Structure Mapping for Transferability of Causal Models

Anonymous Authors[1]

Abstract

Human beings are capable to build rich causal models. They use causal models to transfer the acquired knowledge to similar domains in the form of abstract causal concepts. Humans also learn multiple representations of a single object which characterize the perceptual and causal properties of different objects in the system. The multiple representations help in transferring the knowledge using the combination of perceptual features and causal dynamics. Using this intuition behind how humans learn causal models and constantly use them in new environments, we introduce a transfer-learning framework which uses object-oriented representations to learn the causal relationships between objects in one domain and map the causal relationships across domains for transfer learning. More specifically, we use structure learning techniques to explicitly learn cause and effects of actions in the reinforcement learning domain and use structure mapping to map the learned causal relationships to transfer to the variants of the game with exchangable perceptual features among objects but similar causal dynamics. We demonstrate the feasibility of our approach on a gridworld setting by combining *causal model-based approach* with *model-free approaches* in reinforceent learning.

1. Introduction

In reinforcement learning, two set of approaches are common - *model-free* learning and *model-based* learning. In model-free learning, the agent learns to directly estimate the future reward for different states without building an explicit model of the system while in model-based learning the agent explictly learns the state transition and reward distributions of the system. In model based learning, the model can be prediction-based model where the goal is to accurately predict the future state or it can be a causal model where the goal is to explictly model the variables (causes) responsible for the state transition and their effects. Experimental work in cognitive science has shown that humans and animals use a combination of model-based and model-free algorithms, implicating the co-existence of two "systems" in the brain. (Balleine Dickinson, 1998; Daw, Niv, Dayan, 2005; Dolan Dayan, 2013).

Model-free approaches are more energy-efficient and favorable when agent needs to act quickly using less accurate predictions but model-free approaches don't leverage previous experience and transfer poorly to similar environments with slight variations. Model-based approaches allow the agent to use the prior experiences, simulate the possible future trajectories and act based on the model. But model-based approaches might suffer from large errors if the model is inaccurate. The combination of the model-free and model-based approaches help in leveraging the strengths of both of the models.

Building explicit causal models for the systems provide the advantage of explicit understanding of the dynamics of the system and re-using the causal knowledge to transfer to the variations of the system. For example, while learning a new video game, we first learn about different entities of the game, their perceptual features (e.g. color, shape, material etc.), causal characteristics (like what will happen if you hit a block of wall) and causal relationships between different objects (e.g. switching on the switch causes bulb to light.) Now, if we are presented with a new variant of the game with differently looking objects but similar causal characteristics then we tend to transfer knowledge by mapping the objects seen in the previous game with the objects in the new game based on similar behavior. On the other hand, if we are presented with a new variant of the game with similarly looking objects but different causal characteristics, then we need to augment the causal model of those objects with new causal powers. In either case, multiple characterization of the objects in terms of their perceptual features as well as causal properties provide several advantages. First, this characterization allows the flexibility to decouple the perceptual representation of an object from their causal properties, allowing

