Learning Logic Constraints from Demonstration

Mattijs Baert[1,*], Sam Leroux[1] and Pieter Simoens[1]

[1]IDLab, Department of Information Technology at Ghent University - imec, Technologiepark 126, Ghent, B-9052, Belgium

Abstract

Autonomous agents operating in real-world settings are often required to efficiently accomplish a task while adhering to certain environmental constraints. For instance, a self-driving car must transport its passengers to their intended destination as fast as possible while complying with traffic regulations. Inverse Constrained Reinforcement Learning (ICRL) is a technique that enables the learning of a policy from demonstrations of expert agents. When these expert agents adhere to the environmental constraints, ICRL thus allows for compliant policies to be learned without the need to define constraints beforehand. However, this approach provides no insight into the constraints themselves although this is desired for safety-critical applications such as autonomous driving. In such settings, it is important to verify what is learned from the given demonstrations. In this work, we propose a novel approach for learning logic rules that represent the environmental constraints given demonstrations of agents that comply with them, thus providing an interpretable representation of the environmental constraints.

Keywords

Constraint Inference, Learning from Demonstrations, Rule Induction

1. Introduction

Social norms play a crucial role in shaping individual behavior in modern society, promoting safety and efficiency in human interactions. Artificial agents seeking to integrate into the real world must also adhere to these norms in order to achieve success [1]. These norms can be viewed as constraints on an agent's behavior, and in the framework of reinforcement learning (RL), a constraint-abiding agent can be trained by solving a min-max problem [2], maximizing the reward function (reflecting the goal) while minimizing the cost function (capturing constraint violations). However, in complex environments where constraints are implicit or unknown, it may be necessary to use Inverse Constrained Reinforcement Learning (ICRL) methods to learn these constraints from expert demonstrations. Current ICRL methods iterate over the complete state-action space to determine the most likely constraints [3] or parameterize the cost function using a neural network [4, 5]. The first group of methods offers explainability as the constraints are represented by a set of states (or state-action pairs) [6, 7] or logic rules extracted from this set [8]. The second group of techniques offers scalability to complex problems but has to concede in terms of interpretability although this is crucial for safety-critical applications. In this work, we present a novel approach for obtaining an interpretable description of constraints in environments with a high-dimensional state space. First, we generate states that

NeSy 2023: 17th International Workshop on Neural-Symbolic Learning and Reasoning

*Corresponding author.

mattijs.baert@ugent.be (M. Baert); sam.leroux@ugent.be (S. Leroux); pieter.simoens@ugent.be (P. Simoens)

0000-0002-2413-5298 (M. Baert); 0000-0003-3792-5026 (S. Leroux); 0000-0002-9569-9373 (P. Simoens)

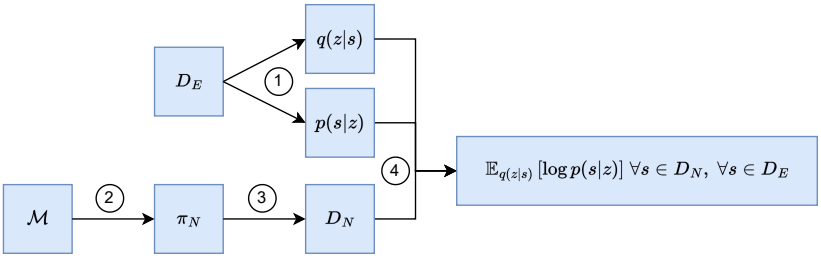


Figure 1: Generation of negative examples from expert trajectories given a model of the nominal environment

are likely to be constrained from trajectories of agents that adhere to the constraints (i.e. expert agents). Next, we utilize a rule induction method to learn a set of rules that capture the constraints in the environment from a set of positive examples (states visited by expert agents) and negative examples (states generated in the previous step). The final outcome is a set of logic rules in disjunctive normal form that represent the environmental constraints. An additional advantage of our method is that it is robust against constraint violations in the expert demonstrations. This is important when learning from human demonstrations because it is possible that some human demonstrations are non-compliant with the constraints.

2. Method

The aim of this study is to learn logical rules that describe the constraints in a specific environment by utilizing trajectories of constraint-abiding agents (i.e., experts) within that environment. Anomalous behavior is rare in real-world scenarios, leading to a limited number of constraint violations in the set of expert trajectories. Therefore, we regard the expert trajectories as a dataset containing only positive examples. However, many current classification and rule induction techniques necessitate both positive and negative examples for training. Therefore, we propose a method for generating negative examples (possible constrained states) given a model of the unconstrained environment (see Sec. 2.1). We acknowledge it is possible that the expert dataset also comprises negative examples, which we will address later in the paper. Once we obtain a dataset comprising both positive and negative labels, rules are learned that can differentiate between the positive and negative examples, i.e. it is possible to determine if an example is positive or negative by evaluating the example on the learned rules (see Sec. 2.2).

