Reward Optimization of Imitation Learning in Offline Multi-Agent Games from Human Feedback

Shuixiunan Zhang* **Zhixuan Fang†**

Abstract

This study explores the integration of human feedback into Imitation Learning (IL) for offline multi-agent reinforcement learning tasks, focusing on cooperative environments where explicit rewards are not accessible. We introduce a reward model trained on human preference data, which is used to filter and optimize the IL dataset, improving agent performance. Experiments show that our method outperforms standard IL by effectively identifying the high-quality trajectories, especially in diversified datasets where demonstration quality varies. This research highlights the potential of using human feedback for reward optimization to improve IL outcomes, fostering more robust and adaptable multi-agent AI systems.

1 Introduction

In complex multi-agent environments, such as autonomous driving, robotics, and strategic games, the ability to train agents that can effectively collaborate and adapt to dynamic scenarios is crucial. Traditional reinforcement learning (RL) methods rely heavily on well-defined reward functions to guide agent behavior. However, designing these reward functions in real-world scenarios is often challenging, if not infeasible, due to the complexity and unpredictability of the environment. This challenge is particularly pronounced in multi-agent settings, where interactions between agents can lead to non-linear and emergent behaviors that are difficult to anticipate [Busoniu et al., 2008].

Imitation Learning (IL) has emerged as a practical solution to these challenges by allowing agents to learn directly from expert demonstrations [Pomerleau, 1988]. In IL, agents attempt to replicate the behavior of human experts, bypassing the need for explicitly defined reward functions. However, while IL has shown promise, its effectiveness is highly dependent on the quality of the provided demonstrations [Ross et al., 2011]. In multi-agent games, where the actions of one agent can significantly impact the outcomes for others, ensuring that agents learn the most effective strategies is critical for success. This need for high-quality, robust learning is further complicated by the variability and potential suboptimality of the demonstrations [Foerster et al., 2018].

Incorporating human feedback into the IL process offers a promising avenue to address these challenges. Human feedback, particularly in the form of preferences or ratings, provides a valuable source of information that can guide the optimization of agent behavior beyond what is possible with demonstrations alone [Christiano et al., 2017]. By leveraging human input, agents can refine their strategies to better align with human expectations and objectives, which is particularly important in applications where agents interact closely with humans or operate in human-centric environments.

This approach to incorporating human preferences has also seen application in the development and fine-tuning of large language models (LLMs). For example, models like GPT-3 and GPT-4 have been fine-tuned using preference-based reinforcement learning techniques to align their outputs with human values and user preferences, resulting in more accurate, contextually appropriate, and user-aligned responses [Ouyang et al., 2022].

*Yao 14, 2021011832, Tsinghua University
†IIIS, Tsinghua University

Preprint. Under review.

The importance of optimizing reward mechanisms in IL, especially in multi-agent settings, cannot be overstated. Effective reward optimization not only improves the learning process but also enhances the overall robustness and adaptability of the agents.

This research focuses on the optimization of reward mechanisms in IL within multi-agent cooperative games, incorporating human feedback to enhance the learning process. By developing methods to filter and refine training data based on preferences, we aim to improve the quality of agent strategies in interactive environments.

To address the variability in dataset quality that can impact IL performance, this study proposes a novel method to filter high-quality trajectories, thereby improving the overall efficacy of IL. Specifically, we aim to implement a reward model based on preference tags to predict dataset quality, which will be used to filter the dataset for IL training. Experimental evaluations within a multi-agent game environment will be conducted to validate the effectiveness of this approach.

2 Related works

RLHF Reinforcement Learning from Human Feedback (RLHF), also known as preference-based reinforcement learning (PbRL), is a paradigm that seeks to improve the training of reinforcement learning agents by incorporating human feedback, typically in the form of preferences or demonstrations. While traditional RL relies on a well-defined reward function, which often desires a lot of task-specific prior knowlege, RLHF only leverages preference signals from human, making it useful in environments where specific reward function are difficult to get. The work by [Wirth et al., 2017] suveryed on the method in RLHF where the expert's preferences are utilized for training instead of a hand-designed numeric reward. These approach has been extended to later works, such as [Ibarz et al., 2018], who combined expert demonstrations and trajectory preferences to enhance agent learning. Another foundational work by [Christiano et al., 2017] provides efficient and practical algorithms to train RLHF agents on various tasks including Atari games and simulated robot locomotion.

