Causal Navigation by Continuous-time Neural Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Imitation learning enables high-fidelity, vision-based learning of policies within rich, photorealistic environments. However, such techniques often rely on traditional discrete-time neural models and face difficulties in generalizing to domain shifts by failing to account for the causal relationships between the agent and the environment. In this paper, we propose a theoretical and experimental framework for learning causal representations using continuous-time neural networks, specifically over their discrete-time counterparts. We evaluate our method in the context of visual-control learning of drones over a series of complex tasks, ranging from short- and long-term navigation, to chasing static and dynamic objects through photorealistic environments. Our results demonstrate that causal continuous-time deep models can perform robust navigation tasks, where advanced recurrent models fail. These models learn complex causal control representations directly from raw visual inputs and scale to solve a variety of tasks using imitation learning.

1 Introduction

Unlike machine learning systems, natural learning systems excel at generalizing learned skills beyond the original data distribution. This is due to the mechanisms they deploy during their learning process, such as active use of closed-loop interventions, accounting for distribution shifts, and the temporal structures [67, 10]. These factors are largely disregarded or are engineered away in the development of modern machine learning systems.

To take the first steps towards resolving these issues, we can investigate the spectrum of causal modeling [53]. At one end of the spectrum, there are physical models of agents and environments described by differential equations. These physical models allow us to investigate interventions, predict future events using past information, and can describe the statistical dependencies in the system.

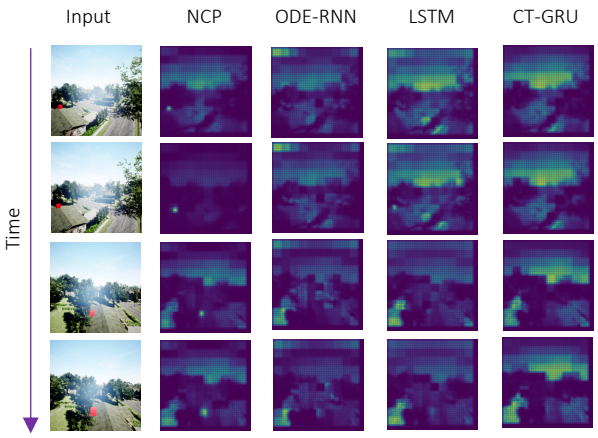

Figure 1: **Causal navigation from raw visual inputs**. Given a sequence of raw RGB inputs (left) a drone is trained to navigate towards the red-cube target. We visualize the saliency maps (right) for each model. Neural circuit policies [35] (a specific representation of CT-RNNs) can learn causal relationships (i.e., attend to the red-cube) directly from data while other models fail to do so. ODE-RNNs [59], LSTM [26] and CT- Gated Recurrent Units [43]. Saliency maps are computed by the visual backprop algorithm [3].

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

On the other end of the spectrum, statistical models allow us to construct dependencies and make predictions given independent and identically distributed (i.i.d.) (see Fig. 2). While physical models provide complete descriptions, it is intractable to define the differential equation systems that effectively model high-dimensional and complex sensory data such as vision data. For long, causal modeling frameworks aims at bridging this gap to extract statistical dependencies while constructing causal graphs [69, 49] or structural causal models[67] to intervene and explain relationships.

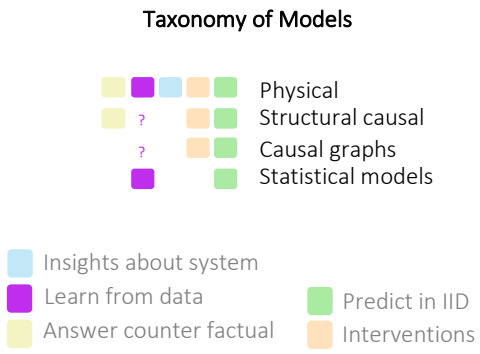

Figure 2: Taxonomy of models in the spectrum of causality. Figure data is taken from [53].