2.1. Generating Negative Examples

Figure 1 depicts the procedure for generating negative examples, the different steps are numbered and referred to from the text. In a first step, (1) we model the distribution of the states visited by the expert agents P_E. Since calculating P_E is intractable for all but very simple environments, a Variational AutoEncoder (VAE) is trained on the states visited during the expert trajectories D_E optimizing the evidence lower bound objective. The VAE consists of an encoder network modelling the posterior distribution $q(z|s)$ of a latent variable z given the observed state s. The decoder network maps z back to the state space using the likelihood distribution $p(s|z)$.

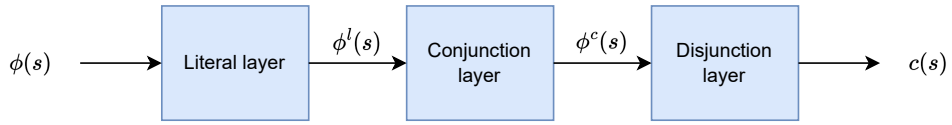
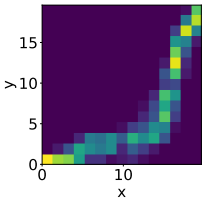


Figure 2: Neural-symbolic rule induction network

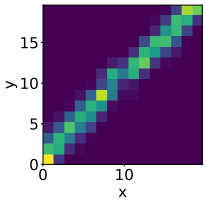
The reconstruction error can then be used as a measure of how likely the input originates from a distribution similar to P_E. We assume that a Markov Decision Process (MDP) $\mathscr{M}$ of the unconstrained environment is available, we refer to this as the nominal MDP. Given $\mathscr{M}$, the optimal nominal policy π_N is obtained using reinforcement learning (RL) (2). Next, a set of nominal trajectories D_N is sampled from the nominal policy π_N (3). States occurring in trajectories sampled from π_N are high-value states since π_N is optimal. When such a state results in a high reconstruction error when passed through the trained VAE, this means this state is not very likely to be visited by the expert. We reason there should be some constraints which prevents the expert from visiting this high-value state. Following this rationale, we identify possible constraints as high-value states which are not likely to be visited by the expert thus resulting in a high reconstruction error. At last, a labeled dataset is build by calculating the normalized reconstruction error for all states visited during trajectories sampled from both the nominal policy D_N and the expert trajectories D_E (4). Until now, our assumption was that the expert trajectories do not include any constraint violations. However, when gathering human trajectories, it is plausible that some constrained states are present in the obtained trajectories. In cases where the number of constraint violations are minimal, the impact on the learned distributions by the VAE is insignificant. Consequently, constrained states will still cause a substantial reconstruction error.

2.2. Rule Induction

In this study, we utilize a neural-symbolic architecture that is based on relational rule networks, as proposed by Kusters et al. [9]. This fully differentiable neural network can, after convergence, be interpreted as a logical formula in disjunctive normal form. An overview of the architecture is presented in Figure 2. The input of the network is a k-dimensional real-valued vector representation of the state, denoted as $\phi(s)$. The first layer, known as the *literal layer*, learns literals as hyperplanes dividing the feature space. The output of this layer is an L-dimensional vector, denoted as $\phi^l(s) \in [0,1]^L$, where each dimension corresponds to the evaluation of one of the L literals. The *conjunction layer* produces a C-dimensional vector, denoted as $\phi^c(s) \in [0,1]^C$, where each dimension is the result of a weighted conjunction of $\phi^l(s)$. Finally, the last layer takes a weighted disjunction of all values of $\phi^c(s)$ resulting in a value $c(s) \in [0,1]$ indicating if s is constrained. The use of this architecture allows us to learn using gradient descent while also having the ability to interpret the rules in a human-understandable format. We refer to appendix A for details on the implementation and values of hyperparameters.

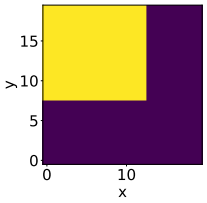


(a) Expert

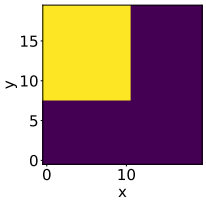


(b) Nominal

Figure 3: State visitation frequencies



(a) Ground truth



(b) Rule network

Figure 4: Constrained states

3. Preliminary Results

We perform a preliminary experiment on a simple navigation task in a continuous environment. The agent's goal is to navigate from the bottom left corner to the top right corner in as few steps as possible, but some part of the environment is inaccessible, e.g. reflecting a newly laid lane. The environment is depicted in figure 4a with the ground truth constraints (in yellow). The nominal policy reflects the *desire line* that an agent takes which does not adhere to the constraints. The nominal policy is learned using Proximal Policy Optimization (PPO) [10]. To obtain expert trajectories, we learn the true expert policy using Reward Constrained Policy Optimization (RCPO) [2] given the ground truth constraints. The expert and the nominal trajectories are depicted in figure 3a and 3b respectively. The state's vector representation $\phi(s)$ is a two dimensional vector which contains the x- and y-coordinates. Because of the simplicity of the environment we could iterate over the complete state space and visualize the classification boundary of the learned classifier (see fig 4b). The following rule is extracted from the network, defining constrained states:

