2022 IEEE International Conference on
Systems, Man, and Cybernetics (SMC)

Advances in Preference-based Reinforcement Learning: A Review

1st Youssef Abdelkareem
Electrical and Computer Engineering
University of Waterloo
Waterloo, Canada
yafathi@uwaterloo.ca

2nd Shady Shehata
Mohamed bin Zayed University
of Artificial Intelligence
Abu Dhabi, UAE
shady.shehata@mbzuai.ac.ae

3rd Fakhri Karray
Electrical and Computer Engineering
University of Waterloo
Waterloo, Canada
karray@uwaterloo.ca

2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC) | 978-1-6654-5258-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/SMC53654.2022.9945333

***Abstract*—Reinforcement Learning (RL) algorithms suffer from the dependency on accurately engineered reward functions to properly guide the learning agents to do the required tasks. Preference-based reinforcement learning (PbRL) addresses that by utilizing human preferences as feedback from the experts instead of numeric rewards. Due to its promising advantage over traditional RL, PbRL has gained more focus in recent years with many significant advances. In this survey, we present a unified PbRL framework to include the newly emerging approaches that improve the scalability and efficiency of PbRL. In addition, we give a detailed overview of the theoretical guarantees and benchmarking work done in the field, while presenting its recent applications in complex real-world tasks. Lastly, we go over the limitations of the current approaches and the proposed future research directions.**

***Index Terms*—Preference-based Reinforcement Learning, Theoretical Guarantees, Benchmarking.**

I. Introduction

Reinforcement learning (RL) is a sub-field of machine learning that has been widely implemented in many applications such as robot control [1], games [2], and medical domains [3]. A learning agent continuously interacts with an environment by doing actions and receives feedback signals (rewards) with the target of maximizing the cumulative reward by the end of the interaction phase. The reward received is assumed to be numerically generated from a specific reward function. The main issue that arises is the high sensitivity of the performance to the design of the reward function. Specifically, one of the challenges in reward engineering is called Reward Hacking which occurs when the agent comes up with ways to maximize the cumulative rewards without executing the intended task [4]. Reward Shaping is also another problem that involves finding the optimal balance between providing rewards that direct the agent to the final goal (extrinsic motivation) while guiding them to also do the intended task at the same time (intrinsic motivation) [5].

The field of preference-based reinforcement learning (PbRL) promises a solution to the aforementioned problems. It revolves around providing the agent with non-numeric reward signals in the form of pairwise preferences rather than absolute rewards. This shift in approach broadens the scope of RL algorithms to non-expert users where accurate reward engineering is no longer required. In this work, we present a coherent framework for PbRL, summarized in Figure 1. Our framework is an extension of the one proposed by [6] where we include the most recent advances in PbRL.

We start with a formal definition of the problem in Section II, followed by the different design choices in Section III. We then go over the PbRL algorithms with theoretical guarantees in Section IV and the available benchmarking frameworks in Section V. Applications of PbRL in the Natural Language Processing (NLP) domain are presented in Section VI. Lastly, we conclude by analyzing the shortcomings of the surveyed methods and propose future research directions in Section VII.

II. Problem Formulation

In PbRL algorithms, we are trying to solve the traditional RL problems using preferences between pairs of states, actions, or trajectories rather than absolute numerical rewards. The Markov Decision Process (MDP) for Preferences (MDPP) [6] is represented as a sextuple $(S, A, \mu, \delta, \gamma, \rho)$. Similar to the original MDP, S and A denote the state and action spaces that could be either discrete, with sizes $|S|$ and $|A|$, or continuous. $\delta(s'|s, a)$ represents a stochastic state transition model, while γ is the discount factor $\in [0, 1)$, $\mu(s)$ is the initial state distribution and h is the horizon length in finite-horizon settings. Trajectories (τ) define a sequence of state-action pairs and $\pi(a|s)$ represents the policy. The difference between MDP and MDPP is that a preference relation over trajectories $\tau_1 \succ \tau_2$, where τ_1 is more preferred than τ_2, is received by the agent instead of the numeric reward signal $r(s, a)$. $\rho(\tau_1 \succ \tau_2)$ denotes the probability that a preference relation holds. Regarding the objective, we would like the agent to learn an optimal policy π^* that generates trajectories that satisfy the set of all preference relations ζ received from the expert during training. Formally, the objective of PbRL for a single preference relation $\tau_i \succ \tau_j \in \zeta$ is

$$\tau_i \succ \tau_j \leftrightarrow \pi^* = argmax_\pi(Pr_\pi(\tau_i) - Pr_\pi(\tau_j)), \quad (1)$$

$$Pr_\pi(\tau) = \mu(s_0) \prod_{t=0}^{|\tau|} \pi(a_t|s_t)\delta(s_{t+1}|s_t, a_t), \quad (2)$$

which assumes that the optimal policy is the one that maximizes the difference between the probabilities of obtaining the more preferred trajectory (dominating) and the less preferred one (dominated).