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

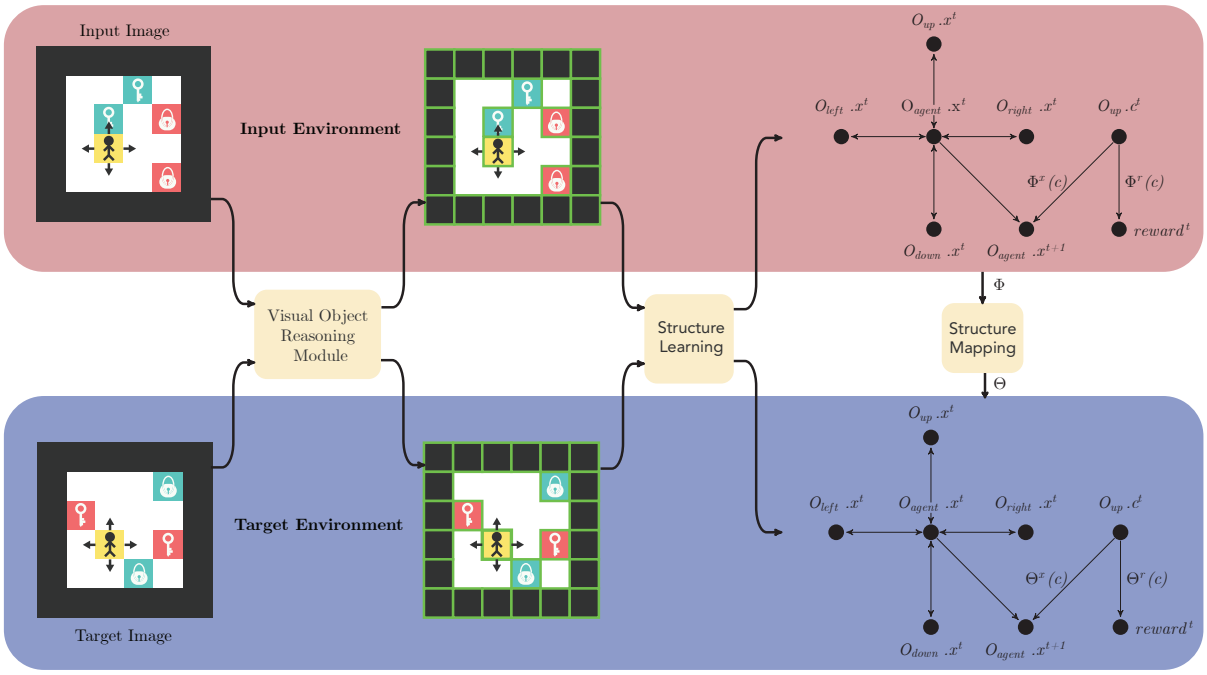

Figure 1. High-level schematic diagram to show the overall approach for transfer learning

combinatorial generalization over systems with different combination of these characteristics. Second, it allows to build abstract causal knowledge in the form of high-level causal concepts. (e.g. Hitting a block of wall causes no movement while wall can be red, blue or black in color). [(Lake et al., 2016), (Kemp et al., 2010), (Lake et al., 2015)].

Object-oriented MDPs (Diuk et al., 2008) provides a structured way to model the environments naturally in the form of the objects and object interactions as we perceive them. It is an efficient way of modelling as it factorizes the state space over different objects and thus, reduces large state-space. Also, it provides an abstraction by attributing the effect of the objects to object's characteristics rather than a property of the specific location in the state space, which is the case in pixel-based representation (Assuming the model is trained on images). We adopt object-oriented formalism in this work as it is more convenient to depict causal relationships between the objects.

In this work, we combine ideas from object-oriented MDPs (Diuk et al., 2008), structure learning (Zheng et al., 2018) and structure mapping theory (Gentner, 1983) for transfer learning . We learn object-oriented representations and use continuous-optimization based structure learning algorithm (Zheng et al., 2018) to discover the causal structure of the environment. Discovered causal structure and associated perceptual attributes of the object allow us to characterize descriptive and causal characteristics of the object. We learn a mapping from objects in the source domain to the target domain based on the causal structure mapping which guides the transfer of policy from the source domain to the target domain.

2. Preliminaries and Problem Statement

2.1. Notation

The traditional formalism for the reinforcement learning problem is the Markov Decision Process (MDP). An MDP is a five-tuple $(\mathcal{S}, \mathcal{A}, T, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is the set of actions, $T(s^{(t+1)}|s^{(t)}, a^{(t)}$ is the probability of transforming from state $s^{(t)} \in \mathcal{S}$ to $s^{(t+1)} \in \mathcal{S}$ after action $a^{(t)} \in \mathcal{A}$, $\mathcal{R}(r^{(t+1)}|s^{(t)}, a^{(t)})$ is the probability of receiving the reward $r^{(t+1)\in R}$ after executing action $a^{(t)}$ while in state $s^{(t)}$, and $\gamma \in [0, 1]$ is the rate at which future rewards are exponentially discounted.

In object-oriented MDP, we assume that the state consists of N objects (or entities) and each object has M attributes.**[Is this assumption important for the method?]**. O_i refers to the i^{th} object and let $\alpha_{i,j}^{(t)}$ refer to the j^{th} attribute value of the i^{th} object at time t. $O_i^{(t)} = (\alpha_{i,1}^{(t)}, \ldots \alpha_{i,M}^{(t)})$ to refer

to the state of the i^{th} object at time t. The complete state of the MDP is modeled by the network at time t is then $s^{(t)} = (O_1^{(t)} \dots O_N^{(t)})$. Example of an attribute α_{ij} is {color, x-position, y-position etc}.

