The Thirty-Second International Florida
Artificial Intelligence Research Society Conference (FLAIRS-32)

Explaining Reward Functions in Markov Decision Processes

Jacob Russell, Eugene Santos Jr.
Thayer School of Engineering
Dartmouth College
14 Engineering Drive
Hanover, NH 03755
{Jacob.A.Russell.th, Eugene.Santos.Jr}@Dartmouth.edu

Abstract

Rewards in Markov Decision Processes (MDP) define the behavior of the model. Without a clear interpretation of what the reward function is and is not capturing, one cannot trust their model nor diagnose when the model is giving incorrect recommendations. Increasing complexity of state-of-the-art models used to represent the reward function and model-free methods that attempt to avoid representing this function make trusting the model much more difficult. We map these reward functions onto a standard classification problem where we can explain what factors the model considers in making decisions in local and global contexts and quantify whether the fit of the reward function is likely to be good for explaining the behavior of the model. We evaluate our proof-of-concept on both the standard version and a modified version of the Object World domain to add more nonlinearity.

Introduction

Human trust in machine recommendations depends upon whether humans can understand what the model is capturing (Herrmann, Kloth, and Feldkamp 1998). In Markov Decision Processes (MDPs), the behavior of the system is dependent upon the reward function. These reward functions are susceptible to many different problems (Russell and Norvig 2016; Amodei and Clark 2016; Amodei et al. 2016) and are therefore difficult to specify. If we can explain what the reward function is and is not capturing, it should be easier for a user to understand what the reward function is doing.

Our goal is to understand what a specific MDPs reward function has captured and to communicate the meaning behind the reward function to a human analyst. Oftentimes, human intuition can well outperform machines but humans have limited working memory sizes compared to machines. This means that reducing the information to a manageable set can increase human understanding of the model. Human trust in any model is correlated with their understanding of the model. Sometimes humans are interested in understanding the overall behavior as would be captured by a global model, but other times they are interested in specific cases that may be more important to not make a mistake in. To build trust, we provide local and global explanations of what the model captured. In cases where a human has an expectation of what the reward function should entail, they can evaluate whether simplified explanations match their goal.

In order to understand what these algorithms are capturing, we propose two methods of explanation: a global method — which describes what factors are usually important across all decisions, and a local method — which describes in specific contexts or situations what factors were considered. We evaluate both models in the Object World (Levine, Popovic, and Koltun 2011) domain as a proof of concept. In order to have faith in the model, it should be able to give both types of explanations and they should be consistent with each other. Crucially, if the global and local explanations are inconsistent, we can identify that we need a more complicated model to explain the reward function (or a different selection of hyperparameters) without needing ground truth.

In the following section, we review existing work related to explanation in MDPs. We then describe our approach to explanation, highlighting what we can explain, and how we can be confident in our explanations. We conclude by summarizing what we found and describing how we plan to update our approach in the future.

Related Work

MDPs are used in a both inverse reinforcement learning (including behavioral cloning, imitation learning, and apprenticeship learning) and reinforcement learning. We focus on the more classical reinforcement learning for simplicity, but note that the approach works for any case that has a reward function. In reinforcement learning a MDP is a 5-tuple:

$$M = (S, A, P_a(\cdot,\cdot), R_a(\cdot,\cdot), \gamma) \tag{1}$$

Where S is a set of states, A is a set of actions, $P_a(s, s')$ is the probability that action $a \in A$ will transition from state s at time t to state s' at time $t+1$, $R_a(s, s')$ is the reward for the same transition, and $\gamma \in [0, 1]$ is the discount factor controlling for a trade-off between immediate and future rewards.

The goal of reinforcement learning is usually to identify a policy $\pi^*(s) \to a$ from a set of policies such that given a state s, we determine the (expected) best action a to take for all states or to use the reward function itself to guide agents as they learn.

