Methods to Represent and Fit Utility Functions in Preference-based Reinforcement Learning

Weinan Qian
Yuanpei College
Peking University
ypqwn@stu.pku.edu.cn

Abstract

Utility holds paramount significance in comprehending human decision-making processes, with the principle of maximum expected utility widely embraced across various domains, including Preference-based Reinforcement Learning (PbRL). This approach addresses numerous challenges within Reinforcement Learning (RL), particularly those concerning the reward function it employs, by adopting the concept of human utility. By utilizing a utility function that doubles as the reward function, PbRL empowers agents to make more rational decisions that closely align with human intentions. Therefore, it becomes imperative to explore methods for representing and adapting the utility function within PbRL. This paper aims to introduce two distinct approaches for representing and learning the utility function, leveraging collected human preferences between trajectory pairs. Additionally, it delves into an analysis of their respective merits and limitations within practical scenarios.

1 Introduction

Utility serves as a pivotal factor in human decision-making processes across various academic domains, spanning philosophy, economics, and game theory. The prevailing hypotheses in these disciplines often propose that agents opt for rational decisions by aiming to maximize their expected utility, as articulated in the principle of maximum expected utility [19]. In specific contexts, for instance when we discuss a plant's profit or an action's reward, utility can be explicitly quantified as *cardinal utility*, enabling discussions on marginal utility and facilitating the derivation of optimal choices for agents. However, in the majority of cases, we can only obtain *ordinary utility*, where utility is only discernible through preference pairs of two choices, rendering its explicit form elusive. Consequently, the effective representation and alignment with human utility emerge as crucial imperatives across numerous research domains.

Reinforcement Learning (RL) has found widespread applications across diverse domains such as games and robotics [9, 11]. Nevertheless, the success of RL is heavily contingent on the initial knowledge embedded within the reward function [17]. Reward engineering poses substantial challenges, including intricate issues like reward hacking [3], reward shaping [12], and infinite rewards [18]. Furthermore, relying solely on absolute reward values often leads to poor robustness, as minor alterations in rewards may significantly impact the learned policy, resulting in markedly different decisions with certain probabilities. In addressing these challenges, Preference-based Reinforcement Learning (PbRL) emerges as a solution. This paradigm facilitates learning from non-numerical feedback, representing human ordinary utility in sequential domains. Its primary objective is to alleviate the aforementioned complexities, making RL more adaptable to a broader spectrum of tasks and accessible to non-expert users [17]. By doing so, it endeavors to extend the applicability of RL in diverse real-world scenarios.

Consequently, it holds paramount significance to delve into discussions regarding viable methodologies for representing and fitting utility functions based on human preferences within the domain

of PbRL. To facilitate these discussions in subsequent sections, we will commence by providing a concise overview of the PbRL algorithms' overall pipeline in Sec. 2. Subsequently, in Sec. 3, we will present two distinct methods for representing and learning human utility, as well as conduct an analysis, scrutinizing their respective advantages and limitations.

2 Pipeline of Preference-based Reinforcement Learning

Initially, we offer a succinct introduction to the workflow within PbRL. In addition, this section includes the establishment of essential mathematical notations, intended for utilization in Sec. 3.

The algorithm comprises a policy network $\pi : \mathcal{S} \to \mathcal{A}$, where $\mathcal{S}$ represents the state space and $\mathcal{A}$ denotes the action space, and an utility function U as the guiding criterion for policy updates, which maps a trajectory $\tau = \{(s_i, a_i)\}_{i=0}^{n} \in (\mathcal{S} \times \mathcal{A})^{n+1}$ to a real numerical value. The utility function U is parameterized by a set of learnable parameters, and undergoes continuous updates throughout the training process, refining its fit to human preferences with greater precision.