2.2. Strcuture Learning in Object Oriented-MDPs

The problem of structure learning in OO-MDPs can be formulated as learning a directed acyclic graph $G^a = (V, E)$ with object attributes as vertices $V \in R^{N \times M}$ and $E \in R^{NM \times NM}$.

We assume that the edges between the attributes in the time stamp (e.g. $\alpha_{ip}^{(t)}$ and $\alpha_{jq}^{(t)}$) are allowed but are interpreted as more of relational dependencies (e.g. spatial relationship between x-position of the up neighbor and x-position of the agent) rather than a causal dependency. Although it might be possible that instantaneous causal effects might be allowed in some settings. Please note that in OO-MDPs each G^a is specific to an action a in the action space because difference actions might result in different causes for state transitions, resulting in different graphs. For example, if an agent takes an up action, then the only attributes corresponding to the object in the up action might be valid causes for the state transition of the agent.

We use the continuous optimization for structure learning framework (NOTEARS) provided by (Zheng et al., 2018) to learn G^a.

Let's represent the current state and next state of attributes as $X = \{\alpha_{11}^{(t)}, \alpha_{12}^{(t)} \dots \alpha_{NM}^{(t)}, \alpha_{11}^{(t+1)}, \alpha_{12}^{(t+1)}, \dots \alpha_{NM}^{(t+1)}\}$ and $d = NM$, dimesnion of X.
The optimization problem is formulated as below:

$$min_f l(f)$$

subject to

$$G^a(f) \in DAG$$

where $L(f) = \frac{1}{n} \sum_{j=1}^{d} l(x_j, f_j(X))$

Assume that $f_j : R^d \to R$ and l is a loss function like MSE, L1 loss etc. In this work, we approximate f using parametric family of neural networks represented by parameters Θ. Thus, optimization problem becomes:

$$min_\theta \frac{1}{n} \sum_{j=1}^{d} l(x_j, f(X, \theta))$$

subject to

$$h(W(\theta)) = 0$$

where $W(f) = W(f_1, f_2 \dots f_d) \in R^{d \times d}$ encodes the graph edges $G^a(\Theta)$, i.e. $[w(f)]_{kj} = \partial_k f_{j\,L_2}$ and $h(W(\theta)) = tre^{W \cdot W} - d$ encodes the acyclicity constraint. Refer (Zheng et al., 2018) and (Zheng et al., 2020) for more details.

This approach provides several advantages compared to the integer-programming based structure learning used in Schema Networks for similar OO-MDP setting (Kansky et al., 2017). First, the representation for attributes used by the schema networks was binary to allow for integer programming based optimization. This limits the scope of their approach as binary representation might be computationally challenging for the attributes with high dimensions or magnitudes (e.g. positions). NOTEARS, on the other hand provide no such restriction and allows for using continuous and discrete variables for the structure learning. Second, NOTEARS can learn state-space abstraction in the structure by capturing relational dependencies. For example, if two objects interact only when they are adjacent to the other, then interaction between attributes of different objects is learned in the functional form of $f_k(j)$ irrespective of the absolute positions of the objects. An example adjacency matrix

2.3. Simulation of Trajectories using Causal Dynamics Model

Pseudocode for learning policy using causal dynamics model is described in 1 and high-level idea is explained below.

1. Let's assume that the agent starts exploring the environment using a random policy for initial T_0 timesteps.
2. Agent learns the structural model G^a for each action using collected (state, action, next_state, reward) tuples for T_0 timesteps. More specifically, it learns sparse functional mappings $f_j(X)$ for each attribute x_j where X is the input vector as defined in Section 2.2.
3. Simulate the set of the trajectories from the learned causal dynamics model and learn the optimal policy by choosing set of actions which lead to the positive rewards, either by updating Q function or heuristically searching for trajectories which might lead to the positive rewards.