In this paper, we aim to use continuous-time (CT) neural networks [17, 8, 24] equipped with causal structures in order to get closer to the properties of physical models. In particular, we look into different representations of CT networks to see under what conditions and formulation they can form a causal model. We discover that the class of liquid time-constant networks (LTCs) [24] which are expressive continuous-time models constructed by bilinear approximation of neural ordinary differential equations [8], satisfy the properties of a causal model. The uniqueness of their solution and their ability to capture interventions make their forward- and backward- mode causal. Therefore, they can impose inductive biases on their architectures to learn robust and causal representations [35]. In this paper, we analyze how certain continuous-time (CT) neural networks are dynamical causal models. Furthermore, to justify the theoretical results, we empirically validate that these properties scale to high-dimensional visual input data in complex learning environments. We propose a series of control and navigation tasks for an end-to-end autonomous flight across various levels of complexity and temporal reasoning. We observe that traditional deep-learning models are capable of solving this task on offline, passive datasets but fail when deployed in closed-loop, active testing settings [10]. On the other hand, we find that only LTC-based models are able to complete the tasks in closed-loop interaction with the environments by capturing the true causation of the tasks during learning.

Specifically, consider a visual navigation task wherein a drone agent should learn to fly from point A to a target fixated at a point B, given only a sequence of raw visual inputs. The mapping between incoming pixels to the temporal and physical structure of the navigation is causal if we can explain and intervene at each step to observe how navigation decisions are decided based on the input pixels. We observe that in a neighborhood environment, agents based on LTCs (such as Neural Circuit Policies [35]) learn to stably attend to the target at point B throughout the task's time horizon (Fig. 1).

Therefore, *causes* of navigation to point B at a next time step is the agent's learned attention profile on the input images. These causal mappings are not present in other deep models, as shown in Fig. 1. In this paper, we show how LTCs' ability to capture causal structures directly from visual inputs results in improved robustness through generalization as well as interpretable decision making.

Summary of Contributions.

1. Theoretical evidence for the capability of CT neural networks in learning causal structures;
2. Extensive experimental results supporting the effectiveness of CT causal models in high-dimensional visual drone navigation tasks with varying memory-horizons; and
3. Robustness analysis of the CT causal models in closed loop-testing within real-world scenarios.

2 Related Works

In this section, we describe research works that are closely related to the core findings of the paper.

Causal Learning – The dominant approach toward learning causal models are graphical methods [64, 61], which try to model cause-effect relationships as a directed graph [49, 51]. Bayesian networks further combine graphical models with Bayesian methods [61, 32] to decompose the learning problem into a set of Bayesian inference sub-problems. [72] showed the effectiveness of such an approach for learning causal structures in nonlinear time-series via a set of linear models. Continuous-time

Bayesian networks further adapted the idea to modeling cause-effect relationships in time-continuous processes [46, 47, 19, 48]. A different approach for causal modeling of time-continuous processes is to learn ODEs, which under certain conditions imply a structured causal model [60]. zIn this work, we describe a class of continuous models that has the ability to account for interventions and therefore captures the causal structures from data.

Continuous-time Models – CT models manifest a large range of benefits compared to discretized deep models. They can perform adaptive computations through continuous vector fields realized by advanced ODE solvers [8]. They are strong in modeling time-series data and realize memory and parameter efficiency [8, 35]. A large number of alternative approaches have tried to improve and stabilize their training, namely the adjoint method [18], use neural ODEs in complex time-series prediction tasks [59, 36] and to characterize them better for inference and density estimation tasks [13, 14, 28, 23, 27, 55, 39]. In this work, we prove an important property of the CT network: We show that a bilinear approximation of Neural ODEs can give rise to expressive causal models.