MARL Multi-Agent Reinforcement Learning (MARL) is a subfield of RL where multiple agents interact in the same environment, maximizing their own reward. This interaction can be cooperative and competitive, while agents in cooperative game share a same reward and agents in competitive game compete with others to maximize its own. The work by [Lowe et al., 2020], introduced the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm specifically designed to handle these mixed environments by allowing agents to learn decentralized policies using centralized training. One of the seminal works in MARL by [Huh and Mohapatra, 2024] provides a survey of the key challenges and approaches in this domain, where the non-stationarity of environments and the scalability issues arise when multiple agents are involved. More recently advances, such as [Foerster et al., 2018] proposed an approach to stabilize experience replay in MARL, addressing one of the critical issues of training stability in multi-agent settings. These foundational works have set the stage for ongoing research in MARL, which continues to explore new algorithms and methods to improve agent coordination and learning efficiency in complex, multi-agent environments. Yet most of the settings require a specific numeral reward, leaving the preference-based RLHF unexplored.

IL Imitation Learning (IL) is a technique where agents learn by mimicking the behavior of an expert without explicit reward function. A straightforward form of IL is Behavior Cloning, which is popularized by [Pomerleau, 1988]. It maps the observations into experts actions correspondingly. This simple and efficient method offers a direct and intuitive approach to learning complex behaviors, but it still suffers from a series of limitations like covariate shift problem, where small errors in predictions leads to significant deviations from expert behavior. From automatic driving to games, a series of IL algorithms have been proposed to address the challenges, including DAGGER [Ross et al., 2011], GAIL [Ho and Ermon, 2016], BCO [Torabi et al., 2018]. Despite the advantages of these approaches, noisy or suboptimal demonstrations can still lead to poor policy performance [Brown et al., 2019], which has motivated research into methods for filtering and improving demonstration data.

Offline MARL Offline Multi-Agent Reinforcement Learning (Offline MARL) is an emerging field that addresses the challenges of training multiple agents in complex environments using pre-collected datasets without the need for online interaction [Levine et al., 2020]. This approach is particularly

valuable in scenarios where real-time data collection is expensive, risky, or even impractical. Unlike traditional MARL, where agents continuously interact with the environment to gather data, Offline MARL must overcome the challenge of learning effective policies from a static dataset that may not adequately cover the state-action space. Techniques such as CQL [Kumar et al., 2020], MOPO [Yu et al., 2020], and MOReL [Kidambi et al., 2021] have been explored to address these challenges.

3 Preliminaries

Cooperative Markov Games We consider an episodic time-inhomogeneous cooperative Markov game $\mathcal{M}$, consisting of m agents, a shared state space $\mathcal{S}$, an individual action space $\mathcal{A}_i$ for each agent $i \in [m]$, and a joint action space $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \cdots \times \mathcal{A}_m$. The game proceeds over a time horizon H with an initial state s_1, state transition probabilities $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2, \cdots, \mathcal{P}_H)$, where $\mathcal{P}_h : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ defines the probability distribution over the next state given the current state and joint action at step h. The shared reward function $R = \{R_h(\cdot \mid s_h, \mathbf{a}_h)\}_{h=1}^H$ is common to all agents, where $R_h(s_h, \mathbf{a}_h) \in [0, 1]$ represents the reward received by all agents at step h.

At each step $h \in [H]$, all agents observe the current state s_h and simultaneously choose their actions $\mathbf{a}_h = (a_{h,1}, a_{h,2}, \cdots, a_{h,m})$. The next state s_{h+1} is then sampled from $\mathcal{P}_h(\cdot \mid s_h, \mathbf{a}_h)$, and the shared reward r_h is obtained from $R_h(\cdot \mid s_h, \mathbf{a}_h)$. The game terminates at step $H + 1$, and the objective is for all agents to cooperatively maximize the total collected reward.