III. Design Choices

A. Preference Type Design Choices

In the literature, the types of preferences that are received from experts could be divided into action, state, and trajectory preferences.

1) Action Preferences: Action preferences reduce the preference relation to the comparison of a pair of actions for the same state, where $a_1 \succ_s a_2$ denotes that action a_1 is more preferred than action a_2 for state s.

Utilizing action preferences that optimize short-term rewards would be hard as they're only valid for a given state. [7] uses action-based preferences that deal with long-term optimality and evaluate the relation $a_1 \succ_s a_2$ by doing a roll-out for trajectories starting by state s, doing action a, and following the estimate of the policy to get the expected returns of each action. Action preferences with long-term optimality are considered demanding for experts since the long-term outcome should be known.

2) State Preferences: Regarding state-preferences, a relation $s_1 \succ s_2$ indicates that s_1 is preferred over s_2. Since those relations correspond to segments of the state space, they give more information compared to action preferences and are less demanding to the expert since no comparisons between actions are needed. [8], [9] follow long-term optimality by proposing that selecting the most preferred successor state for every state could be a viable solution in their long-term state-based setting. [10] uses short-term state preferences and mitigates their issues by trying to estimate a cost function for every state using the support vector ranking approach [11].

3) Trajectory Preferences: The most common type of preference relations are trajectory preferences where $\tau_1 \succ \tau_2$ indicates that trajectory τ_1 dominates over τ_2. Such preferences are desirable since they can be easily evaluated by experts by assessing the full trajectories and their results. Most of the methods presented in the rest of this paper use trajectory preferences. A general challenge is relating the final preference over trajectories to their most relevant states and actions.

B. Learning Problem Design Choices

There have been several proposed approaches to use the preference feedback received from the expert to optimize the policy. Our main focus will be on approaches that directly learn a policy distribution, or estimate a utility function. Other approaches like [7] learn a preference model, usually modeled as a classifier, to predict whether a preference relation holds between two actions for a given state.

1) Learning a Policy: Some methods directly derive an estimation of the policy. [12] learns a policy distribution using a Bayesian likelihood function. Specifically, they utilize the posterior distribution of policies $P(\pi|\zeta)$ given the preference relations to sample two parameterized policies which are used to generate two full trajectories τ_1, τ_2 starting from the same state. The sampled trajectories are transformed into a trajectory preference relation $(\tau_1 \succ \tau_2)$ and added to the existing buffer storing all the preference relations. The posterior policy distribution $P(\pi|\zeta)$ is represented with Bayes theorem through the multiplication of the prior distribution of the policy $P(\pi)$ with the likelihood of all the trajectory preferences $P(\tau_1 \succ \tau_2|\pi) \in \zeta$ and approximated using Markov Chain Monte Carlo Simulation [13]. The downside is that the likelihood is modeled in terms of the euclidean distance between the policy-realized trajectories which constrains the algorithm to perform properly on low-dimensional continuous state spaces only. [14], [15] mitigate this issue by not requiring a distance function in their objective. They make a direct policy comparison by presenting trajectory preference queries comparing trajectories $\tau^{\pi_1}, \tau^{\pi_2}$ generated by the two policies π_1, π_2. The sets of all preference relations and policies are used to generate a ranking between policies returning the highest-ranked policies as the optimal ones. However, their method requires a higher number of preference queries than [12], as their preferences are non-reusable, making it not feedback-efficient.

2) Learning a utility function: Learning a policy directly can be highly sample-inefficient, therefore, some methods try to estimate a surrogate utility function $U(x)$, where x denotes trajectories or state-action pairs, to extract more information from the preferences. This utility function is analogous to the reward function seen in RL, however, they are not directly related since the definition of what is optimal is dependent on the views of the expert giving the preference feedback. There are two types of utility functions used in the literature which are *linear* and *non-linear*. For both types, trajectories and state-action pairs are assumed to have a feature vector representation denoted by $\psi(\tau)$ and $\varphi(s, a)$ respectively. The general objective is to obtain the optimal policy by maximizing a *link function* d that represents the difference between the utilities of the dominating and dominated preference relation terms.