Imitation Learning – Imitation learning describes the task of learning an observation-action mapping from human demonstrations [66]. This objective can either be achieved via behavior cloning, which directly learns from observation-action pairs, or indirectly via inverse reinforcement learning [45] which first constructs a reward function from an optimal policy. The most dominant behavior cloning paradigm is based on the DAgger framework [57], which iterates over the steps of expert data collection, supervised learning, and cloned policy evaluation. State-aware imitation learning [68] further adds a secondary objective to the learning task to bias the policy towards states where more training data is available. Recently, imitation learning methods have been adapted to domains where the environment dynamics of the expert and learned policy mismatch [11].

More recent imitation learning one-shot methods pre-train policies via meta-learning to adapt to a task, such that task-specific behavior can be cloned with as little as a single demonstration [12, 74]. Alternatively, generative adversarial imitation learning phrases the behavior cloning problem as a min-max optimization problem between a generator policy and discriminator classifier [25]. [2] extended the method by making the human expert policy end-to-end differentiable. The method has been further adapted to imperfect [73] and incomplete demonstrations [70].

Visual Navigation – Cognitive mapping and planning [21, 22] addresses the problem of learning to navigate from visual input streams by constructing a map of the environment and plan the agent's actions to achieve a given goal. [7] adapted the approach for the goal of exploring and mapping the environment. The alternative method represents the map in the form of a graph [65]. Neural SLAM (simultaneous location and mapping) shares many characteristics of cognitive mapping and planning [5] but takes one step further in separating mapping, pose estimation, and goal-oriented navigation. Target driven navigation, where the agent must find a target in an unknown environment, has seen successes using both semantic segmentation [42], and neural SLAM [6].

3 Problem Setup

In this section, we describe necessary concepts to formally derive the main results of our report.

3.1 Causal Structures

In a structural causal model (SCM), given a set of observable random variables $X_1, X_2, \ldots, X_n$, as vertices of a directed acyclic graph (DAG), we can compute each variable from the following assignment [67]:

$$X_i := f_i(\mathbf{PA}_i, U_i), \quad i = 1, \ldots, n. \tag{1}$$

Here, f_i is a deterministic function of the parents of the event, X_i, in the graph ($\mathbf{PA}_i$) and of the stochastic variable, U_i. One can intuitively think of the causal structure framework as a function estimation problem rather than in terms of probability distributions [69]. Direct causation is implied by direct edges in the graph through the assignment described in Eq. 1. The stochastic variables $U_1, \ldots, U_n$ ensure that a joint distribution $P(X_i|\mathbf{PA}_i)$ is constructed as a general objective [50, 67].

The SCM framework enables us to explore through the known physical mechanisms and functions to build flexible probabilistic models with *interventions*, replacing the "slippery epistemic probabilities, $P(X_i, \mathbf{PA}_i)$, with which we had been working so long" in the study of machine learning systems [49, 67]. *Interventions* can be formalized by the SCM framework, as operations that alter a subset of

properties of Eq. 1. For instance, modifying U_i, or replacing f_i (and as a result X_i) [30]. Moreover, by assuming joint independence of U_is, we can construct *causal* conditionals known as causal (disentangled) factorization, as follows [67]:

$$p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i | \mathbf{PA}_i). \tag{2}$$

Eq. 2 stands for the causal mechanisms by which we can model all statistical dependencies of given observables. Accordingly, Causal learning involves the identification of the causal conditionals, the function f_i, and the distribution of U_is in assignment Eq. 1.

3.2 Differential Equations Can form Causal Structures

Physical dynamics can be modeled by a set of differential equations (DEs). DEs allow us to predict the future evolution of a dynamical system and describe its behavior as a result of interventions. Their coupled time-evolution enables us to define averaging mechanisms for computing statistical dependencies [53]. A system of differential equations enhances our understanding of the underlying physical phenomenon, explains its behavior, and dissects its causal structure.

For instance, consider the following system of DEs: $\frac{d\mathbf{x}}{dt} = g(\mathbf{x}), \;\; \mathbf{x} \in \mathbb{R}^d$, with initial values at x_0, where g is a nonlinear function. The Picard-Lindelöf theorem [44] states that a differential equation of the form above would have a unique solution as long as g is Lipschitz. Therefore, if we unroll the system to infinitesimal differentials using the explicit Euler method, we get: $\mathbf{x}(t+\delta t) = \mathbf{x}(t) + dt f(\mathbf{x})$. This representation under the uniqueness condition shows that the near future events of x are predicted using its past information, thus, forming a causal structure.