2.4. Structure Mapping in target environment

[TO DO]

3. Experimental Evaluation

3.1. Evaluation Environment - Triggers

We provide a proof-of-concept for our approach using a toy-environment (known as Triggers), first introduced in

Algorithm 1 Online Planning using Causal Dynamics Model

Input: Initialize $Q(s,a)$ for all $s \in S$ and for all $a \in A$, initial policy π, lr α, gamma γ
repeat
  $S \leftarrow$ Current state in object-oriented representation.
  $A \leftarrow \epsilon - greedy(S,Q)$
  Execute action A and observe next state S' and reward R
  $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma max_a Q(S',a) - Q(S,A)]$
  Update Causal Dynamics Model with (S,A,R,S') (assuming determinstic environment)
  repeat
    $S \leftarrow$ randomly previously observed state
    $A \leftarrow$ random action previously taken in S
    $R, S' \leftarrow$ Model(S,A)
    $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma max_a Q(S',a) - Q(S,A)]$
  until n steps
until T steps

(Ferret et al., 2019). In this gridworld based environment, the agent can take one of the four possible actions $\{north, south, east\ or\ west\}$. Hitting black walls doesn't change the agent's position and moving in the free space increases or decreases the agent's x-position or y-position by 1, depending on the action. In source environment, green colored boxed represent the keys and red-colored boxes represent the boxes. The agent needs to collect **all** the keys before it can attempt opening one of the doors. If the agent attempts to open the door even if one of the keys is there, agent received negative reward -1. If the agent succesfully opens each door, it recieves $+1$ reward. The key boxes disappear when collected and the door boxes disappear when opened succesfully. In one of the target enviroments, we switch the colors of the keys and doors. The goal is to transfer the knowledge from source to target game by mapping the behavior of objects (keys and doors) and learn efficiently.

The reason behind choosing this environment as proof-of-concept is (1) it is a simple environment which contains different types of entities with each having different descriptive features and characteristic causal behavior (agent, walls, free space, keys and doors) (2) The credit assignment can be well-evaluated in this setting as discussed in (Ferret et al., 2019) because getting rewards are attributed to collecting keys. In model-free environments, credit assignment is one of the ways to backtrack to the important states to understand the reasons behind the agent's behavior. Building an explicit causal model is an alternative way which directly tries to reason the effects of actions beforehand and identify the causes for observing the states.

3.2. Structure Learning

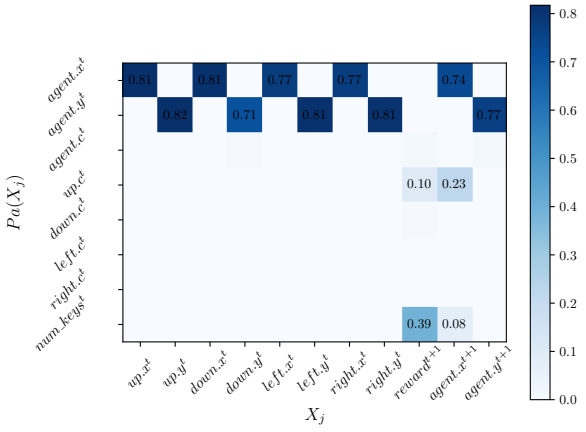

Figure 2. L_2 norm $\|\Phi_j(X)\|_{L_2}$ of Weight Values for learned bayesian graph G^{up} for action 'UP'. Attributes on y-axis are potential parents (causes) of the corresponding j_{th} attribute on x-axis (X_j), with parameters Φ_j. Positive values of weights indicate edge between parents and children.

Figure 2 shows the results of the structure learning algorithm NOTEARS for learning parameteric DAGs using Neural Networks. The results are only shown for graph learned for action UP. For rest of the actions (DOWN, LEFT and RIGHT), similar results were observed and are included in Appendix (Cite appendix). We trained the algorithm on the Trigger environment with different size of the state space (by varying grid's width and height from 5 to 75) and different numbers and locations of the keys and doors. num_keys is the variable which represents the number of keys present in the environment at each time step.

In Figure 2, we observe that the attribute future reward r^t has total number of keys (num_keys) and color of the 'up' neighbor ($up.c^t$) as the parents, as expected. For agent's next positions x^{t+1} and y^{t+1} , agent's current x and y positions and color of the 'up' neighbor are the causes. We also see dependecies between x and y positions of the neighbors and agent due to spatial dependency between them. Assuming agent as the reference of frame, relational edge can be assumed as directed edge from agent to the neighbor.