a) Linear utility functions: This type of utility is formulated for trajectories in terms of an unknown weight vector θ yielding $U(\tau) = \theta^T \psi(\tau)$. Methods that use such type of linear utility formulate the link function for a trajectory preference relation as follows $d(\theta, \tau_1 \succ \tau_2) = \theta^T(\psi(\tau_1) - \psi(\tau_2))$ and find the optimal θ that maximizes the link function for all preference relations. The optimization depends on choosing a proper loss function which differs based on the proposed methods. [16] incorporated the hinge loss that is approximated using the ranking SVM method following [17] and estimated the hand-coded feature representations $\psi(\tau)$ for the trajectories using ϵ-means clustering [18]. [19] proposes an enhancement by accounting for the inaccuracies of the expert through the introduction of a piece-wise loss. The loss represents the inaccuracies using a ridge noise model controlled by a dynamically changing hyper-parameter to consider preferences that change over time.

b) Non-linear utility functions: There are different ways to introduce non-linearity in the utility functions. Our focus is on recent methods that utilize deep neural networks (DNNs) as a non-linear representation for the utility. Those methods can be categorized based on using online or offline RL algorithms in their formulation.

b.1) DNNs with Online RL: [20] is the first method to represent the parameters θ of the utility function $U_\theta(s, a)$ with a DNN. Due to their high representational capacity, DNNs opened the door to experiment on more challenging robotic [21] and Atari [22] tasks. In addition, the experts are queried with trajectory segments (σ) instead of full trajectories (τ) which reduces the effort done by them compared to [16]. The utility of the segments are sum-decomposable in terms of the non-linear state-action utilities yielding $U_\theta(\sigma) = \sum_i U_\theta(s_i, a_i)$. The link function for a trajectory segment preference relation $d(\theta, \sigma_1 \succ \sigma_2)$ is modeled as a probability function following the Bradley-

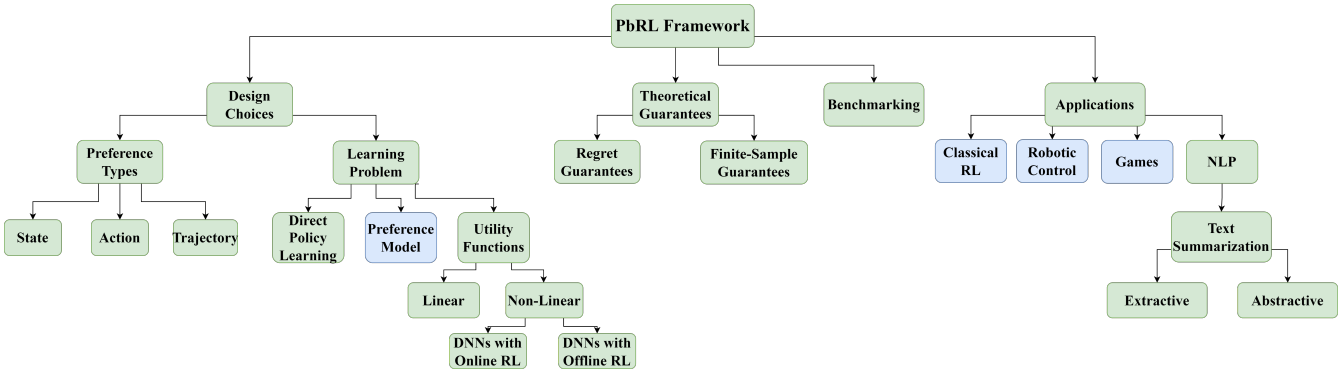

Fig. 1. Hierarchical structure of our PbRL framework. Topics in green are surveyed in this work, while the ones in blue are not within our scope. "DNNs" denotes Deep Neural Networks and "NLP" denotes Natural Language Processing.

Terry model [23] and optimized using the cross-entropy loss with the labels being the preference feedback received from the experts.

$$d(\theta, \sigma_1 \succ \sigma_2) = \frac{e^{U_\theta(\sigma_1)}}{e^{U_\theta(\sigma_1)} + e^{U_\theta(\sigma_2)}} \tag{3}$$

One could look at the approach of [20] as a model-free RL algorithm, where the state-transition dynamics are unknown and the reward model corresponds to the utility function $U_\theta(s, a)$ which mimics the reward received from the environment. Based on that, they utilize online policy gradient approaches like A2C [24] and TRPO [25] to optimize the policies while using the learned utility as an estimate of the reward received from the environment with each interaction. This online learning setting needs many interactions with the environment and numerous preference queries to reach the optimal policy. [26] focus on adding a supervised human-preference estimator that learns to mimic the human preferences to decrease the number of preference queries presented to the expert by depending on the estimated preferences for some samples. Furthermore, they allow the expert feedback to be any continuous value in the range [0,1] to deal with weak preference relations between trajectory segments. Both contributions enhanced the performance and efficiency compared to [20].