Explanation of MDPs is typically done within this setting of reinforcement learning once a policy has already been learned. Most explanations focus on identification of relevant features that are captured by the policy such as in (Elizalde et al. 2007), integration of these relevant features into natural language explanation generation functions (as in (Elizalde et al. 2008b; 2009; Dodson, Mattei, and Goldsmith 2011; Khan, Poupart, and Black 2009)), and evaluation of explanation systems with real users (as in (Elizalde et al. 2008b; 2009)). (In medicine) humans have been shown to have preferences for natural language explanations (Witteman, Renooij, and Koele 2007). In robotics, explanation has been used as a means for communicating spacial information about robot behavior to humans (Korpan et al. 2017). Mapping from feature importance onto natural language is typically done in templates such as in (Elizalde et al. 2008b; 2009). This modular decomposition with templates implies that other approaches may reuse the same natural language generation techniques as long as they have their own method for identifying important features. We therefore focus on feature importance for explanations and assume there is a mapping onto natural language if required.

To identify important features Elizalde et al. 2007; 2008a; 2009 mapped factored MDPs onto a Dynamic Bayesian Networks (DBNs), thereby reducing the state space size for complex MDPs. From the DBNs they use automated and manual approaches to generating explanations. In the manual approach, they used domain experts to identify variables of interest and determine which variables were important and used the variables as explanations as recommendations to help humans decide which action to take. Their automated method maps the MDP to a DBN, performs variable elimination, ranks the importance of the variables, and then uses a weighted linear combination of variable importance based upon a heuristic distance to the value functions within a decision-tree.

Their approach propagates value functions and uses a heuristic to trade-off between current and future rewards. We argue that the reward function should already be making this trade-off and therefore explanations that focus on the reward function instead of the policy should address the same problem without requiring heuristics. We define our explanation approach in the following section for both local and global features.

Approach

Given a MDP, we generate explanations as lists of which features were important, globally and locally. Globally important features are features that are considered generally relevant. These global features are relevant over many different contexts. Locally important features describe why an action was taken in terms of the features that we see or expect to see when we are taking an action from a specific state. Typical MDP applications are in robotics where performance can be evaluated objectively, but this need not be strictly true. We focus on explanation instead of accuracy because in many contexts it is more important to answer the questions: “why are we getting the result that we are getting?”, “Is the answer that we are getting reasonable?”, and finally, we ask, “what can we do to get a better answer?”

How we perform explanation depends on properties of our reward function structure because properties of the reward function define what it can learn. We do not just want to know which local and global features were important, but also if interactions between the local and global explanations can also be meaningful. We hypothesize that if the local and global explanations are sufficiently similar, then our model has a very good fit. If there is a discrepancy between local and global explanations, then we hypothesize that either the local explanations are overfitting to the noise or the global explanations are underfitting and are not complex enough to capture the underlying behavior, or both and we need to use a more complicated model to capture the relationships.

There are minor differences in our representation of the reward function from other approaches. Early work on MDPs allocated rewards to triples (s, a, s') without having to know any of the features on individual states. The modern equivalents, however, often store the reward values on 2-tuples (s, a), or just on the states themselves s. In order to support all possible mappings, we define our rewards on triples, but the problem we are solving only uses rewards on the end state s'. We are mapping our reward function onto supervised learning in order to explain the learned rewards. With rewards stored only on 2-tuples, we miss some of the information that is relevant in explaining decisions. Our reward function is, therefore, learned on 3-tuples so that the explanations can look at the expectation of the results of the action. Our rewards can generally be continuous, $R_a(s, s') \in \mathbb{R}$, or discrete with the rewards as integers, $R_a(s, s') \in \mathbb{Z}$. For simplicity, we use the discrete formulation of the Object World scenario (as implemented in (Alger 2016)) where rewards are integers. Given this integer formulation, explanation can map onto a standard classification problem where we concatenate vectors of the attributes $atts(s)$ for our states and the actions taken as features and then predict our reward values as the response variable as shown in Equation 2.

$$X = \begin{bmatrix} atts(s_1) & a_1 & atts(s'_1) \\ \vdots & \ddots & \vdots \\ atts(s_n) & a_n & atts(s'_n) \end{bmatrix} Y = \begin{bmatrix} R_{a_1}(s_1, s'_n) \\ \vdots \\ R_{a_n}(s_n, s'_n) \end{bmatrix} \tag{2}$$

If our rewards were continuous instead of discrete, our explanation method would map onto a standard regression problem and the same technique would apply.