We denote the joint policy by $\pi = (\pi_1, \pi_2, \cdots, \pi_m)$, where the individual policy for agent i is represented as $\pi_i = (\pi_{1,i}, \pi_{2,i}, \cdots, \pi_{H,i})$, with each $\pi_{h,i} : S \to \Delta(A_i)$ defining the Markov policy for agent i at step h.

The state value function and state-action value function under the joint policy π are defined as:

$$V_h^\pi(s_h) := \mathbb{E}_\pi \left[\sum_{t=h}^{H} r_t(s_t, \mathbf{a}_t) \mid s_h \right], \quad Q_h^\pi(s_h, \mathbf{a}_h) := \mathbb{E}_\pi \left[\sum_{t=h}^{H} r_t(s_t, \mathbf{a}_t) \mid s_h, \mathbf{a}_h \right],$$

where $\mathbb{E}_\pi = \mathbb{E}_{s_1, \mathbf{a}_1, r_1, \cdots, s_{H+1} \sim \pi, \mathcal{M}}$ denotes the expectation over the random trajectory generated by policy π.

The optimal joint policy π^* in a cooperative Markov game is one that maximizes the total expected reward:

$$\pi^* = \arg\max_\pi V_1^\pi(s_1),$$

where $V_1^\pi(s_1)$ is the expected return starting from the initial state s_1.

The goal in cooperative settings is to find a policy π^* that leads all agents to achieve the highest possible cumulative reward, ensuring that their actions are well-coordinated to optimize the collective outcome.

Offline MARLHF We focus on Offline Multi-Agent Reinforcement Learning from Human Feedback (MARLHF). It extends the framework of offline RL to multi-agent environments where agents learn from pre-collected datasets, using human feedback rather than reward as the signal. Unlike online learning where agents interact with the environment in real-time, offline MARLHF operates solely on a pre-collected preference dataset generated by an unknown behavior policy π_b in a Markov game $\mathcal{M}$.

Rewards are not directly provided in the dataset; instead, they are inferred from human preferences.

The dataset consists of trajectory pairs (τ, τ'), where each trajectory $\tau = (s_1, \mathbf{a}_1, s_2, \mathbf{a}_2, \ldots, s_{H+1})$ is sampled from the distribution $\mathcal{P}(s_1, \mathbf{a}_1, s_2, \ldots, s_{H+1}) = \prod_h \pi^b(\mathbf{a}_h \mid s_h)\mathcal{P}(s_{h+1} \mid s_h, \mathbf{a}_h)$ under the behavior policy π^b. Human feedback is provided in the form of preference signals y, where y is a binary variable indicating which of the two trajectories is preferred. This preference is modeled by a Bernoulli distribution:

$$\mathbb{P}(y=1 \mid \tau, \tau') = \frac{\exp\left(\sum_{h=1}^{H} r_h(s_h, \mathbf{a}_h)\right)}{\exp\left(\sum_{h=1}^{H} r_h(s_h, \mathbf{a}_h)\right) + \exp\left(\sum_{h=1}^{H} r_h(s'_h, \mathbf{a}'_h)\right)}$$

The goal of MARLHF in cooperative settings is to learn an optimal joint policy π^* that maximizes the expected cumulative reward for all agents, leveraging the feedback provided in the dataset to guide learning despite the offline nature of the data.