b.2) DNNs with Offline RL: Offline methods train on examples saved in a buffer to reduce the interactions with the environment. [27] proposed utilizing the famous Soft Actor-Critic (SAC) offline RL algorithm [28] with the same utility function settings used by [20] to handle the sample and feedback inefficiency problems. To do that, they had to account for the dynamically changing estimate of the utility function by relabelling all the agent's past sequences with the corresponding new utility values whenever the utility function is updated with new preferences. Additionally, they tackled the limited diversity in the initial preference queries given to the expert, which happens due to the random initialization of the policy, by including an unsupervised pre-training step. The main limitation of this approach is the high cost of relabelling all the preferences stored in the buffer. [29] build on the offline learning approach of [27] by introducing a pseudo-labeling procedure on the existing unlabeled preference relations stored in the offline dataset. Specifically, an unlabeled preference relation $\sigma_1 \succ \sigma_2$ is predicted to hold if the probability estimated with the link function from (3) exceeds 50% and doesn't hold otherwise. Only confident labels with high probability are used. Moreover, they introduce the first usage of data augmentation approaches in PbRL to enhance the regularization and feedback efficiency even more. Such augmentation is done by randomly cropping labeled segments under the assumption that the preference label should be consistent with the re-scale and shift of the individual segments.

IV. Theoretical Guarantees

There is a current research target aiming to develop novel PbRL algorithms that are tractable for theoretical analysis. Those algorithms focus on reaching either regret or finite-sample guarantees which will be discussed in detail in this section.

A. Regret Guarantees

Regret denotes the difference between the current expectation of total rewards and the maximum rewards generated by the optimal policy. Providing theoretical guarantees on the bounds of the regret has been an important aim for various RL and bandit approaches [30], [31]. DPS [32] is the first paper to propose a PbRL algorithm with solid regret guarantees. They base their algorithm on the Thompson Sampling algorithm [33] while formulating Bayesian regret bounds by borrowing the information-theoretic concepts from [34]. DPS uses trajectory preferences within a model-based approach. They assume having a non-linear utility function $U(s, a)$ for state-action pairs to estimate the reward model. Both the reward and transition dynamics models are represented as Bayesian posterior distributions. Similar to [20], the trajectory utilities are a summation of state-action utilities yielding $U(\tau) = \sum_i U(s_i, a_i)$. The link function is linear with $d(\theta, \tau_1 \succ \tau_2) = U(\tau_1) - U(\tau_2)$ where θ represents the parameters of the distribution of the utility function. Their algorithm samples two distinct pairs of transition and utility models from their distributions and applies value iteration to yield the two corresponding policies. The transition and utility distributions are then updated using the queried preferences history present in a buffer. By proving the asymptotic convergence of the transition and reward models, they were able to reach an asymptotic sublinear regret rate of $|S|\sqrt{2|A|Nh\log|A|}$, where N is the number of iterations of the algorithm.The main limitation of [32] is the exponential complexity in the time horizon h for the

asymptotic convergence of the reward and transition models. [35] proposed a more general PbRL algorithm with regret guarantees by assuming that the underlying utility function represents non-Markovian rewards. The utility function is linear in terms of the trajectory features with dimension d, such that $U_\theta(\tau) = \theta^T \psi(\tau)$. They utilize the same link function in (3) in terms of full trajectories and add an extra L2 regularization term on θ. The foundation of their approach is based on bounding θ under a parameter Q with the rescaling concepts used in [36]. Consequently, they propose two algorithms that assume having known and unknown transition models achieving near-optimal regret bounds of $O'(Qd\log(N/\delta)\sqrt{N})$, with a probability of at least $1-\delta$, and $O'((\sqrt{d}+h^2+|S|)\sqrt{dN}+\sqrt{|S||A|Nh})$, respectively. The notation O' hides any logarithmic factors in the variables.