Since we have a goal of both global and local explanation, we use a simple, explainable model for a global explanation and use local interpretations for our local explanations. The global model that we fit was a decision-tree (as implemented in scikit-learn(Pedregosa et al. 2011)) and we use Locally Interpretable Model-agnostic Explanations(Ribeiro, Singh, and Guestrin 2016) (LIME) to explain local behavior. These selections were made for ease of access and ease of interpretability, but the approach is general to any classifier. An advantage of decision-trees (and random forests) is that they have a well-defined measure of importance in Gini

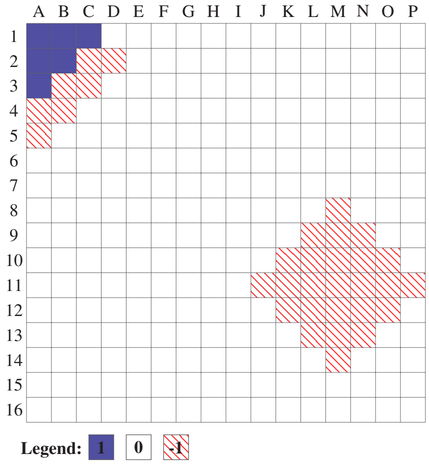

Figure 1: Object World ground truth rewards

Importance (Leo et al. 1984).

We generate our $N \times N$ Object World grid and then our ground truth reward function using value iteration. An Object World space is defined by the five actions that the agent can take to move to any of the four cardinally adjacent cells (including wrapping around the board) or stay in the cell in which they are currently at within the grid. There are randomly placed objects. Each object has one of C inner and outer colors. The features that the agent can see are binary $|2CN|$ binary indicators of whether the inner and outer colors are observable within a distance d unitary steps. The true reward for the game is well-defined for a state:

$$R_{ground}(s, a, s') = \begin{cases} 1 & dist(c_0^{s'}) \leq 3, dist(c_1^{s'}) \leq 2 \\ -1 & dist(c_0^{s'}) \leq 3, dist(c_1^{s'}) > 2 \\ 0 & otherwise \end{cases} \tag{3}$$

The well-defined reward function affords humans expectations of which features should be considered important globally. This expectation is important because we believe that the human to whom we are explaining the model has enough understanding of the component pieces that an oversimplified model is acceptable (or perhaps preferred). The (binary) indicator features for colors $c_0^{s'}$ and $c_1^{s'}$, for example, should be important in determining the correct behavior since we know that we are optimizing the reward function. So we perform the mapping onto supervised learning for an instance of Object World. The ground truth rewards in this Object World environment are depicted in Figure 1.

Generically, our mapping from the problem onto the matrix is flexible enough to use a policy, a sampling from trajectories, or enumerate all possible transitions. For this paper, we use the optimal policy as given by value iteration.

We build and run an Object World scenario with a grid size of 16 and only two colors. After fitting the policy with a global decision-tree, we yield our global feature importances (Gini) as shown in Table 1. To denote that features come from s', we use "dest" as a prefix and if they come from s, we use "start." We represent inner colors with IC and outer colors with OC. We note that the most important features as fit by the decision-tree selects only 8 out of 133 possible features as important, reducing the number of features that a human would have to interpret to a manageable quantity. Additionally, we note that the features are all within a distance of four, and the most important feature in determining the action is whether we are within 2 units of color 0.