Though our methods are applied on the cooperative Markov games, they could also be extended to general-sum games aiming to reach a Nash equilibrium where each player maximizes an individual reward. In this context, each player can observe a binary signal y_i indicating the preference, and the preference model is extended to:

$$\mathbb{P}(y_i=1 \mid \tau, \tau') = \frac{\exp\left(\sum_{h=1}^{H} r_{h,i}(s_h, \mathbf{a}_h)\right)}{\exp\left(\sum_{h=1}^{H} r_{h,i}(s_h, \mathbf{a}_h)\right) + \exp\left(\sum_{h=1}^{H} r_{h,i}(s'_h, \mathbf{a}'_h)\right)}, \quad \forall i \in [m].$$

Behavioral Cloning Behavioral Cloning (BC) proposed by [Pomerleau, 1988] is an imitation learning approach that frames the problem as supervised learning. Given a dataset $\mathcal{D} = \{\tau^i\}_{i=1}^N$ of expert demonstrations, where each trajectory $\tau^i = (s_1^i, \mathbf{a}_1^i, s_2^i, \mathbf{a}_2^i, \ldots, s_H^i, \mathbf{a}_H^i)$ consists of state-action pairs generated by an expert policy π^E, the goal is to learn a policy π that replicates the expert's behavior.

The objective in BC is to minimize the difference between the actions predicted by the learned policy π and the actions taken by the expert in the dataset. This is achieved by minimizing the following loss function:

$$\mathcal{L}_{\text{BC}}(\pi) = \mathbb{E}_{(s_h, \mathbf{a}_h)\sim\mathcal{D}}\left[\|\mathbf{a}_h - \pi(s_h)\|^2\right],$$

where $\pi(s_h)$ represents the action predicted by the learned policy at state s_h, and $\mathbf{a}_h$ is the corresponding action taken by the expert. The learning process aims to find a policy π that closely matches the expert's policy π^E across the states encountered in the dataset.

BC is straightforward to implement and can be effective in environments where a large amount of high-quality expert data is available. We use BC as our basic method for IL experiments and apply reward optimization on this algorithm.

4 Experiments

4.1 Pipeline

Our experiment pipeline is designed to systematically evaluate the performance of offline MARLHF by leveraging mixed joint policy datasets and refining models through reward prediction and optimization. The pipeline consists of the following key steps:

- **Dataset Preparation and Preference Pair Simulation**: We begin by preparing datasets containing mixed joint policy trajectories, as detailed in Section 4.3. These trajectories are generated using a combination of expert, rookie, random, and mixed policies. Once the datasets are ready, we simulate preference pairs to represent human feedback as modeled in Equation 3 in the preliminary section.
- **Reward Model Training**: Next, we use nuerl networks to train a reward model in order to predict step-wise rewards $r_\phi(s_h, \mathbf{a}_h)$ associated with each trajectory. The reward model is tasked with learning from the simulated preference pairs y, where it should estimate the reward signals for different states and actions.
- **Reward Optimization**: After training the reward models, we perform reward optimization to refine the dataset. We filter the original dataset based on the reward model's predictions, retaining only a certain ratio of trajectories with high accumulated predicted rewards $\sum_{h=1}^{H} r_\phi(s_h, \mathbf{a}_h)$. This selective filtering helps to focus the training on high-quality data, which is expected to improve the learning process of the imitation model.

- **Imitation Model Training**: With the filtered demonstration data, we proceed to train an imitation model. This model is designed to mimic the behavior reflected in the filtered trajectories, effectively learning a policy that aligns with the high-reward actions identified by the reward model. The imitation model serves as an essential component for guiding the final policy optimization.
- **Fine-Tuning**: Finally, we test the performance of the models by fine-tuning the imitation model, using it as a reference during the evaluation phase of an offline MARL algorithm.

4.2 Environment Settings

The experiments are set up on the codebase of JaxMARL [Rutherford et al., 2023], a modern framework for MARL that provides robust and efficient tools for implementing and testing various algorithms.

We selected the Multi-Agent Particle Environment (MPE) as the testbed of our experiments due to its well-established use as a benchmark for MARL algorithms. The MPE environment provides a diverse set of tasks that are instrumental in evaluating the performance and robustness of MARL techniques.

Specifically, we have chosen three tasks: Simple spread, Simple reference, and Simple tag.