B. Finite-Sample Guarantees

[37] is the only PbRL algorithm in the literature to derive finite-sample guarantees in terms of the number of preferences queries and the number of interactions with the environment (number of steps). Trajectory preferences are used under the observation that those preferences need to be noisy to derive a unique optimal policy. The target of the algorithm is to efficiently obtain an ϵ-optimal policy. They utilize black-box PAC-Dueling Bandits (P-DB) algorithms like [38] and [39] to make policy comparisons based on the collected trajectory preferences. Specifically, one of their proposed algorithms (PEPS) explores the state space by synthesizing a reward function, similar to reward-free RL [40], and optimizes it using a tabular RL algorithm (EULER; [41]). The P-DB algorithm then generates action queries during learning which are transformed into trajectory preference queries by rolling out the trajectories with the current policy estimated with EULER. If the P-DB algorithm requires no target accuracy in advance, the PEPS method reaches an ϵ-optimal policy with a step complexity of $O(\frac{h^2|S|^2|A|\iota}{\epsilon^2}+\frac{|S|^4|A|h^3\iota^3}{\epsilon})$ and a preference query complexity of $O(\frac{h|S|^2|A|\iota}{\epsilon^2})$, where ι denotes the log factors. The main limitations of this approach are the sub-optimal sample complexity and the dependency on the guarantees of the underlying P-DB algorithm which only hold under some restrictions on the structure of the preferences.

V. Benchmarking

There has been a large focus in the literature to create benchmarks for various RL domains such as Offline RL [42], and Safe RL [43]. [44] is the first paper to propose a benchmark for the consistent evaluation of PbRL algorithms without relying on expensive human feedback. To achieve that, they simulate an expert providing preference based on the total sum of the ground-truth rewards while explicitly accounting for the human errors. Modeling the stochasticity of the simulated expert preferences is achieved by following the link function formulation in (3) to model the preference probability of segments in terms of ground-truth rewards yielding

$$P(\sigma_1 \succ \sigma_2) = \frac{e^{\beta\sum_{i=0}^{T}\gamma^{T-i}r(s_i^1,a_i^1)}}{e^{\beta\sum_{i=0}^{T}\gamma^{T-i}r(s_i^1,a_i^1)}+e^{\beta\sum_{i=0}^{T}\gamma^{T-i}r(s_i^2,a_i^2)}}, \tag{4}$$

where β controls the degree of expert determinism and γ is a discount factor used to model the short-sightedness by giving higher weights to recent rewards. In addition, incomparable queries are skipped if the total segment reward is less than a threshold. The simulated expert is allowed to make mistakes by flipping (4) randomly with probability ϵ.

[44] quantitatively measures the performance of PbRL algorithms by normalizing the average predicted returns for the true reward. In their experiments, they focus on comparing the performance of the state-of-the-art deep PbRL algorithms with non-linear utility functions [20], [27] explained in Section III-B2. The tasks used have absolute-valued states and dense rewards and are taken from the DeepMind Control Suite [45] and the Meta-world benchmark [46]. The results of the experiments indicate that both [20] and [27] only perform well in cases where the expert doesn't make errors while the performance significantly degrades once the expert preferences become more stochastic.

VI. Application Areas

PbRL algorithms have been used in practical applications that involve robot teaching tasks, board games, and others. We refer the reader to a detailed overview of those application domains discussed in [6]. In this survey, we focus on the application of PbRL algorithms to the text summarization task in Natural Language Processing (NLP) which predicts a qualitative summary by extracting the important information in an input piece of text. Most of the methods base their algorithms on supervision using ground-truth summaries generated by experts [47]. Due to the high cost of collecting large amounts of reference summaries, there was a shift to reward-based RL approaches [48]. However, the rewards used in those methods are mainly based on ROUGE scores which do not directly correlate with the quality of the summaries as judged by humans [49]. This consequently motivated exploring preferences instead of rewards to achieve a higher correlation with the quality of the summaries as judged by humans The two main categories of the task are *extractive* and *abstractive* summarization.

A. Extractive summarization

Summaries of this type are built by extracting specific sentences from the original text. [50] is one of the initial approaches utilizing PbRL algorithms for the task. In their MDP formulation, the state is the summary made so far and actions correspond to possible sentences to be extracted. They follow a linear utility-based PbRL approach discussed in Section III-B2 with the link function formulation in (3). Similar to [20], the utility resembles the reward function and is used to optimize the policy using a simple policy gradient RL algorithm. [51] proposed enhancements by including a better preference querying approach and a neural policy learning method with temporal differences.

B. Abstractive Summarization

Summaries of this type are built by inducing the underlying ideas and concisely stating them. Its high degree of subjectivity makes it much more challenging than the extractive task. [52] capitalized on the large capacity of pre-trained transformer models to try solving the abstractive task effectively. They use two pre-trained transformer

models to represent the non-linear utility function and the underlying policy that generates summaries given the text input. The policy model is optimized using the Proximal Policy Estimation (PPO) algorithm and the utility function is periodically trained with the online collection of preferences. Their summarization results turned out to be highly extractive due to the bias of the experts to prefer copied summaries and the online querying made it hard to ensure the quality of the expert responses. [53] solved the previous problems by shifting to a batch setting for preference collection instead of a fully online one, while consistently communicating with experts to ensure more relevant preference responses which lead to more abstractive results. [54] shifted the application from summarization of moderate size English text to whole books by recursively decomposing the books into smaller sections to be easily evaluated by human experts.