A typical PbRL algorithm conducts the following three steps iteratively to update the policy network and utility function [6, 17]:

1. The policy π interacts with the environment to produce a set of trajectories $\{\tau_1, \cdots, \tau_k\}$.
2. Randomly generate one pair of trajectories (τ_1, τ_2) as a query of preference. The preference relation will be evaluated by human comparison. Whether to allow weak preference, *i.e.*, allowing the indifference between τ_1 and τ_2, depends on the actual settings in different tasks.
3. Leverage the preference relation obtained in Step 2 to update the parameters of the utility function, and utilize this refined utility function that served as the reward function to update the parameters of π.

In the realm of improving the performance of PbRL, the generation of trajectories and preference queries offers several beneficial methods. Strategies such as directed homogeneous exploration [14], heterogeneous exploration [1], and user-guided exploration [20] significantly contribute to generating more sensible trajectories for enhanced human evaluation. Similarly, for preference query generation, methods like exhaustive [5], greedy [2], and interleaved approaches [16] exist, each offering distinct advantages. While these generation methodologies are often crucial in practical PbRL implementations, we omit these specific details since they are less directly related to the focus of our discussion. Instead, we will concentrate on elucidating the presentation and learning process of the utility function in Sec. 3.

3 Methods to Represent and Fit Utility Functions

Next, we introduce two methods for representing and fitting utility functions: Bradley-Terry models [4] and linear utility functions [17]. Both of these representations offer simplicity in accommodating human preferences owing to the explicit form of their loss functions. However, they possess distinct advantages and limitations compared to an alternate method, which will be succinctly analyzed in the subsequent subsections.

3.1 Bradley-Terry Model

Bradley-Terry Model offers an intuitive method to measure the probability of preference relation using the utility function. In this model, the predicted probability that τ_1 is preferred to τ_2, *i.e.*, $\tau_1 \succ \tau_2$, is defined as the softmax of two utility values:

$$\Pr(\tau_1 \succ \tau_2) = \frac{\exp U(\tau_1)}{\exp U(\tau_1) + \exp U(\tau_2)}.$$

In order to more effectively represent the utility, this method decomposes the overall utility on an entire trajectory to the utility term of the state and action in each step:

$$U(\tau) = \sum_{i=0}^{n} \hat{U}(s_i, a_i), \text{ where } \tau = \{(s_i, a_i)\}_{i=0}^{n}.$$

In practice, τ will be replaced by its randomly sampled consecutive segment $\sigma = \{(s'_i, a'_i)\}_{t=0}^{m-1}$ with a fixed length m This substitution aims to enhance human evaluation since the original trajectory might be excessively long for precise preference assessment. Consequently, the predicted probability will be

$$\Pr(\tau_1 \succ \tau_2) = \frac{\exp \sum_{t=0}^{m-1} \hat{U}(s_t^{(1)}, a_t^{(1)})}{\exp \sum_{t=0}^{m-1} \hat{U}(s_t^{(1)}, a_t^{(1)}) + \exp \sum_{t=0}^{m-1} \hat{U}(s_t^{(2)}, a_t^{(2)})}.$$

Then, the formulation of the loss function employed for updating the utility function $\hat{U} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$is defined as follows: For a pair of trajectories (τ_1, τ_2), an expert is tasked with evaluating their preference. The outcome is recorded as a probability distribution μ over $\{1, 2\}$. If $\tau_i \succ \tau_j$ then $\mu_i = 1,\ \mu_j = 0$. In cases where the expert expresses indifference between the two trajectories, *i.e.*, $\tau_i \sim \tau_j$, both μ_1 and μ_2 are assigned a value of 0.5. If the expert deems the trajectories as incomparable, the pair is skipped and not included in the utility updating process. Utilizing this ground truth, the loss function is defined as the cross-entropy loss between the model predictions and the actual human labels:

$$\mathcal{L}(\hat{U}) = -\left[\sum_{\tau_1, \tau_2, \mu} \mu_1 \log \Pr(\tau_1 \succ \tau_2) + \mu_2 \log \Pr(\tau_2 \succ \tau_1)\right].$$