Feature	**Importance**
destOC0Less2	0.7356
destIC0Less1	0.1411
actionSouth	0.0403
startIC0Less3	0.0355
destIC1Less1	0.0198
destIC1Less3	0.0118
destIC0Less2	0.0097
destIC0Less3	0.0061

Table 1: All non-zero global GINI importance features for an Object World scenario

So the global explanation captured some useful behavior for interpreting what the model will do. Only a single noise factor was identified (the outer color for color 0). We still need to evaluate whether different features are important within different contexts, however. We fit LIME using a decision-tree as a predictor. LIME works by perturbing inputs using a euclidean distance and weighting points ((s, a, s') in our case) that are more proximal higher than distant points in order to smooth predictions of what the correct reward value will be for the state. We sample 26 nearest neighbors (10%) using the euclidean distance between the features, holding out the triple that we are predicting. In order to ensure that we do not have a degenerate solution space, we ensure that we have at least one sample from each reward value. We show the top results from LIME explaining moving from cell A1 to cell A2 from Figure 1 in Table 2.

The variation in the predicted importance of the individual features depends upon the relationships across the classes. While the model-fitting by LIME is a decision-tree like in the global predictions, perturbing the inputs enables it to describe how local features change the space without having to consider every feature. In this case, the predicted class given the local features was strongly weighted towards predicting reward=1. The simplification afforded by using a local explanation meant that a single feature described which reward class we were in (startOC1Less2>0.00). Again, all of the important factors contain a distance measure of within 3 units, like the actual ground truth reward.

To compare whether features from using the global model to predict the reward from being in cell A1 and taking action move south, we ran a Spearman correlation between the local and global features. The use of Spearman is important because the decision tree learns a (potentially) nonlinear relationship, so we only care whether the *ordering* of the features is similar between the local and global cases. This is similar to a trust score (Jiang, Kim, and Gupta 2018). For moving from A1 to A2, the Spearman correlation between the local and global explanations is 0.1208, with

reward=-1	
Feature	**Importance**
destIC1Less1>0.00	0.1517
startOC1Less0>0.00	0.0362
actionEast>0.00	0.03
startIC0Less1>0.00	0.0295
destOC0Less0≤0.00	-0.0274
startIC0Less0>0.00	0.0219
actionWest≤0.00	-0.0209
startIC1Less0≤0.00	0.0136
actionStay≤0.00	0.0094
reward=0	
Feature	**Importance**
destIC1Less1>0.00	-0.9646
startOC1Less2>0.00	-0.9642
startIC0Less0>0.00	0.0629
startOC0Less3>0.00	-0.0591
startIC0Less3>0.00	-0.0509
startOC1Less0>0.00	0.0497
actionEast>0.00	0.0427
destIC1Less2>0.00	-0.0221
startIC1Less0≤0.00	0.02
reward=1	
Feature	**Importance**
startOC1Less2>0.00	0.9623
destIC1Less1>0.00	0.8129
startOC1Less0>0.00	-0.0859
startIC0Less0>0.00	-0.0848
actionEast>0.00	-0.0727
startOC0Less3>0.00	0.0565
startIC0Less3>0.00	0.0466
startIC0Less1>0.00	-0.0368
startIC1Less0≤0.00	-0.0336

Table 2: Top 9 important features per class from LIME

$p = 3.118\mathrm{e}^{-5}$. So there is a significant correlation or consistency between the order of consideration of factors between the local and global model for this point, implying that the global model described the same behavior as the local model and likely had a good fit. The low correlation 0.1208 seems poor at first glance, however, the features are not truly independent. $dist(c_0^{s'}) < 3$ could have been learned on $dist(c_0^{s'}) > 2$ and is left out of the Spearman correlation. Had features been orthogonal, this value would have been higher.