VII. Analysis & Future Work

It was seen how the integration of deep RL algorithms into the PbRL framework by [20] revolutionized the scalability of PbRL to complex tasks and motivated a large amount of follow-up work that enhanced the efficiency successfully. Some concurrent work [35] managed to put solid foundations of theoretical guarantees for PbRL algorithms proving their robustness, while others proposed a benchmarking tool [44] for consistent and fair comparison of PbRL algorithms. Preferences also proved their effectiveness in achieving human-desired performance in challenging real-world tasks such as text summarization [53]. However, open problems still exist in the current literature work along with potential future research directions.

A. Formulation and performance of PbRL algorithms

State-of-the-art methods perform poorly whenever the experts make mistakes in their preferences [44]. A possible solution could be to explicitly consider the stochasticity of the expert while designing the link functions. Also, to further enhance the feedback and sample efficiency, one could experiment with using model-based approaches like [32] in more challenging environments [20]. Additionally, not enough research has been done on handling incomparable trajectories, that could have contradicting preferences, to get Pareto-optimal policies. This could be addressed by learning a set of utility functions that are non-scalar (multi-dimensional) and utilizing multi-objective RL methods [55] to get the corresponding Pareto-optimal policies. A limiting assumption in most work is the fact that all the trajectories in the preferences start at the same state. Such constraint could be relieved by utilizing the advantage function to represent the utility in terms of the expected rewards from the different initial states. Some methods [53], [56] utilized extra supervision signals like expert demonstrations to supplement the low amount of information gained from preferences. Other signals that could be worth experimenting with are providing explanations along with the preferences or enhancing the model output by allowing the experts to directly edit them. Furthermore, concepts from representation learning [57] could be borrowed to extend PbRL to partially observable RL or sparse reward settings which require a rich representation of states.

B. Safety in PbRL

To the best of our knowledge, no prior work investigated the application of PbRL in risk-averse domains. This could be implemented by over-weighing high-risk trajectories within the preferences to prioritize learning not to prefer them. In addition, further analysis should be made on adapting PbRL methods in real-world scenarios while mitigating the possibility of malicious users incurring bias in the preferences to let the model learn undesirable behaviors.

C. Theoretical Guarantees

Future work could focus on coming up with algorithms that exhibit both regret and finite-sample guarantees at the same time. Also, extending one of the approaches with regret or finite-sample guarantees to work within complex state-action spaces or infinite horizon settings can allow for their utilization in real-world domains.

D. Applications of PbRL

The state-of-the-art summarization approach by [53] still suffers from large feedback inefficiency. An interesting direction could integrate the pseudo-labeling approach by [29] to label the existing summaries automatically without querying the expert. Moreover, there are different NLP applications involving subjective tasks which could leverage PbRL to learn human-desired behaviors and these include dialogue, machine translation, and question answering.

VIII. Conclusion

PbRL algorithms have demonstrated the possibility of utilizing human preferences as reward signals without resorting to explicit reward engineering. This survey presented the most recent advances in the field coherently while providing insights on current open problems and potential research directions. We conclude that utility-based PbRL algorithms, especially ones with non-linear formulations, provide the stepping stone to generalizing PbRL to more complex and practical application domains. However, the high cost associated with the preference feedback from experts and environment interactions creates an important research target to achieve both feedback and sample efficiency. In addition, the recent formulation of concrete regret and finite-sample guarantees initiated the tractable theoretical analysis of PbRL and future work could focus more on relaxing the assumptions of existing methods while operating under more complex environment settings. Moreover, the introduction of an open-source benchmarking tool is expected to advance the consistent evaluation of PbRL methods. Lastly, PbRL proved to reach human-desired behavior in text summarization and future work should be focused on expanding it to other subjective and challenging real-world tasks.

References

[1] J. Kober, J. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, pp. 1238–1274, 09 2013.