This approach offers a concise method to represent and optimize the utility function through the minimization of cross-entropy loss. It involves the utilization of a neural network to express the utility of a state-action pair, leveraging the distinctive characteristics of both states and actions. This approach grants the flexibility to adapt utility representations according to varying state-action dynamics. However, this flexibility introduces challenges in identifying an appropriate model that effectively represents utility amidst dynamic scenarios, often entailing a balance between variance and bias. Empirical findings from experiments [6] highlight the potential improvements achievable in specific environments using this method. Nevertheless, these improvements exhibit instability, and the selection of different concrete implementation methods (e.g., variations in query generation approaches) leads to disparate outcomes across diverse environments. This underscores the necessity for meticulous human oversight to prevent suboptimal performances when employing this method.

3.2 Linear Utility Function

Many researchers in the field of PbRL have employed linear utility functions for their simplicity in representation and training. This approach leverages a trajectory feature vector $\psi(\tau) \in \mathbb{R}^d$ to provide a more comprehensive description of a trajectory. The utility of τ is then represented as $U(\tau) = \theta^\top \psi(\tau)$, where $\theta \in \mathbb{R}^d$ is a learnable vector. To effectively fit such a linear utility function, the loss function $\mathcal{L}$ can be defined as a weighted sum of the pairwise disagreement loss L:

$$\mathcal{L}(\theta, \zeta) = \sum_{i=1}^{|\zeta|} \alpha_i L(\theta, \zeta_i),$$

where ζ represents the set of all the preference relations $\{\zeta_i\}_{i=1}^{|\zeta|}$, and α_i signifies the weight or importance attributed to ζ_i which varies for different choices of L. Each preference relation ζ_i comprises two trajectories τ_{i1}, τ_{i2}, demonstrating a strong preference relation where $\tau_{i1} \succ \tau_{i2}$. This implies that $U(\tau_{i1}) = \theta^\top \psi(\tau_{i1}) > U(\tau_{i2}) = \theta^\top \psi(\tau_{i2})$. The pairwise disagreement loss is predicated on the difference in utilities between these trajectories:

$$d(\theta, \zeta_i) = \theta^\top (\psi(\tau_{i1}) - \psi(\tau_{i2})).$$

Numerous options exist for $L(\theta, \zeta_i)$. For examples, certain algorithms [13, 15] define $L(\theta, \zeta_i)$ as the hinge loss $L(\theta, \zeta_i) = \max\{0, 1 - d(\theta, \zeta_i)\}$, a formulation conducive to optimization by Support Vector Machine (SVM) ranking algorithms [8]. Other approaches model the likelihood function $p_\theta(\zeta_i)$ and seek to minimize the negative log-likelihood, *i.e.*, $L(\theta, \zeta_i) = -\log(p_\theta(\zeta_i))$. The selection of the likelihood function is also flexible. For instance, Kupcsik et al. [10] leveraged cumulative likelihood:

$$p_\theta(\zeta_i) = \Phi\left(\frac{d(\theta, \zeta_i)}{\sqrt{2}\sigma_p}\right) = \Phi\left(\frac{\theta^\top(\psi(\tau_{i1}) - \psi(\tau_{i2}))}{\sqrt{2}\sigma_p}\right),$$

where $\Phi(\cdot)$ represents the Cumulative distribution function (CDF) of a standard normal distribution, and σ_p denotes a noise term that accommodates feedback noise. The pursuit of minimizing the negative log-likelihood, particularly when adopting the likelihood in a cumulative style, has been established as a convex optimization task, particularly when utilizing preference feedback exclusively [7]. Consequently, this approach leads to a unique global minimum, rendering the problem more deterministic in nature.