3.3. Agent's Performance in Source Environment

Figure 4 shows the performance of trained tabular-Q learning agent for 100 episodes averaged over 100 trials on Triggers environment with two keys and two doors. **[TO DO: Include performance graph for Causal Dynamics Model -Q agent]**

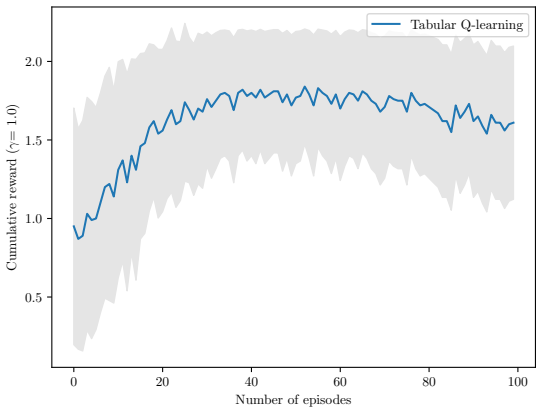

Figure 3. Baseline performance of tabular-Q learning agent

3.4. Transfer to Target Environment

Figure 4 shows the performance of trained tabular-Q learning agent for 100 episodes averaged over 100 trials on Triggers environment with two keys and two doors. **[TO DO: Include performance graph for Causal Dynamics Model -Q agent]**

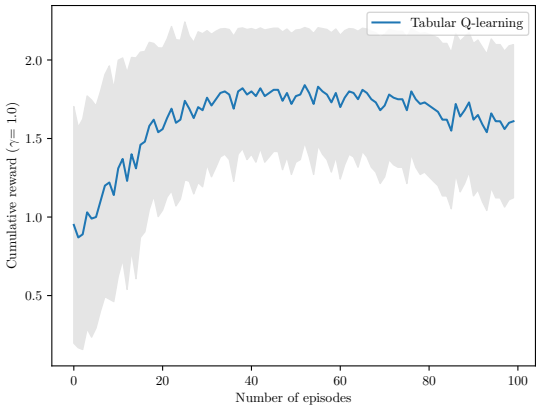

Figure 4. Baseline performance of tabular-Q learning agent

4. Related Work

Structure learning is crucial to discover the causal structure from data. Recent advances in the structure learning algorithms (Zheng et al., 2018), (Zheng et al., 2020) has allowed formulating the problem of structure learning as continuous optimization problem, making structure learning more efficient than other structure learning algorithms for which search space is combinatorial and computationally expensive.

Structure mapping theory (Gentner, 1983) provides formalism for analogical reasoning by mapping relations from one domain to the another domain. The theory can also be used in transfer learning tasks for reinforcement learning where analogical mapping between the objects based on the causal behavior can provide an efficient way to transfer knowledge. It epmhasizes on the structural constraints on the mapping. In particular, mapping based on the higher-order relationships (e.g. relational dependencies among objects) are preferred over first-order relationships (common attributes of the object). (Lee & Holyoak, 2008) discuss that structure mapping based on the causal relationships can lead to the better strength in analogical inference as compared to pure relational dependencies.

5. Conclusion

TO DO

References

Diuk, C., Cohen, A., and Littman, M. L. An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 240–247, 2008.

Ferret, J., Marinier, R., Geist, M., and Pietquin, O. Self-attentional credit assignment for transfer in reinforcement learning, 2019.

Gentner, D. Structure-mapping: A theoretical framework for analogy. *Cognitive science*, 7(2):155–170, 1983.

Kansky, K., Silver, T., Mély, D. A., Eldawy, M., Lázaro-Gredilla, M., Lou, X., Dorfman, N., Sidor, S., Phoenix, S., and George, D. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics, 2017.

Kemp, C., Goodman, N. D., and Tenenbaum, J. B. Learning to learn causal models. *Cognitive science*, 34 7:1185–243, 2010.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. Building Machines That Learn and Think Like People. *arXiv e-prints*, art. arXiv:1604.00289, Apr 2016.

Lee, H. S. and Holyoak, K. J. The role of causal models in analogical inference. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 34(5):1111, 2008.

Zheng, X., Aragam, B., Ravikumar, P., and Xing, E. P. DAGs with NO TEARS: Continuous Optimization for Structure Learning. In *Advances in Neural Information Processing Systems*, 2018.

Zheng, X., Dan, C., Aragam, B., Ravikumar, P., and Xing, E. P. Learning sparse nonparametric DAGs. In *International Conference on Artificial Intelligence and Statistics*, 2020.