[2] K. Shao, Z. Tang, Y. Zhu, N. Li, and D. Zhao, "A survey of deep reinforcement learning in video games," *CoRR*, vol. abs/1912.10944, 2019. [Online]. Available: http://arxiv.org/abs/1912.10944

[3] C. Yu, J. Liu, S. Nemati, and G. Yin, "Reinforcement learning in healthcare: A survey," *ACM Comput. Surv.*, vol. 55, no. 1, nov 2021. [Online]. Available: https://doi.org/10.1145/3477600

[4] D. Amodei, C. Olah, J. Steinhardt, P. F. Christiano, J. Schulman, and D. Mané, "Concrete problems in AI safety," *CoRR*, vol. abs/1606.06565, 2016. [Online]. Available: http://arxiv.org/abs/1606.06565

[5] A. Y. Ng, D. Harada, and S. J. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999), Bled, Slovenia, June 27 - 30, 1999*, I. Bratko and S. Dzeroski, Eds. Morgan Kaufmann, 1999, pp. 278–287.

[6] C. Wirth, R. Akrour, G. Neumann, and J. Fürnkranz, "A survey of preference-based reinforcement learning methods," *J. Mach. Learn. Res.*, vol. 18, pp. 136:1–136:46, 2017.

[7] J. Fürnkranz, E. Hüllermeier, W. Cheng, and S.-H. Park, "Preference-based reinforcement learning: a formal framework and a policy iteration algorithm," *Machine Learning*, vol. 89, pp. 123–156, 2012.

[8] C. Wirth and J. Fürnkranz, "First steps towards learning from game annotations," in *Workshop Proceedings - Preference Learning: Problems and Applications in AI at ECAI 2012*, 2012.

[9] ——, "On learning from game annotations," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 7, 2015.

[10] M. Zucker, J. A. Bagnell, C. G. Atkeson, and J. J. Kuffner, "An optimization approach to rough terrain locomotion," *2010 IEEE International Conference on Robotics and Automation*, pp. 3589–3595, 2010.

[11] R. Herbrich, T. Graepel, and K. Obermayer, "Support vector learning for ordinal regression," 1999.

[12] A. Wilson, A. Fern, and P. Tadepalli, "A bayesian approach for policy learning from trajectory preference queries," in *NIPS*, 2012.

[13] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan, "An introduction to mcmc for machine learning," *Machine Learning*, vol. 50, pp. 5–43, 2004.

[14] R. Busa-Fekete, B. Szörényi, P. Weng, W. Cheng, and E. Hüllermeier, "Preference-based evolutionary direct policy search," 2013.

[15] ——, "Preference-based reinforcement learning: evolutionary direct policy search using a preference-based racing algorithm," *Machine Learning*, vol. 97, pp. 327–351, 2014.

[16] R. Akrour, M. Schoenauer, and M. Sebag, "April: Active preference-learning based reinforcement learning," *ArXiv*, vol. abs/1208.0984, 2012.

[17] R. Herbrich, T. Graepel, and C. Campbell, "Bayes point machines," *J. Mach. Learn. Res.*, vol. 1, pp. 245–279, 2001.

[18] R. O. Duda and P. E. Hart, "Pattern classification and scene analysis," in *A Wiley-Interscience publication*, 1973.

[19] R. Akrour, M. Schoenauer, M. Sebag, and J.-C. Souplet, "Programming by feedback," in *ICML*, 2014.

[20] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," in *NIPS*, 2017.

[21] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012.

[22] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents (extended abstract)," in *IJCAI*, 2015.

[23] R. A. Bradley and M. E. Terry, "Rank analysis of incomplete block designs: I. the method of paired comparisons," *Biometrika*, vol. 39, p. 324, 1952.

[24] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *ICML*, 2016.

[25] J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz, "Trust region policy optimization," *ArXiv*, vol. abs/1502.05477, 2015.

[26] Z. Cao, K. Wong, and C.-T. Lin, "Weak human preference supervision for deep reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, pp. 5369–5378, 2021.

[27] K. Lee, L. Smith, and P. Abbeel, "Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training," *ArXiv*, vol. abs/2106.05091, 2021.

[28] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *ICML*, 2018.

[29] J. Park, Y. Seo, J. Shin, H. Lee, P. Abbeel, and K. Lee, "SURF: Semi-supervised reward learning with data augmentation for feedback-efficient preference-based reinforcement learning," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=TfhfZLQ2EJO

[30] D. Russo and B. V. Roy, "Learning to optimize via posterior sampling," *Math. Oper. Res.*, vol. 39, pp. 1221–1243, 2014.

[31] O. D. Domingues, P. M'enard, M. Pirotta, E. Kaufmann, and M. Valko, "Regret bounds for kernel-based reinforcement learning," *ArXiv*, vol. abs/2004.05599, 2020.