Thus, a DE system is a causal structure that allows us to process the effect of interventions on the system. On the other side of the spectrum of causal modeling [53], pure statistical models allow us to learn structures from data with little insight about causation and associations between epiphenomena. Since causality aims to bridge this gap, in this paper, we propose to construct causal models with continuous-time neural networks.

3.3 Time-Continuous Neural Networks

Time-continuous networks are a class of deep learning models with their hidden states being represented by continuous ordinary differential equations (ODEs) [17]. The hidden state $x(t)$ of a neural network f is computed by the solution of the initial value problem below [8]:

$$\frac{dx}{dt} = f(\mathbf{x}(t), t, \theta), \quad \mathbf{x} \in \mathbb{R}^d, \tag{3}$$

where f is parametrized by θ. CT models enable approximation of a class of functions which we did not know how to generate otherwise [8, 24]. They can be used in both inference and density estimation with constructing a continuous flow to model data, efficiently [20, 58, 13]. CT models can be formulated in different representations. For instance, to achieve stability, CT recurrent neural networks (CT-RNNs) were introduced in the following form [17]:

$$\frac{d\mathbf{x}(t)}{dt} = -\frac{\mathbf{x}(t)}{\tau} + f(\mathbf{x}(t), t, \theta), \tag{4}$$

where the term $-\frac{\mathbf{x}(t)}{\tau}$ derives the system to equilibrium with a time-constant τ. To increase expressivity [56], liquid time-constant networks (LTCs) with the following representation can be used [24]:

$$\frac{d\mathbf{x}(t)}{dt} = -\Big[\frac{1}{\tau} + f(\mathbf{x}(t), t, \theta)\Big] \odot \mathbf{x}(t) + f(\mathbf{x}(t), t, \theta) \odot A. \tag{5}$$

In Eq. 5, $\mathbf{x}^{(D\times 1)}(t)$ is the hidden state of an LTC layer with D cells, $\tau^{(D\times 1)}$ is the fixed internal time-constant vector), $A^{(D\times 1)}$ is an output control bias vector, and $\odot$ is the Hadamard product. Here, the ODE system follows a bilinear dynamical system [52] approximation of Eq. 3 to construct an input-dependent nonlinearity in the time-constant of the differential equation. This was shown to significantly enhance the expressive power of continuous-time models in robotics and time-series prediction tasks [24, 35]. In the following, we show how CT models can be designed as causal structures to perform more interpretable real-world applications.

4 Results

In this section, we describe the main results. We first explain how a continuous-time model identified by Eq. 3, standalone, cannot satisfy the causal structure properties [53] even under Lipschitzness of its network. We then show that the class of liquid time-constant networks can enable causal modeling.

4.1 Causal Modeling with Continuous-time Networks

Neural ODEs are not causal models. Let f be the nonlinearity of a continuous-time neural network. Then the learning system defined by Eq. 3 cannot account for interventions (change of the environment conditions), and therefore does not form a causal structure even if f is Lipschitz-continuous.

To describe this in detail, we unfold the ODE by infinitesimal differentials as follows:

$$\mathbf{x}(t+\delta t) = \mathbf{x}(t) + dt f(\mathbf{x}, \theta). \tag{6}$$

If f is Lipschitz continuous, based on Picard-Lindelöf's existence theorem, the trajectories of this ODE system are unique and thus invertible. This means that, at least locally, the future events of the system can be predicted by its past values. Since the transformation is invertible, this setting is also true if we run the ODE backward in time (e.g., during the training process).