- **Simple Spread**: In this task, a group of agents must spread out to occupy different landmarks while avoiding collisions with others.
- **Simple Tag**: In this task, adversarial agents work together to catch evading agents, while the evaders attempt to avoid capture. The original 1v3 adversarial setup was modified into a cooperative scenario by fixing the evader with a MAPPO pretrained policy to better evaluate the cooperative agents' performance.
- **Simple Reference**: In this task, agents must reach target landmarks, with the locations known only to other agents, necessitating effective communication to succeed.

These tasks provide a comprehensive framework to evaluate the effectiveness and adaptability of our approach in various multi-agent environments. They also highlight environments that are sensitive to dataset coverage, where a dataset dominated by mixed policies can disrupt effective cooperation in the multi-agent settings. Additionally, we choose cooperative settings in our experiments to better evaluate the effectiveness of our approach in improving agent performance through higher-quality data and reward optimization.

In our experiments, the preference of trajectories are simulated based on the ground-truth reward values from the environment. While these preferences are used to train the reward model, the true reward values are only employed for preference generation and are not included in any training sections. This ensures that the training process remains realistic and reliant solely on the simulated human feedback.

4.3 Dataset

We manually designed 3 trajectory datasets generated by varying the ratio of mixed joint policies to illustrate the impact of dataset diversity and quality on the training process. Different combinations of policies can provide varying levels of information to the training model, thereby influencing its ability to accurately learn and generalize. Our training datasets were prepared as follows:

- Train agents of different policies with traditional offline MARL algorithm. Specifically, we employed Value Decomposition Networks (VDN) [Sunehag et al., 2017], which is a Q-learning-based architecture for cooperative games. It works by decomposing the team's value function into individual agent value functions: $Q_h(s, \mathbf{a}) = \sum_{i=1}^{n} Q_{h,i}(s, a_i)$.
  - Expert Policy: all the agents are trained till converge.
  - Rookie Policy: all the agents are trained at an early stop.
  - Random Policy: all the agents use a uniform random action.
  - Mixed Policy: one agent of rookie policy and others of expert policy.
- The trained agents were then used to collect raw trajectories.

	Expert	Unilateral	Rookie	Trivial
Diversified	1	1	1	1
Mix-Expert	3	0	0	1
Pure-Expert	4	0	0	0

Table 1: Final datasets mixed with various ratios.

- We mixed these trajectories in different ratios to create four datasets for MARLHF training. The specific mixing ratios are detailed in Table 1.

By strategically varying the composition of these datasets, we aim to explore not only the direct impact of policy diversity and quality on model performance but also how these factors influence the model's ability to handle uncertainty during the offline training. This design allows us to gain insights into the robustness of multi-agent systems when faced with heterogeneous environments and varying agent capabilities.

Our datasets consists of 38,400 trajectories in each environment. For each dataset, we generated trajectory pairs amounting to 10 times the number of individual trajectories. Preference labels were then assigned to these pairs within the mixed datasets.

To modulate the randomness of the preferences, we also introduced a steepness parameter that scales the standardized reward. This setup ensures that our datasets are robust and well-suited for thoroughly evaluating the performance of our methods.

4.4 Imitation Learning with Reward Optimization

In this experiment, we use a reward model to enhance the effectiveness of our Imitation Learning (IL) agent through reward optimization. The reward model is trained to predict rewards based on the collected dataset, with configurations tailored to each environment. In specific, we use a Multi-Layer Perceptron (MLP) in Simple Spread and Simple Reference environment, and use an RNN in Simple Tag environment.

It is worth mentioning that learning an effective reward model from an offline dataset is particularly difficult due to the limitations in dataset size and diversity. Offline datasets often represent only a small portion of the possible trajectory space, limiting the amount of information available for training. This constraint makes it challenging to learn a reward model that accurately captures the underlying dynamics and objectives of the environment. Moreover, any biases or gaps in the dataset can significantly impact the quality of the learned reward model, leading to inaccuracies in the agent's policy.