$$x < 10.9 \wedge y > 7.6. \tag{1}$$

This corresponds with an intersection over union (IoU) of 0.86 with the ground truth constraints. We conclude that the learned rule is a good estimate of the ground truth constraints presented in figure 4a. Section B in the appendix provides additional results on the robustness against constraints violations by the expert agents.

4. Conclusion

In this work we outlined a novel method for learning behavioral constraints from expert demonstrations represented as a logical formula. This is the first method which is able to learn constraints in environments with a continuous state space while representing the learned constraints in an interpretable fashion. We presented preliminary results on a simple navigation task. In future work, we will validate our method on more complex environments with intricate constraints. This includes real-world traffic scenarios where demonstrations are obtained from human agents [11, 12, 13]. These datasets interface with CommonRoadRL [14] which can provide the nominal MDP (i.e. model of the unconstrained environment). We could extend this method to learning constraints in high-order logics by using neural-symbolic classifiers which can learn first-order logic [15, 16] or signal temporal logic formulae [17]. Another interesting

directive is on how these logic constraints can be transferred to an autonomous agent for guaranteeing constraints are never violated. One possibility is to augment the learned logic formulae on the policy network [18]. Another interesting use case is anomaly detection by validating observations on the learned rules.

Acknowledgments

This research was partially funded by the Flemish Government (Flanders AI Research Program).

References

[1] S. Russell, Human compatible: Artificial intelligence and the problem of control, Penguin, 2019.

[2] C. Tessler, D. J. Mankowitz, S. Mannor, Reward constrained policy optimization, in: International Conference on Learning Representations, 2019. URL: https://openreview.net/forum?id=SkfrvsA9FX.

[3] D. R. Scobee, S. S. Sastry, Maximum likelihood constraint inference for inverse reinforcement learning, in: International Conference on Learning Representations, 2020. URL: https://openreview.net/forum?id=BJliakStvH.

[4] S. Malik, U. Anwar, A. Aghasi, A. Ahmed, Inverse constrained reinforcement learning, in: International Conference on Machine Learning, PMLR, 2021, pp. 7390–7399.

[5] G. Liu, Y. Luo, A. Gaurav, K. Rezaee, P. Poupart, Benchmarking constraint inference in inverse reinforcement learning, in: The Eleventh International Conference on Learning Representations, 2023. URL: https://openreview.net/forum?id=vINj_Hv9szL.

[6] A. Gaurav, K. Rezaee, G. Liu, P. Poupart, Learning soft constraints from constrained expert demonstrations, in: The Eleventh International Conference on Learning Representations, 2023. URL: https://openreview.net/forum?id=8sSnD78NqTN.

[7] A. Glazier, A. Loreggia, N. Mattei, T. Rahgooy, F. Rossi, B. Venable, Learning behavioral soft constraints from demonstrations, in: Workshop on Safe and Robust Control of Uncertain Systems at the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), 2021.

[8] M. Baert, S. Leroux, P. Simoens, Inverse reinforcement learning through logic constraint inference, Machine Learning (2023) 1–26.

[9] R. Kusters, Y. Kim, M. Collery, C. d. S. Marie, S. Gupta, Differentiable rule induction with learned relational features, in: NeSy 22: 16th International Workshop on Neural-Symbolic Learning and Reasoning, 2022.

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms., CoRR abs/1707.06347 (2017). URL: http://dblp.uni-trier.de/db/journals/corr/corr1707.html#SchulmanWDRK17.

[11] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kümmerle, H. Königshof, C. Stiller, A. de La Fortelle, M. Tomizuka, INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps, arXiv:1910.03088 [cs, eess] (2019).

[12] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, L. Eckstein, The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections, in: 2020 IEEE Intelligent Vehicles Symposium (IV), 2020, pp. 1929–1934. doi:10.1109/IV47402.2020.9304839.

[13] R. Krajewski, J. Bock, L. Kloeker, L. Eckstein, The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems, in: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018, pp. 2118–2125. doi:10.1109/ITSC.2018.8569552.

