Interpretable Reward Learning via Differentiable Decision Trees

Akansha Kalra, Daniel S. Brown
University of Utah
akanshak@cs.utah.edu, dsbrown@cs.utah.edu

Abstract

There is an increasing interest in learning rewards and models of human intent from human feedback. However, many methods use blackbox learning methods that, while expressive, are hard to interpret. We propose a novel method for learning expressive and interpretable reward functions from preference feedback using differentiable decision trees. We test our algorithm on two test domains, demonstrating the ability to learn interpretable reward functions from both low- and high-dimensional visual state inputs. Furthermore, we provide preliminary evidence that the tree structure of our learned reward functions is useful in determining the extent to which a reward function is aligned with human preferences.

1 Introduction

We consider the problem of AI-alignment via reward learning [14]. While neural networks are highly capable of learning reward functions for a variety of tasks, the representations they learn are often uninterpretable beyond the first few layers. This inherently begs the question of how we can trust what our neural network has learned if we can not comprehend and infer the representations it has learned. If we can not understand the objective that a robot or AI system has learned, then it is difficult to know if the AI's behavior will be aligned with what we want it to do. This is particularly significant in tasks where human safety is on the line, for example in autonomous navigation systems and assistive robots.

We propose a novel method for learning interpretable reward functions by using pairwise preferences to learn a differentiable decision tree model of a reward function. This enables us to learn an expressive reward function that is still interpretable and can be explained via a discrete number of decisions. We evaluate our approach on two different domains and the results show that our approach enables interpretability for low-dimensional as well as high dimensional state and action space.

To train our tree we leverage human feedback in the form of pairwise preference labels over trajectories. It is often easier for someone to qualitatively rank two or more behaviors than to demonstrate a good behavior. Learning from pairwise preferences over trajectories is a common approach to learning reward functions and corresponding RL policies [19, 6]. It has been shown that preference learning allows generalizing to various domains even when provided sub-optimal demonstrations without any explicit preferences and can achieve better-than-demonstrator performance [4]. Despite requiring pairwise preferences, prior work has shown that other forms of feedback, such as demonstrations [5], e-stops [10], and corrections [15], can all be represented in terms of preferences, enabling our approach to be applied, even when pairwise preference labels are not explicitly available. Prior works on learning reward functions from preferences typically either assumes access to a set of hand-designed reward features [17, 2] or uses deep convolutional or fully connected networks for reward learning [6, 4, 13]. In contrast to prior work, we seek to learn expressive, but also interpretable reward functions from preferences via Differential Decision Trees [9].

ML Safety Workshop, 36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Differential Decision Trees (DDTs), also referred to as Soft Decision Trees [9, 11] seek to combine the flexibility of neural networks with the interpretable structure of decision trees. Differentiable decision trees have previously mainly been applied to classification tasks. Recent work has investigated using DDTs for reinforcement learning tasks [18, 7] or used classical (non-differentiable) decision trees to explain a given reward function in an MDP [16]; however, none of these previous methods explore using decision trees for reward function learning. To the best of our knowledge, the only prior work on learning reward functions using decision trees is recent work by Bewley et al. [1]. Similar to our approach, Bewley et al. propose to learn a decision tree given preferences over trajectories. However, their approach is not differentiable and thus requires an indirect, multi-stage optimization approach. Furthermore, internal nodes in their approach have the form $(s, a)_d \geq c$ for each dimension d of the state-action space and threshold c. This approach divides the state-action space into axis aligned hyperrectangles, which works for lower-dimensional spaces, but does not scale to higher-dimensional state and action spaces. By contrast, our proposed approach is fully differentiable enabling us to learn reward functions directly from preferences via gradient descent. Additionally, our internal nodes contain learnable parameters, enabling us to learn reward functions over both high- and low-dimensional state and action spaces while still retaining the interpretability of a decision tree.

2 Differential Decision Trees for Reward Learning from Preferences

We first define the the most essential parts constituting a decision tree individually and then show how to combine them to form the actual differential decision tree.

Internal Nodes Each non-leaf node i has a linear layer with learned parameters $\mathbf{w}_i$ and a bias term b upon which sigmoid activation function, σ, is applied to derive the routing probability given an input $\mathbf{x}$. For a given node i the probability of going to the leftmost branch is thus defined as

$$p_i(x) = \sigma(\beta(\mathbf{x}\mathbf{w}_i + b)). \tag{1}$$

In this way, each internal node depends directly on the input (Figure 1). The differentiable decision tree learns a hierarchy of decision boundaries that determine the routing probabilities for each input. An inverse temperature, β, is included to above the equation for controlling the degree of soft decisions.

