Intelligent Agent, Smart Enough to be Stupid

Nayoung Oh[1]

Abstract: To enhance the satisfaction of a partner, it is necessary to adjust skills following the partner. Therefore, this paper proposes a "Mimicking Transformer" trained on the learning history of an on-policy RL model.

Keywords: Human-AI interaction, Cooperative AI, Decision Transformer

1. Introduction

When you play with your young nephew, doing your best is not the best way to keep playing with him regardless the type of games, cooperative or competitive. Instead, you need to adjust your skill to be similar to him. However, current reinforcement learning approaches to tasks with a human partner only focus on achieving better performance on the task. To reflect how people play with the partner, this paper suggests 'Mimicking Transformer' trained on the learning history of an on-policy RL model.

2. Method

2.1 Mimicking Transformer

One of the easiest ways to understand a partner is analyzing the past history of the partner. Furthermore, the 'understanding' can be used to infer a suitable action of an ego agent, mimicking the policy of the partner. However, most of conventional reinforcement learning approaches do not utilize the past history to predict a future history. Based on the first-order Markov assumption as in Eq (1), it neglects the previous history and only uses the right before state.

$$P(s_t|s_{t-1}, s_{t-2}, \cdots s_1) = P(s_t|s_{t-1}) \quad \cdots \ (1)$$

In the case of cooperative AI, the partner agent can be the part of states. Thus, as the AI agent works with diverse partners, the partners will choose different actions, changing the states in different ways. Therefore, the first-order Markov assumption does not hold well in the cooperative setting. Therefore, the conventional RL setting is not suitable for the target environment and difficult to extend to mimicking behavior patterns.

Therefore, the "Decision Transformer" architecture [1] is adopted to train the cooperative AI agent. In this paper, the modified model is called "Mimicking Transformer". Instead of taking a sequence of reward, state, action, the Mimicking Transformer takes a sequence of state, action of the partner, action of myself to predict an action of myself based on the previous history. As the purpose of the Mimicking Transformer is not getting high reward but achieving similar behavior patterns to the partner, it does not use a reward.

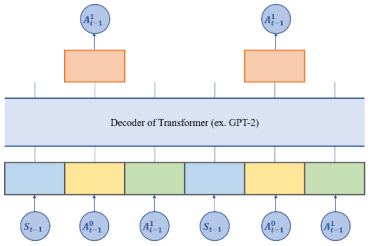

Figure 1. The Mimicking Transformer architecture

2.2 Data Generation for Training Mimicking Transformer

To train the Mimicking Transformer, sequences of states and actions of multiple agents are necessary. Collecting human cooperation sequence data is the best way to gather optimal sequences, but too costly. Therefore, I train an on-policy RL model to collect sequences. Training one RL model is easier and requires less time than training multiple models in a multi-agent setting. Furthermore, since one model decides the actions of multiple agents, the multiple agents are likely to share similar behavior patterns.

3. Experiments

3.1 Test Space

In the test space, two agents should gather coins in the 10×10 grid world. The agent can choose action among up, down, left, right, and None. There are always two coins so that two agents can divide up their works. Two agents need to gather 5 coins to finish the task, as soon as possible.

To check whether an agent can learn both high-level and low-level pattern, I set 4 combinations of behavior patterns.

Type	Target Coin	Trajectory Pattern
Type 0	Close to me	Left-right prefer
Type 1	Close to me	Up-down prefer
Type 2	Far from you	Left-right prefer

1. School of Computing, KAIST, Daejeon, Korea (nyoh@kaist.ac.kr)

Type 3	Far from you	Up-down prefer

Table 1. Sample Behavior Patterns

For example, the type 0 agent selects the coin close to itself and approaches it by moving left or right first and up or down later.