[14] X. Wang, H. Krasowski, M. Althoff, Commonroad-rl: A configurable reinforcement learning environment for motion planning of autonomous vehicles, in: IEEE International Conference on Intelligent Transportation Systems (ITSC), 2021. doi:10.1109/ITSC48978.2021.9564898.

[15] H. Dong, J. Mao, T. Lin, C. Wang, L. Li, D. Zhou, Neural logic machines, in: International Conference on Learning Representations, 2019. URL: https://openreview.net/forum?id=B1xY-hRctX.

[16] R. Riegel, A. Gray, F. Luus, N. Khan, N. Makondo, I. Y. Akhalwaya, H. Qian, R. Fagin, F. Barahona, U. Sharma, et al., Logical neural networks, arXiv preprint arXiv:2006.13155 (2020).

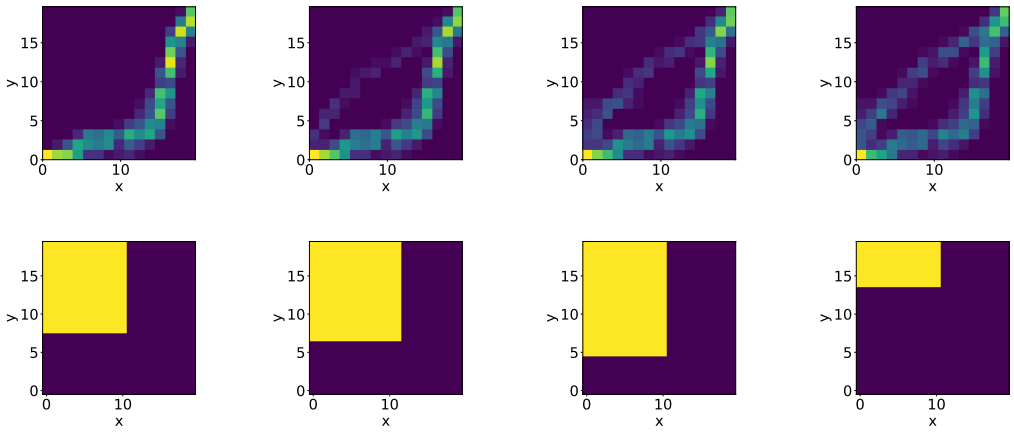
[17] R. Yan, A. Julius, Neural network for weighted signal temporal logic, arXiv preprint arXiv:2104.05435 (2021).

[18] K. Ahmed, S. Teso, K.-W. Chang, G. V. den Broeck, A. Vergari, Semantic probabilistic layers for neuro-symbolic learning, in: A. H. Oh, A. Agarwal, D. Belgrave, K. Cho (Eds.), Advances in Neural Information Processing Systems, 2022. URL: https://openreview.net/forum?id=o-mxIWAY1T8.

[19] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

A. Implementation Details

In this section, we elaborate on the implementation for the experiment presented in section 3. The VAE is configured with three linear layers each with a ReLU activation function. We train the VAE for 500 epochs using Adam optimizer [19] with a learning rate of 0.01 and a batch size of 64. For the relational rule net, we configure $L = 10$ (number of literal layers) and $C = 25$ (number of conjunctions). We train this network for 1500 epochs, using Adam optimizer with a learning rate of 0.001 and a batch size of 64. Other parameters are set to the values mentioned in the original paper [9]. A weighted random sampler is used to select training samples to ensure the network is trained on a balanced dataset because the number of valid states (low reconstruction error) is almost always larger then the number of constrained states encountered during the obtained trajectories,



(a) 0%, IoU=0.86 (b) 10%, IoU=0.81 (c) 20%, IoU=0.67 (d) 30%, IoU=0.40

Figure 5: The top panel of the figure shows the state visitation frequency of expert trajectories, where varying portions of expert agents ignore the constraints and take the desire line. The bottom panel displays the learned constraints obtained from the corresponding expert trajectories

B. Additional Results

We provide additional results on the robustness of our method against constraint violations by the expert. Figure 5 illustrates the learned constraints in cases where expert trajectories include a portion of trajectories from agents that ignore the constraints. The following rules were extracted from the network.

When 10% of the expert trajectories originate from agents ignoring the constraints:

$$x < 11 \wedge y > 6.2. \tag{2}$$

When 20% of the expert trajectories originate from agents ignoring the constraint:

$$x < 10 \wedge y > 4.4. \tag{3}$$

When 30% of the expert trajectories originate from agents ignoring the constraint:

$$x < 10 \wedge y > 13.8. \tag{4}$$

In conclusion, our method has demonstrated the capability to effectively learn the appropriate constraints from expert trajectories, even when up to 20% of the trajectories violate these constraints. This finding suggests the robustness of our approach and its potential utility for learning from human experts.