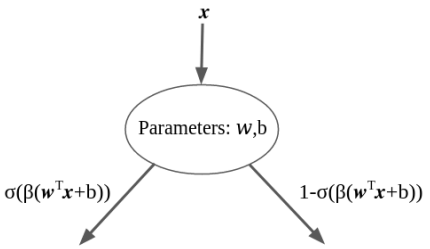


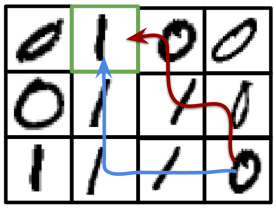
Figure 1: Routing Probability of an internal node

Leaf Nodes Because each leaf node now has a probability associated with it, we can think of the DDT as a hierarchical mixture of experts [12]. Following Frosst et al. [9] we model each leaf node as a softmax distribution over the number of output classes c. So a leaf node l has a parameter ϕ^l is vector of dimension $1 \times c$ where each element of the leaf distribution is sampled randomly from a uniform distribution and the probability distribution of a leaf is given by Q^l is defined by

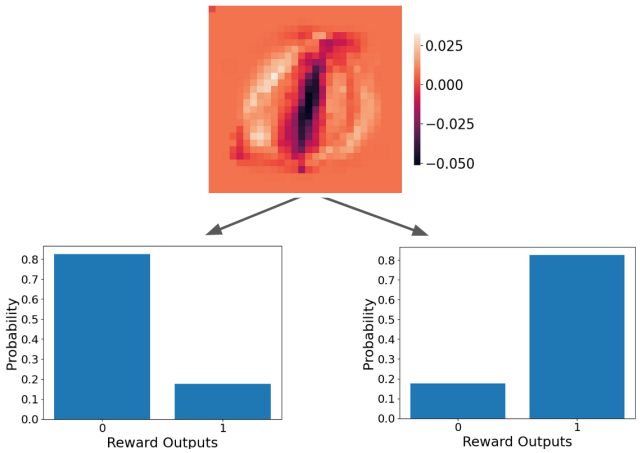
$$Q_c^l = \frac{e^{\phi_c^l}}{\sum_{c'} \phi_{c'}^l} \tag{2}$$

Reward Predictions We now describe how to use a soft decision tree for reward learning. We assume that the user can specify a range of possible reward values suitable for the task (note that optimal policies are invariant to reward scale, so, in practice, choosing the range [0,1] or [-1,1] will usually suffice). We then discretize this space and define a discrete vector of reward values $\mathbf{R} = (r_0, r_1, \ldots, r_{c-1})$, where c denotes the number of classes for the DDT, where each class index i is assigned reward value r_i. To ensure differentiability, we require soft reward estimates, rather than having the DDT output a single discrete prediction based on a single path through the tree.

We accomplish this as follows. First, the tree of depth d is built by $2(d-1)+1$ internal nodes and $2d$ leaves. Given an input $\mathbf{x}$, the path probability from root node to a leaf ℓ is denoted by $P^\ell(\mathbf{x})$ and the reward of the tree is sum over all leaves of path probability of reaching that leaf P^ℓ multiplied



(a) A pair of trajectories that both reach the terminal state (green border). The blue trajectory (which visits more 1's) is preferred over the red trajectory.



(b) DDT of depth 1 with heatmap at the root node illustrating the difference in routing probability of each pixel when it is on versus off. The bar graphs depict the probability distribution of reward outputs at each leaf node.

Figure 2: MNIST Gridworld

with dot product of probability distribution at that leaf $\mathbf{Q}^\ell$ with $\mathbf{R}$.

$$r_\theta(\mathbf{x}) = \sum_\ell P^\ell(\mathbf{x})(\mathbf{Q}^\ell \cdot \mathbf{R}) \tag{3}$$

Note that this gives us a weighted soft reward that can be used when learning from preference labels. At test-time, we can output a single reward prediction by simply returning the maximum probability reward that corresponds to the leaf node with maximum routing probability.

Training via Trajectory Preference Labels Given preferences over trajectories of the form $\tau_i \prec \tau_j$, where $\tau = (s_1, s_2, ...s_T)$, we can train our entire differentiable decision tree via the following cross entropy loss that results from the Bradley Terry model of preference labels [3]:

$$\mathcal{L}(\theta) \approx -\sum_{\tau_i \prec \tau_j} \log \frac{\exp \sum_{s \in \tau_j} r_\theta(s)}{\exp \sum_{s \in \tau_i} r_\theta(s) + \exp \sum_{s \in \tau_j} r_\theta(s)}. \tag{4}$$

3 Experiments and Results

We evaluate the interpretablility of the learned reward function on two domains: MNIST Gridworld Environment and OpenAI gym's Cartpole environment. In both the environments the DDT is trained using preference learning—the DDT has no access to the actual ground truth reward.