To address the challenges, we apply a technique of Mean Squared Error (MSE) regularization along the time axis, as proposed in previous work [Zhang et al., 2024]. This regularization method was employed to smooth the reward predictions over time by limiting abrupt changes between adjacent time steps, thereby preventing the reward model from focusing excessively on a few specific moments within the trajectory. The regularization was formulated as:

$$L_{\text{RM}}(\phi) = -\mathbb{E}_{(\tau_1,\tau_2)\sim\mathcal{D}}\left[\log \sigma(y(r_\phi(\tau_1) - r_\phi(\tau_2)))\right] + \frac{\alpha}{\text{Var}_{\mathcal{D}}(r_\phi)} L_{\text{MSE}}(\phi, \tau), \tag{1}$$

where the regularization term L_{MSE} is defined as:

$$L_{\text{MSE}}(\phi, \tau) = \mathbb{E}_{\mathcal{D}}\left[\sum_{h=1}^{H-1} \|r_\phi(s_h, \mathbf{a}_h) - r_\phi(s_{h+1}, \mathbf{a}_{h+1})\|_2^2\right]. \tag{2}$$

The regularization coefficient α was set to 1, and the variance of the predicted rewards was computed over the training set to dynamically scale the regularization term. This technique was incorporated into our model training pipeline in improving performance and ensuring stable reward prediction.

Once the reward model is trained, it is used to filter the demonstration dataset for the IL model. Specifically, the predicted rewards are used to select the trajectories that are most likely to lead to successful outcomes.

filter ratio	1.0	0.8	0.6	0.4
Diversified	-19.5	-15.5	-11.9	**-11.0**
Mix-Expert	-18.4	-12.9	**-10.9**	**-10.8**
Pure-Expert	**-10.9**	**-10.9**	**-10.8**	**-10.9**

Table 2: Test rewards of IL agents in Simple Spread.

filter ratio	1.0	0.8	0.6	0.4
Diversified	-23.6	-17.7	-13.0	**-11.7**
Mix-Expert	-21.7	-14.0	**-11.0**	**-11.0**
Pure-Expert	**-11.0**	**-11.0**	**-10.9**	**-11.0**

Table 3: Test rewards of IL agents in Simple Reference.

filter ratio	1.0	0.8	0.6	0.4
Diversified	26.6	30.4	35.1	**35.8**
Mix-Expert	30.5	35.3	**37.0**	36.5
Pure-Expert	**37.6**	36.8	37.4	36.3

Table 4: Test rewards of IL agents in Simple Tag.

For IL agents, we use the Behavioral Cloning (BC) algorithm (cf. Section 3), implemented as an RNN. We chose BC due to its simplicity and effectiveness in offline reinforcement learning scenarios. IL agents are trained on the filtered trajectories as the demonstration data, using the predicted rewards to guide the learning process.

The experiment results presented in Tables 2, 3, and 4 demonstrate the impact of filtering the dataset on the performance of IL agents across different environments. The filtering ratio is a parameter that controls the proportion of the dataset retained for training. A filtering ratio of 1.0 means no data was filtered out, while lower ratios indicate progressively more stringent filtering.

The results consistently show that applying filtering to the dataset improves the performance of IL agents. This improvement is particularly pronounced in more diversified datasets. For instance, we observe that as the filtering ratio decreases, the IL agents trained on Diversified datasets achieve significantly better test rewards.

In the Pure-Expert rows, the improvement is less noticeable, since expert demonstrations already provide high-quality data, leaving less room for improvement through filtering. However, even in these cases, applying filtering does not degrade performance or introduce harmful biases.

In the Mix-Expert rows, we observe that adding even a small amount of lower-quality data to an otherwise high-quality dataset can lead to a noticeable drop in performance. This outcome highlights the sensitivity of IL algorithms to the presence of suboptimal trajectories, which can dilute the learning signal and result in poor generalization. However, the application of reward optimization effectively mitigates this issue.

The Diversified dataset presents an even more challenging scenario, as it contains a smaller proportion of expert data and a larger amount of suboptimal trajectories. Despite this challenge, the success of the filtering process in improving IL performance is evident. This suggests that our reward model is successfully identifying and prioritizing the most informative trajectories, thereby significantly contributes to the training of IL agents.