So given a reasonable amount of agreement between the models, in combination with the global explanation seemingly matching our local explanation. Similar to a Pearson correlation, the actual desired goodness of fit between the local and global models most be weighed for each problem. Keep in mind that this is telling us that the local model considered similar features to the global model, a stronger statement than just that their predictions were similar. In cases where the local model and the global model are not consistent with each other, this approach enables us to find potentially poorly learned rewards, while the features themselves explain what factors were considered differently between the models. The regular Object World is not a good example to demonstrate inconsistencies between the local and global models because they considered features nearly the same way in almost every case.

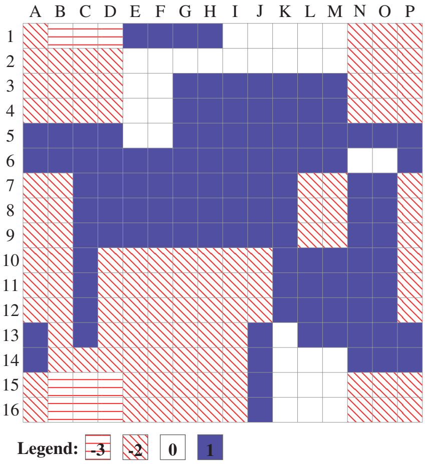

Figure 2: NLObjectWorld ground truth rewards

We have demonstrated that when there is a lot of consistency between the local and global models it may be a good fit, however, we would like to explore a case where there is inconsistency. As a means to this end, we adapt the global reward function from Object World to create Nonlinear Object World (NLOW). We replace Equation 3 with an indicator function shown in Equation 4 and a new nonlinear reward function based upon the xor function (using $\oplus$ to represent xor) in Equation 5.

$$I_{c_i^{s'}} = \begin{cases} 1 & dist(c_i^{s'}) \leq 3 \\ 0 & otherwise \end{cases} \tag{4}$$

$$R_{ground}(s,a,s') = \begin{cases} 1 & I_{c_0^{s'}} \oplus I_{c_1^{s'}} \oplus \ldots I_{c_n^{s'}} = 1 \\ \sum_{i=0}^{n} -I_{c_i^{s'}} & otherwise \end{cases} \tag{5}$$

With these equations we can generate a new NLOW grid and use the same methodology for explanation. We specify $n = 3$, meaning that the only important features for our reward function are distances between inner colors 0–2, all other features (both inner and outer colors) are noise.

We note that the nonlinear interactions make a more difficult to interpret reward space. We also increased the complexity by adding additional colors. The increased complexity of the problem makes it a little bit difficult to directly compare the Gini values because the number of classes has changed. We do seem to get much lower Gini importance values as compared to the regular Object World problem (shown in Table 3). This reduction in the Gini values could theoretically be attributed to the more complex space and no single factor contributing as much to as large of a reduction in the variance of the problem. Since we know the ground truth function, we can identify that any factor with a distance larger than 4 should have no impact on the reward function.

Among our top 10 important factors in our global model, 7 out of 10 factors consider irrelevant colors, implying that we might not have gotten as great of a fit with the global explanations. This is both expected and desired in order to highlight potential limitations of the global model.

Feature	Importance
destOC2Less4	0.248
destIC0Less3	0.0764
destOC11Less8	0.0758
destIC2Less3	0.0643
startOC4Less6	0.0599
destIC10Less7	0.0509
startOC10Less7	0.0356
destIC9Less2	0.0342
destIC9Less7	0.0335
startOC1Less8	0.0307

Table 3: Top 10 global GINI importance features for a NLObjectWorld scenario

The decision-tree algorithm had a harder time fitting the data for the global function because nonlinear interactions meant that it had to sacrifice local accuracy in order to get good general predictions. How does this actually appear in the relationship between the local and global models? Out of the 256 samples only 96 had significant correlations between the local and global models. None of the 96 Spearman r-values exceeded 0.05 and most were very close to 0.0.