3.1 MNIST Gridworld Environments

For our first experiment, we examine whether we can learn rewards from pairwise preference labels over sequences of high-dimensional visual observations. To test this setting, we construct a visual gridworld using MNIST digits as the reward features for each grid cell. We begin by randomly sampling trajectories of variable length from MNIST where each trajectory is a sequence of states and each 28×28 greyscale input image is treated as an individual state. Given two such sequences of trajectories of equal length, the preference label is assigned based on comparison between the sum of ground truth labels of each input image comprising the respective trajectories. To test whether the DDT can learn interpretable routing features at the internal nodes, we train a DDT of depth 1 as in the simplest soft decision tree with 1 root node and 2 leaf nodes on pairs of preference demonstrations over trajectories consisting of images of zeros and ones (see Figure 2a for an example pairwise

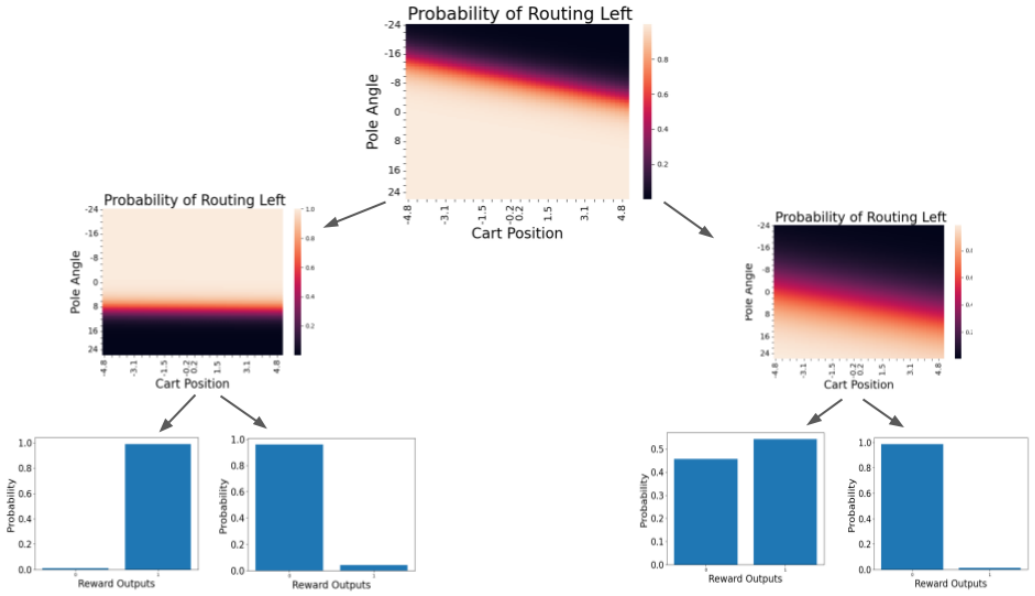


Figure 3: Visualization of the learned Cartpole reward DDT of depth 2 with heatmaps on internal nodes depicting the learned routing probability and with bar graphs on leaves depicting probability distribution of the reward outputs

trajectory comparison). We set $\mathbf{R} = (0.0, 1.0)$. For this experiment, we use a learning rate of 0.001, weight decay of 0.05, and the Adam optimizer. After training the reward DDT, we construct a heatmap by toggling each pixel in a greyscale image and taking the difference in routing probabilities at the root node. The resulting heatmap demonstrates the fact that DDT learns to branch based on features that visually correspond to a 0 (routes left) and 1 (routes right).

3.2 Cartpole

For our second experiment, we use the classic Cartpole environment from OpenAI Gym. Each state in the cartpole environment has 4 values corresponding to cart position, cart velocity, pole angle and pole velocity. For simplicity, we consider the reward features to be comprised of only cart position and pole angle. We generate a wide variety of trajectories by simply running a random policy in the environment for 200 steps for each trajectory. Since we assume no access to a reward function, we also do not have access to any kind of terminal or done flag (since this would leak significant information about the true reward [8]). Thus, we ignore the done flag in the cartpole environment and keep accumulating states in the trajectory for 200 timesteps, even if the pole falls over. We design a synthetic preference labeler that assigns the reward of +1 only if the cart position $x \in [-2.4, 2.4]$ and the pole angle $\theta \in [-12^\circ, +12^\circ]$ and a 0 otherwise.Pairwise preferences are assigned based on comparison between the cumulative rewards that each trajectory in the pair scores. Given pairwise preference labels over these suboptimal trajectories, we train a DDT of depth 2: the tree has 3 internal nodes and 4 leaf nodes. We use $\mathbf{R} = (0.0, 1.0)$. We use a learning rate and weight decay both equal to 0.001 and the Adam optimizer.