This approach offers a more straightforward means of representing human utility. The linear assumption simplifies representation forms and streamlines optimization, often leading to convex optimization problems when seeking the optimal utility function parameters. Additionally, computing the utility function is more computationally efficient. However, this simplicity in form might compromise its generalization capabilities. Models based on linear assumptions may have a substantially smaller VC dimension compared to neural network-based utility representations introduced in Sec. 3.1. Moreover, designing an effective method to accurately represent a trajectory using a feature function $\psi(\cdot)$ presents considerable challenges within this framework.

4 Conclusion

In this paper, we present two methods aimed at effectively representing and fitting human utilities within the framework of PbRL: the Bradley-Terry model and the linear utility function. Both approaches offer intuitive means of creating concise utility functions that align well with human preferences. The former method, leveraging neural networks, possesses higher potential for generalization but tends to be more challenging to train and doesn't consistently outperform traditional RL algorithms. The latter method employs a simpler linear function to capture human preferences across different trajectories, simplifying both its representation and training processes. However, this method also exhibits shortcomings such as limited generalization ability and the complexity of designing features for diverse trajectories. While advancements in PbRL are evident, achieving definitively satisfactory results remains a distant goal. We remain optimistic that researchers in this field will eventually uncover a method capable of striking a balance between enhanced generalization capabilities and mitigated training difficulties.

References

[1] Riad Akrour, Marc Schoenauer, and Michèle Sebag. Preference-based policy learning. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD-11)*, 2011. 2

[2] Riad Akrour, Marc Schoenauer, Michèle Sebag, and Jean-Christophe Souplet. Programming by feedback. In *International Conference on Machine Learning (ICML)*, 2014. 2

[3] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. In *arXiv preprint arXiv*, 2016. 1

[4] Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. 2

[5] Róbert Busa-Fekete, Balázs Szörényi, Paul Weng, Weiwei Cheng, and Eyke Hüllermeier. Preference-based reinforcement learning: evolutionary direct policy search using a preference-based racing algorithm. *Machine Learning*, 97(3):327–351, 2014. 2

[6] Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 2, 3

[7] Wei Chu and Ghahramani. Preference learning with gaussian processes. In *International Conference on Machine Learning (ICML)*, 2005. 4

[8] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-02)*, 2002. 3

[9] Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013. 1

[10] Andras Kupcsik, David Hsu, and Wee Sun Lee. Learning dynamic robot-to-human object handover from human feedback. In *Proceedings of the 17th International Symposium on Robotics Research (ISRR-15)*, 2015. 3

[11] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. 1

[12] Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning (ICML)*, 1999. 1

[13] Thomas Philip Runarsson and Simon M. Lucas. Imitating play from game trajectories: Temporal difference learning versus preference learning. In *IEEE Conference on Computational Intelligence and Games (CIG-12)*, 2012. 3

[14] Aaron Wilson, Alan Fern, and Prasad Tadepalli. A bayesian approach for policy learning from trajectory preference queries. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012. 2

[15] Christian Wirth and Johannes Furnkranz. An optimization approach to rough terrain locomotion. In *International Conference on Robotics and Automation (ICRA)*, 2012. 3

[16] Christian Wirth, Johannes Furnkranz, and Gerhard Neumann. Model-free preference-based reinforcement learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2016. 2

[17] Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18 (136):1–46, 2017. 1, 2

[18] Yufan Zhao, Michael R. Kosorok, and Donglin Zeng. Reinforcement learning design for cancer clinical trials. *Statistics in medicine*, 28(26):3294–3315, 2009. 1

[19] Yixin Zhu, Tao Gao, Lifeng Fan, Siyuan Huang, Mark Edmonds, Hangxin Liu, Feng Gao, Chi Zhang, Siyuan Qi, Ying Nian Wu, et al. Dark, beyond deep: A paradigm shift to cognitive ai with humanlike common sense. *Engineering*, 6(3):310–345, 2020. 1

[20] Matt Zucker, J. Andrew Bagnell, Christopher G. Atkeson, and James Kuffner. First steps towards learning from game annotations. In *Proceedings of the ECAI Workshop on Preference Learning: Problems and Applications in AI*, 2012. 2