4.5 Fine-tuning Results

We train the fine-tuned agents by taking the IL model as a reference during the evaluation phase of an offline MARL algorithm. Consistent with the approach outlined in [Ouyang et al., 2022], we applied a per-step Kullback-Leibler (KL) divergence reward to enforce constraints between the learned policy

filter ratio	1.0	0.8	0.6	0.4
Diversified	-23.7	-23.4	-21.2	-21.0
Mix-Expert	-21.1	-20.8	-20.4	-20.1
Pure-Expert	-20.1	-20.4	-20.1	-19.8

Table 5: Test rewards of VDN agents with IL as reference in Simple Spread.

filter ratio	1.0	0.8	0.6	0.4
Diversified	-17.9	-17.9	-17.9	-17.7
Mix-Expert	-18.5	-18.6	-18.6	-18.6
Pure-Expert	-31.4	-29.7	-29.7	-31.9

Table 6: Test rewards of VDN agents with IL as reference in Simple Reference.

IL filter ratio	1.0	0.8	0.6	0.4
Diversified	30.1	32.4	37.3	**37.5**
Mix-Expert	33.0	34.3	34.6	**35.0**
Pure-Expert	39.0	39.2	**39.6**	38.9

Table 7: Test rewards of VDN agents with IL as reference in Simple Tag.

$\pi_{\mathbf{w}}$ and the reference policy π_{ref}, which is estimated via imitation learning. The objective function for the policy $\mathbf{w}$ was defined as:

$$\text{objective}(\mathbf{w}) = \mathbb{E}_{\tau\sim\pi_{\mathbf{w}}}\left[\sum_{h=1}^{H} r_\phi(s_h, \mathbf{a}_h) - \beta \log \frac{\pi_{\mathbf{w}}(s_h, \mathbf{a}_h)}{\pi_{\text{ref}}(s_h, \mathbf{a}_h)}\right],$$

where r_ϕ is our standardized reward model, and β is the KL reward coefficient.

In our experiments, we utilized VDN for policy optimization. Since VDN does not provide an explicit form of the policy $\pi_{\mathbf{w}}(s_h, \mathbf{a}_h)$, we replaced this term with a uniform probability $1/|A|$, where $|A|$ is the number of possible actions. This substitution allowed us to approximate the KL reward, guiding the IL agent to favor actions that align with the reference policy's preferred choices.

The experiment results in Table 7 show the test rewards of VDN agents using IL as a reference model in the Simple Tag environment, evaluated across different IL filter ratios. Remark that the IL filter ratio controls the proportion of the dataset used for training the IL model, with lower ratios indicating stricter filtering.

We observe that VDN agents using IL as a reference model outperform the IL agents alone across all filter ratios. This performance boost can be attributed to the fact that our algorithm effectively integrates the learned policy from IL with its own policy optimization process, utilizing reward predictions provided by the reward model. The IL agents, having already distilled useful knowledge from the expert data, serve as a strong foundation upon which VDN builds and further refines the policy.

Additionally, the performance of the fine-tuned VDN models using different IL agents as references mainly aligns with the IL agents' own performance, indicating that the IL model's quality directly influences the effectiveness of the fine-tuned agent. However, we do observe that agents trained on the Diversified dataset surpasses that of those trained on the Mix-Expert dataset. This can be explained by the fact that the reward model, when trained on a more diversified dataset, becomes more adept at identifying trajectories. During VDN training, this refined reward model replaces the true reward function, which is not directly accessible in the offline MARLHF setting. As a result, the VDN agents benefit from a more accurate and contextually relevant reward signal, leading to better overall performance.

5 Discussion

5.1 Conclusion

This study presents a novel approach to enhancing Imitation Learning (IL) in offline multi-agent reinforcement learning environments by integrating human feedback into the learning process. By employing a reward model trained on human preferences, we effectively filter and optimize the dataset used for IL, allowing agents to focus on the most informative trajectories. This approach proves especially effective in diversified datasets where the quality of demonstrations varies. The success of this approach in multi-agent cooperative tasks underscores its potential for broader applications, including scenarios where designing explicit reward functions is impractical or where expert demonstrations are scarce or of mixed quality.