The local models, ended up being more accurate like moving from cell A3 to A4 with features shown in Table 4 where exactly one inner color appeared in the reward function. While it performed alright for this point, consideration of how to proceed must be done carefully if the local and global explanations are inconsistent. The best explanation is that another model fitting with a more nonlinear model should be performed.

reward=1	
Feature	**Importance**
startOC1Less3>0.00	0.6917
startIC4Less5≤0.00	0.5426
startOC4Less6≤0.00	-0.2778
startIC0Less0≤0.00	-0.1052
startIC10Less1≤0.00	0.0919
destIC0Less0≤0.00	0.0858
startIC11Less0≤0.00	-0.0824
destOC4Less0≤0.00	-0.0591
destIC8Less0≤0.00	-0.0538
startIC8Less0≤0.00	0.053

Table 4: Top 10 local important features for the top predicted class from LIME for moving from cell A3–A4

When there is no relationship between the local and global model and we need to fit a more complex model or select different hyperparameters (such as a random forest with more trees) if we want to have more confidence in our explanation of the rewards.

Discussion and Conclusions

We demonstrated a mapping from Markov Decision Processes onto supervised learning in order to explain which features were important in deriving the reward values. Our explanations consist of lists of features that are generally important coming from a global model and lists of features that are important only in specific contexts from our local model. By comparing the ordering of features through juxtaposing these local and global models, we could identify when we had a good fit and when we had a poor fit without needing to look at the real ground truth rewards. We verified the fit with explanations of factors that were important using reward functions that had intuitively obvious properties to identify problems (for a human) showing that the simpler reward function of Object World was learned well by our global model and demonstrably consistent with the local model. We also showed that our extension to a more nonlinear version, NLObjectWorld, was not captured well by the global model, highlighting that we can evaluate whether a global model is sufficient to explain the behavior. When the global model is not sufficient to explain the behavior, we must use care when using either model. The Gini coefficients and the importance values from LIME both assume that the model was correct and come up with explanations given the assumption. Our work extends LIME to characterize whether the correct features are actually captured in both locally and globally and enables us to understand which explanations are likely to be more useful.

Our demonstration is intended as a proof-of-concept. Comparing and contrasting the differences between the local and global explanations can help us evaluate whether the reward function from our MDP is consistent with itself without requiring any true ground truth. In domains where ground truth is unavailable or expensive, this may be a potential first step to verify reward correctness. This consistency can be used for model selection but it needs to be treated similarly to the Pearson correlation coefficient in linear regression, where it is selected appropriately to the problem being solved. While the preliminary results appear interesting, we still have a lot that we would like to explore with this technique.

Future work

The approach to both global and local fitting can only train on the features that it can see across a single Markovian step. Giving the classifier access to Q-values from Q-learning or V-values from value iteration would enable explanations of expected rewards. Since this was a proof of concept, the classifiers used were simple but using random forests enable the same explanations. Additional possible relationships could be captured through a custom classifier or objective function.

Reward functions are not always manually specified. Particularly when people are interested in solving increasingly complex problems with MDPs that include significant numbers of variables they use other techniques to get the reward functions and they will have only a sampling of interactions between variables in functional relationships that may not be straightforward. Inverse reinforcement learning (IRL) approaches (and also behavioral cloning, apprenticeship learning, and imitation learning) are state of the art on a variety of problems including Chess, Shogi, and Atari games (Mnih et al. 2015; Silver et al. 2016; 2017) to approximating reward functions or policies. The biggest critiques of these approaches are that they play differently than humans (Hutson 2018) and that missing explanations of what the algorithm is learning is a severe limitation (Bratko 2018). So if we could generate explanations of what these approaches are capturing we could increase trust in these approaches. If we could understand what these approaches are not capturing, we could potentially identify how to adjust the algorithms to solve problems that they currently cannot perform well on, such as StarCraft II (Vinyals et al. 2017).

Acknowledgements

This work supported in part by ONR Grant No. N00014–1–2154, ONR/NPS Grant No. N00024–15–1–0046, AFOSR Grant No. FA9550–15–1–0383, DURIP Grant No. N00014–15–1–2514.