[32] E. R. Novoseller, Y. Sui, Y. Yue, and J. W. Burdick, "Dueling posterior sampling for preference-based reinforcement learning," in *UAI*, 2020.

[33] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, pp. 285–294, 1933.

[34] D. Russo and B. V. Roy, "An information-theoretic analysis of thompson sampling," *ArXiv*, vol. abs/1403.5341, 2016.

[35] A. Pacchiano, A. Saha, and J. N. Lee, "Dueling rl: Reinforcement learning with trajectory preferences," *ArXiv*, vol. abs/2111.04850, 2021.

[36] L. Faury, M. Abeille, C. Calauzènes, and O. Fercoq, "Improved optimistic algorithms for logistic bandits," in *ICML*, 2020.

[37] Y. Xu, R. Wang, L. F. Yang, A. Singh, and A. W. Dubrawski, "Preference-based reinforcement learning with finite-time guarantees," *ArXiv*, vol. abs/2006.08910, 2020.

[38] Y. Yue and T. Joachims, "Beat the mean bandit," in *ICML*, 2011.

[39] M. Falahatgar, A. Orlitsky, V. Pichapati, and A. T. Suresh, "Maximum selection and ranking under noisy comparisons," in *ICML*, 2017.

[40] C. Jin, A. Krishnamurthy, M. Simchowitz, and T. Yu, "Reward-free exploration for reinforcement learning," in *ICML*, 2020.

[41] A. Zanette and E. Brunskill, "Tighter problem-dependent regret bounds in reinforcement learning without domain knowledge using value function bounds," in *ICML*, 2019.

[42] C. Gulcehre, Z. Wang, A. Novikov, T. L. Paine, S. G. Colmenarejo, K. Zolna, R. Agarwal, J. Merel, D. J. Mankowitz, C. Paduraru, G. Dulac-Arnold, J. Z. Li, M. Norouzi, M. W. Hoffman, O. Nachum, G. Tucker, N. M. O. Heess, and N. de Freitas, "Rl unplugged: A suite of benchmarks for offline reinforcement learning," 2020.

[43] J. Achiam and D. Amodei, "Benchmarking safe exploration in deep reinforcement learning," in *arXiv preprint arXiv:1910.01708, 7:1*, 2019.

[44] K. Lee, L. Smith, A. D. Dragan, and P. Abbeel, "B-pref: Benchmarking preference-based reinforcement learning," *ArXiv*, vol. abs/2111.03026, 2021.

[45] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. de Las Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, T. P. Lillicrap, and M. A. Riedmiller, "Deepmind control suite," *ArXiv*, vol. abs/1801.00690, 2018.

[46] T. Yu, D. Quillen, Z. He, R. C. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," *ArXiv*, vol. abs/1910.10897, 2019.

[47] W. S. El-Kassas, C. R. Salama, A. A. Rafea, and H. K. Mohamed, "Automatic text summarization: A comprehensive survey," *Expert Syst. Appl.*, vol. 165, p. 113679, 2021.

[48] S. Narayan, S. B. Cohen, and M. Lapata, "Ranking sentences for extractive summarization with reinforcement learning," in *NAACL*, 2018.

[49] N. Schluter, "The limits of automatic summarisation according to rouge," in *EACL*, 2017.

[50] Y. Gao, C. M. Meyer, and I. Gurevych, "April: Interactively learning to summarise by combining active preference learning and reinforcement learning," in *EMNLP*, 2018.

[51] Y. Gao, Y. Gao, C. M. Meyer, and I. Gurevych, "Preference-based interactive multi-document summarisation," *Information Retrieval Journal*, pp. 1 – 31, 2019.

[52] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving, "Fine-tuning language models from human preferences," *ArXiv*, vol. abs/1909.08593, 2019.

[53] N. Stiennon, L. Ouyang, J. Wu, D. M. Ziegler, R. J. Lowe, C. Voss, A. Radford, D. Amodei, and P. Christiano, "Learning to summarize from human feedback," *ArXiv*, vol. abs/2009.01325, 2020.

[54] J. Wu, L. Ouyang, D. M. Ziegler, N. Stiennon, R. Lowe, J. Leike, and P. F. Christiano, "Recursively summarizing books with human feedback," *ArXiv*, vol. abs/2109.10862, 2021.

[55] C. Liu, X. Xu, and D. Hu, "Multiobjective reinforcement learning: A comprehensive overview," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, pp. 385–398, 2015.

[56] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei, "Reward learning from human preferences and demonstrations in atari," in *NeurIPS*, 2018.

[57] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," *ArXiv*, vol. abs/2002.05709, 2020.