We note that even though the ground truth preferences are based on both cart position and pole angle, it is usually the case that the pole falls past the desirable range long before the cart leaves the desirable range. Thus, our dataset is biased and we hope to be able to pickup on this bias, and the corresponding misaligned reward function by inspecting the learned reward DDT.

To interpret the learned reward DDT, we visualize the heatmap of routing probability at each internal node (as a function of cart position and pole angle) along with plots of the leaf distribution over output classes. Figure 3 shows the learned reward DDT. It is clear that the learned reward is misaligned: the reward DDT has learned to approximate the preferences over pole angle, but pays much less attention to the pole angle when predicting rewards.

4 Conclusion and Future Work

We formulate a novel approach for learning an expressive and interpretable reward function using preference learning that can scale to high dimensions. Future work include applying our framework to real-world complicated tasks and leveraging more sophisticated techniques for understanding the decisions made within the reward DDT. We are also interested in testing our approach on tasks with more complex reward functions that may require more than two reward classes and trees with larger depth. Another interesting area of future work is to take a reward function that has already been trained by a neural network and distill it into an interpretable differentiable decision tree.

References

[1] Tom Bewley and Freddy Lecue. Interpretable preference-based reinforcement learning with tree-structured reward functions. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pages 118–126, 2022.

[2] Erdem Biyik, Malayandi Palan, Nicholas C Landolfi, Dylan P Losey, Dorsa Sadigh, et al. Asking easy questions: A user-friendly approach to active reward learning. In *Conference on Robot Learning*, pages 1177–1190. PMLR, 2020.

[3] 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.

[4] Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pages 783–792. PMLR, 2019.

[5] Daniel S Brown, Wonjoon Goo, and Scott Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning*, pages 330–359. PMLR, 2020.

[6] Paul F Christiano, Jan Leike, Tom B Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *NIPS*, 2017.

[7] Youri Coppens, Kyriakos Efthymiadis, Tom Lenaerts, Ann Nowé, Tim Miller, Rosina Weber, and Daniele Magazzeni. Distilling deep reinforcement learning policies in soft decision trees. In *Proceedings of the IJCAI 2019 workshop on explainable artificial intelligence*, pages 1–6, 2019.

[8] Pedro Freire, Adam Gleave, Sam Toyer, and Stuart Russell. Derail: Diagnostic environments for reward and imitation learning. *arXiv preprint arXiv:2012.01365*, 2020.

[9] Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017.

[10] Gaurav R Ghosal, Matthew Zurek, Daniel S Brown, and Anca D Dragan. The effect of modeling human rationality level on learning rewards from multiple feedback types. *arXiv preprint arXiv:2208.10687*, 2022.

[11] Hussein Hazimeh, Natalia Ponomareva, Petros Mol, Zhenyu Tan, and Rahul Mazumder. The tree ensemble layer: Differentiability meets conditional computation. In *International Conference on Machine Learning*, pages 4138–4148. PMLR, 2020.

[12] Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

[13] Kimin Lee, Laura Smith, and Pieter Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *arXiv preprint arXiv:2106.05091*, 2021.

[14] Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018.

[15] Shaunak A Mehta and Dylan P Losey. Unified learning from demonstrations, corrections, and preferences during physical human-robot interaction. *arXiv preprint arXiv:2207.03395*, 2022.

[16] Jacob Russell and Eugene Santos. Explaining reward functions in markov decision processes. In *The Thirty-Second International Flairs Conference*, 2019.

[17] Dorsa Sadigh, Anca D Dragan, Shankar Sastry, and Sanjit A Seshia. Active preference-based learning of reward functions. In *Robotics: Science and Systems*, 2017.

[18] Andrew Silva, Matthew Gombolay, Taylor Killian, Ivan Jimenez, and Sung-Hyun Son. Optimization methods for interpretable differentiable decision trees applied to reinforcement learning. In *International conference on artificial intelligence and statistics*, pages 1855–1865. PMLR, 2020.

[19] Christian Wirth, Johannes Fürnkranz, and Gerhard Neumann. Model-free preference-based reinforcement learning. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.