5.2 Limitation

While our approach demonstrates significant improvements in IL through reward optimization, there are some limitations that should be considered. The method may struggle in environments where the reward signal is sparse and difficult to recover through preference signals. The success of our approach also depends on having a sufficient number of trajectories to adequately cover the state space. If the dataset lacks diversity or is too limited in scope, the reward model might not generalize well In such cases, the reward model might not effectively capture the underlying reward structure, which can limit its ability to filter trajectories accurately and, consequently, results in less effective learning.

5.3 Future Study

One promising direction is the development of more sophisticated reward models that can better handle the variability and potential biases in human feedback. Incorporating techniques such as active learning, where the model selectively queries for feedback on the most uncertain or critical trajectories, could enhance the robustness and reliability of the learning process.

Additionally, we see significant opportunities to expand the application of this method to more complex and dynamic environments, particularly in the context of Large Language Models (LLMs). For instance, in scenarios where LLMs are used to generate responses in a multi-agent dialogue system, our reward model could be adapted to filter and optimize the training data based on user preferences and feedback. This would ensure that the LLM-generated responses align more closely with user expectations and improve over time. Seeking for an application of our approach in LLM environments could be a valuable future work.

References

Daniel S. Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations, 2019. URL https://arxiv.org/abs/1904.06387.

Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008. doi: 10.1109/TSMCC.2007.913919.

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2017. URL https://arxiv.org/abs/1706.03741.

Jakob Foerster, Nantas Nardelli, Gregory Farquhar, Triantafyllos Afouras, Philip H. S. Torr, Pushmeet Kohli, and Shimon Whiteson. Stabilising experience replay for deep multi-agent reinforcement learning, 2018. URL https://arxiv.org/abs/1702.08887.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning, 2016. URL https://arxiv.org/abs/1606.03476.

Dom Huh and Prasant Mohapatra. Multi-agent reinforcement learning: A comprehensive survey, 2024. URL https://arxiv.org/abs/2312.10256.

Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in atari, 2018. URL https://arxiv.org/abs/1811.06521.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel : Model-based offline reinforcement learning, 2021. URL https://arxiv.org/abs/2005.05951.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning, 2020. URL https://arxiv.org/abs/2006.04779.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020. URL https://arxiv.org/abs/2005.01643.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments, 2020. URL https://arxiv.org/abs/1706.02275.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Dean A. Pomerleau. Alvinn: an autonomous land vehicle in a neural network. In *Proceedings of the 1st International Conference on Neural Information Processing Systems*, NIPS'88, page 305–313, Cambridge, MA, USA, 1988. MIT Press.

Stephane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning, 2011. URL https://arxiv.org/abs/1011.0686.

Alexander Rutherford, Benjamin Ellis, Matteo Gallici, Jonathan Cook, Andrei Lupu, Gardar Ingvarsson, Timon Willi, Akbir Khan, Christian Schroeder de Witt, Alexandra Souly, Saptarashmi Bandyopadhyay, Mikayel Samvelyan, Minqi Jiang, Robert Tjarko Lange, Shimon Whiteson, Bruno Lacerda, Nick Hawes, Tim Rocktaschel, Chris Lu, and Jakob Nicolaus Foerster. Jaxmarl: Multi-agent rl environments in jax. 2023.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning, 2017.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation, 2018. URL https://arxiv.org/abs/1805.01954.

Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *J. Mach. Learn. Res.*, 18:136:1–136:46, 2017. URL https://api.semanticscholar.org/CorpusID:703818.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization, 2020. URL https://arxiv.org/abs/2005.13239.

Natalia Zhang, Xinqi Wang, Qiwen Cui, Runlong Zhou, Sham M. Kakade, and Simon S. Du. Multi-agent reinforcement learning from human feedback: Data coverage and algorithmic techniques, 2024. URL https://arxiv.org/abs/2409.00717.