3.2 Experiment 1. Is the Mimicking Transformer Suitable?

To check whether the Mimicking Transformer architecture is suitable, I conducted an experiment with the Mimicking Transformer trained on sample data. The sample data are generated by the rule-based model following Table 1. In detail, the first model was trained with only 2,000 trajectories data of type 0 agent (Type 0 T). The second model was trained with 2,000 trajectories of type 0~3 agents (Type 0~3 T).

The performance is evaluated by two metrics. The first one is guessing accuracy. The output action of a model should be same as the action of a rule-based model agent in the same situation. For example, with a partner type 0, the model should act like the type 0 agent. The second one is a reward to check performance of completing the task.

3.3 Experiment 2. Is the RL Learning History Sequence Appropriate?

In experiment 2, the sample data is replaced by the actual learning history of PPO model, total of 2,625 trajectories. The other setting is identical to experiment 1.

4. Results and Discussions

I trained two baseline models. The first baseline model is a PPO model trained with the other PPO model agent (PPO-PPO). The second baseline model is a PPO model trained with a rule-based type 0 model (PPO-Type 0).

4.1 Experiment 1 Result and Discussion

Partner	Type 0 T	Type 0~3 T	PPO-PPO	PPO-Type 0
Type 0	**97.6%**	73.3%	33.2%	25.2%
Type 1	54.7%	**60.1%**	32.0%	19.4%
Type 2	**73.7%**	62.4%	35.7%	20.2%
Type 3	47.1%	**55.4%**	24.1%	17.0%
Average	**68.3%**	62.8%	31.3%	20.4%

Table 2. Experiment 1 Guessing Accuracy

Partner	Type 0 T	Type 0~3 T	PPO-PPO	PPO-Type 0
Type 0	41.4	**42.9**	38.6	38.1
Type 1	**41.8**	40.9	39.1	38.7
Type 2	43.8	**44.7**	21.2	35.1
Type 3	**43.6**	42.7	35.7	31.6
Average	42.7	**42.8**	33.7	35.9

Table 3. Experiment 1 Average Reward

The type 0 trained agent and the type 0~3 trained agent have higher guessing accuracy and better performance on the task than the two baselines. Interestingly, the type 0 trained agent has the highest guessing accuracy toward the type 3 partner. It is probably due to the similarity of type 0 and type 3 agents since a close to me coin and a far from you coin can be identical. However, as the type 0 agent has relatively high accuracy to the other types, it is possible to say the Mimicking Transformer model learns how to mimic generally.

4.2 Experiment 2 Result and Discussion

Partner	Original PPO	Mimicking PPO	PPO-PPO	PPO-Type 0
Type 0	43.8%	**45.3%**	33.2%	25.2%
Type 1	38.6%	**41.8%**	32.0%	19.4%
Type 2	40.6%	**48.3%**	35.7%	20.0%
Type 3	**43.6%**	40.8%	24.1%	17.0%
Average	41.7%	**44.1%**	31.3%	20.4%

Table 4. Experiment 2 Guessing Accuracy

Partner	Original PPO	Mimicking PPO	PPO-PPO	PPO-Type 0
Type 0	**39.8**	39.3	38.6	38.1
Type 1	**39.5**	39.4	39.1	38.7
Type 2	35.5	**36.8**	21.2	35.1
Type 3	38.2	**39.9**	35.7	31.6
Average	38.3	**38.9**	33.7	35.9

Table 5. Experiment 2 Average Reward

With the PPO model learning history, the guessing accuracy and the average reward become lower. However, interestingly, the Mimicking Transformer has higher guessing accuracy and average reward than the original PPO. It means the Mimicking Transformer learns general ways to mimic the partner and it leads to the better performance.

5. Conclusion

With the mimicking transformer trained on the learning history of an on-policy RL model, it is possible to mimic the partner and achieve high accuracy with a naïve partner. By increasing the diversity of collected trajectories, the performance can be improved.

References

[1] Chen, Lili, et al. "Decision transformer: Reinforcement learning via sequence modeling." *Advances in neural information processing systems 34* (2021).