References

Alger, M. 2016. Inverse reinforcement learning. Zenodo. https://doi.org/10.5281/zenodo.555999.

Amodei, D., and Clark, J. 2016. Faulty reward functions in the wild. https://blog.openai.com/faulty-reward-functions.

Amodei, D.; Olah, C.; Steinhardt, J.; Christiano, P.; Schulman, J.; and Mané, D. 2016. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*.

Bratko, I. 2018. Alphazero–what's missing? *Informatica* 42(1).

Dodson, T.; Mattei, N.; and Goldsmith, J. 2011. A natural language argumentation interface for explanation generation in markov decision processes. In Brafman, R. I.; Roberts, F. S.; and Tsoukiàs, A., eds., *Algorithmic Decision Theory*, 42–55. Berlin, Heidelberg: Springer Berlin Heidelberg.

Elizalde, F.; Sucar, L. E.; Reyes, A.; and deBuen, P. 2007. An mdp approach for explanation generation. In *ExaCt*, 28–33.

Elizalde, F.; Sucar, L. E.; Luque, M.; Diez, J.; and Reyes, A. 2008a. Policy explanation in factored markov decision processes. In *Proceedings of the 4th European Workshop on Probabilistic Graphical Models (PGM 2008)*, 97–104.

Elizalde, F.; Sucar, L. E.; Noguez, J.; and Reyes, A. 2008b. Integrating probabilistic and knowledge-based systems for explanation generation. In *ExaCt*, 25–36.

Elizalde, F.; Sucar, E.; Noguez, J.; and Reyes, A. 2009. Generating explanations based on markov decision processes. In *Mexican International Conference on Artificial Intelligence*, 51–62. Springer.

Herrmann, J.; Kloth, M.; and Feldkamp, F. 1998. The role of explanations in an intelligent assistant system. *Artificial Intelligence in Engineering* 12(1-2):107–126.

Hutson, M. 2018. Basic instincts. *Science* 360(6391):845–847.

Jiang, H.; Kim, B.; and Gupta, M. 2018. To trust or not to trust a classifier. *arXiv preprint arXiv:1805.11783*.

Khan, O. Z.; Poupart, P.; and Black, J. P. 2009. Minimal sufficient explanations for factored markov decision processes. In *ICAPS*.

Korpan, R.; Epstein, S. L.; Aroor, A.; and Dekel, G. 2017. Why: Natural explanations from a robot navigator. *arXiv preprint arXiv:1709.09741*.

Leo, B.; Friedman, J. H.; Olshen, R. A.; and Stone, C. J. 1984. Classification and regression trees. *Wadsworth International Group*.

Levine, S.; Popovic, Z.; and Koltun, V. 2011. Nonlinear inverse reinforcement learning with gaussian processes. In *Advances in Neural Information Processing Systems*, 19–27.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, 1135–1144. New York, NY, USA: ACM.

Russell, S. J., and Norvig, P. 2016. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of go with deep neural networks and tree search. *nature* 529(7587):484–489.

Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; et al. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.

Vinyals, O.; Ewalds, T.; Bartunov, S.; Georgiev, P.; Vezhnevets, A. S.; Yeo, M.; Makhzani, A.; Küttler, H.; Agapiou, J.; Schrittwieser, J.; Quan, J.; Gaffney, S.; Petersen, S.; Simonyan, K.; Schaul, T.; van Hasselt, H.; Silver, D.; Lillicrap, T. P.; Calderone, K.; Keet, P.; Brunasso, A.; Lawrence, D.; Ekermo, A.; Repp, J.; and Tsing, R. 2017. Starcraft II: A new challenge for reinforcement learning. *CoRR* abs/1708.04782.

Witteman, C. L.; Renooij, S.; and Koele, P. 2007. Medicine in words and numbers: a cross-sectional survey comparing probability assessment scales. *BMC Medical Informatics and Decision Making* 7(1):13.