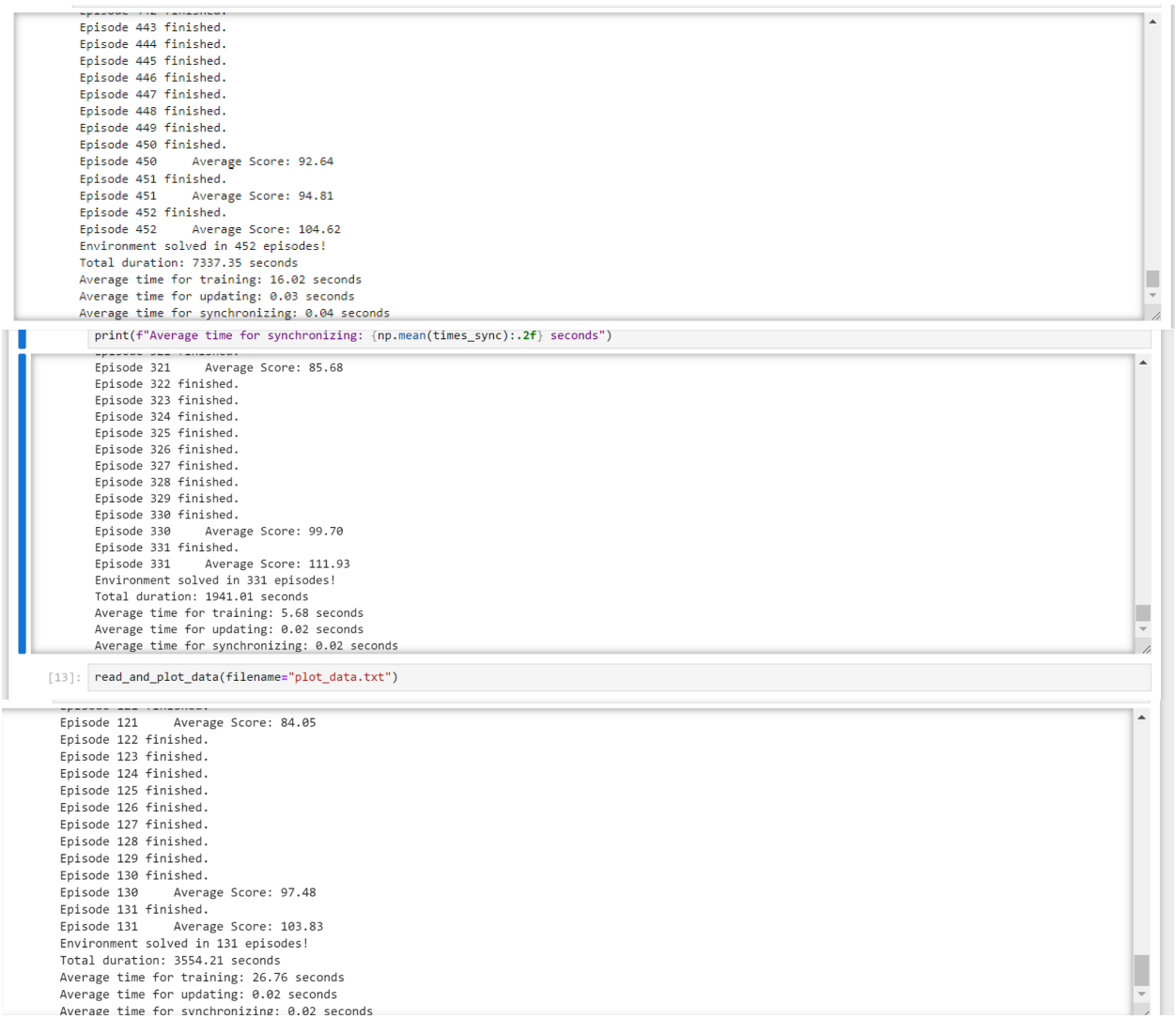
Observations and Analysis

The analysis of the experiments revealed a significant impact of the tuned parameters on training time and training quality. Specifically:

```
[13]: read_and_plot_data(filename="plot_data.txt")
```



- Experiment 5, which involved increasing the number of agents while reducing CPU allocation, resulted in the longest training time. This was attributed to increased local training time due to reduced CPU resources per agent, higher network traffic leading to increased transmission time, and a longer duration for the GlobalNet actor to merge training results due to decreased CPU availability.
- Experiments 3 and 4 also experienced slower training speeds. Increasing num_epochs_agents_train led to a threefold increase in local training time per agent. Doubling the batch_size resulted in increased network overhead and, consequently, longer transmission times.



```
Episode 443 finished.
Episode 444 finished.
Episode 445 finished.
Episode 446 finished.
Episode 447 finished.
Episode 448 finished.
Episode 449 finished.
Episode 450 finished.
Episode 450     Average Score: 92.64
Episode 451 finished.
Episode 451     Average Score: 94.81
Episode 452 finished.
Episode 452     Average Score: 104.62
Environment solved in 452 episodes!
Total duration: 7337.35 seconds
Average time for training: 16.02 seconds
Average time for updating: 0.03 seconds
Average time for synchronizing: 0.04 seconds
```

```
print(f"Average time for synchronizing: {np.mean(times_sync):.2f} seconds")
```

```
Episode 321     Average Score: 85.68
Episode 322 finished.
Episode 323 finished.
Episode 324 finished.
Episode 325 finished.
Episode 326 finished.
Episode 327 finished.
Episode 328 finished.
Episode 329 finished.
Episode 330 finished.
Episode 330     Average Score: 99.70
Episode 331 finished.
Episode 331     Average Score: 111.93
Environment solved in 331 episodes!
Total duration: 1941.01 seconds
Average time for training: 5.68 seconds
Average time for updating: 0.02 seconds
Average time for synchronizing: 0.02 seconds
```

```
[13]: read_and_plot_data(filename="plot_data.txt")
```

```
Episode 121     Average Score: 84.05
Episode 122 finished.
Episode 123 finished.
Episode 124 finished.
Episode 125 finished.
Episode 126 finished.
Episode 127 finished.
Episode 128 finished.
Episode 129 finished.
Episode 130 finished.
Episode 130     Average Score: 97.48
Episode 131 finished.
Episode 131     Average Score: 103.83
Environment solved in 131 episodes!
Total duration: 3554.21 seconds
Average time for training: 26.76 seconds
Average time for updating: 0.02 seconds
Average time for synchronizing: 0.02 seconds
```

Key Factors Influencing Training Results

- Local Training Time: The time agents spend on local model training directly impacts overall training duration. Increasing num_epochs_agents_train significantly adds to this time.
- Network Transmission Time: Larger batch sizes and increased number of agents contribute to higher network traffic, resulting in longer data transmission times.
- Global Model Update Time: The capacity for the GlobalNet actor to merge results is crucial. A higher number of agents and reduced CPU resources per actor extend the time required for model updates.

The experiments underscore the importance of balancing computational resources, network load, and training parameters to optimize distributed reinforcement learning performance. Key insights include:

- Resource Allocation: Optimal CPU allocation is critical for balancing local training efficiency and global model update speed.
- Parameter Tuning: Careful adjustment of num_agents, batch_size, and num_epochs_agents_train can significantly impact training efficiency. A deeper understanding of their interplay is essential for further optimizations.
- Network Efficiency: Minimizing network overhead through efficient data transmission and model update strategies is vital for scaling distributed learning systems.