When the ODE system is trained by maximum likelihood estimation, given an initial weight distribution, the statistical dependencies between the system's variables might emerge from data. The resulting statistical model can predict in i.i.d. setting and learn from data, but it cannot predict under distribution shift or implicit/explicit interventions. Furthermore, The system cannot answer counterfactual questions [41].[1] Although Neural ODEs in their generic representation cannot form causal models, their other forms can help design a causal model.

LTCs are dynamic causal models. LTCs described by Eq. 5 resemble the representation of a simple Dynamic Causal Model (DCM) [16] with a bilinear Taylor approximation [52]. DCMs aim to extracting the causal architecture of a given dynamical systems. DCMs represent the dynamics of the hidden nodes of a graphical model by ODEs, and allow for feedback connectivity structures unlike Bayesian Networks [16]. A simple DCM can be designed by a second-order approximation (bilinear) of a given system such as $d\mathbf{x}/dt = F(\mathbf{x}(t), \mathbf{I}(t), \theta)$, as follows [16]:

$$d\mathbf{x}/dt = (A + \mathbf{I}(t)B)\mathbf{x}(t) + C\mathbf{I}(t) \tag{7}$$

$$A = \frac{\partial F}{\partial \mathbf{x}(t)}\Big|_{I=0}, \;\; B = \frac{\partial^2 F}{\partial \mathbf{x}(t)\partial \mathbf{I}(t)}, \;\; C = \frac{\partial F}{\partial \mathbf{I}(t)}\Big|_{x=0},$$

where $\mathbf{I}(t)$ is the inputs to the system, and $\mathbf{x}(t)$ is the nodes' hidden states. A fundamental property of a DCM representation is its ability to capture both internal and external causes on the dynamical system (interventions). Matrix A stands for a fixed internal coupling of the system. Matrix B controls the impact of the inputs on the coupling sensitivity among the network's nodes (controlling internal interventions). Matrix C embodies the external inputs' influence on the state of the system (controlling external interventions).

DCMs have shown promise in learning the causal structure of the brain regions in complex time-series data [52, 4, 29]. DCMs can be extended to a universal framework for causal function approximation by neural networks through the LTC neural representations [24].

Proposition 1. *Let f be the nonlinearity of a given LTC network identified by Eq. 5. Then the learning system identified by Eq. 5 can account for internal and external interventions by its weight parameters $\theta = \{W_r^{(D\times D)}, W^{(D\times m)}, b^{(D\times 1)}\}$, for D LTC cells, and input size m and $A^{(D\times 1)}$, and therefore, forms a dynamical causal model, if f is Lipschitz-continuous.*

Proof. To prove this, we need to show two properties for the LTC networks: 1) Uniqueness of their solution 2) Existence of intervention coefficients:

Uniqueness. By using the Picard-Lindelöf theorem [44], it was previously shown that for an $F : \mathbb{R}^D \to \mathbb{R}^D$, bounded C^1-mapping, the differential equation:

$$\dot{x} = -(1/\tau + F(x))x + AF(x),$$

[1] A counterfactual question describe a causal relationship of the form: "If X had not occurred, Y would not have occurred [40]"

has a unique solution on $[0, \infty)$. (Full proof in Lemma 5 of [24]).

Interventions Coefficients. Let f be a Lipschitz-continuous activation function such as tanh, then $f(\mathbf{x}(t), \mathbf{I}(t), t, \theta) = tanh(W_r\mathbf{x} + W\mathbf{I} + b)$. If we set $x = 0$, in Eq. 5, then the external intervention coefficients C in Eq. 7 for LTCs can be obtained by:

$$\frac{\partial F}{\partial \mathbf{I}}\Big|_{x=0} = W(1 - f^2) \odot A$$

The corresponding internal intervention coefficients B of Eq. 7, for LTC networks becomes:

$$\frac{\partial^2 F}{\partial \mathbf{x}(t) \partial \mathbf{I}(t)} = W(f^2 - 1) \odot \big[2W_r f \odot (A - x) + 1\big]$$

This shows that by manipulating matrices W, W_r, and A one can control internal and external interventions to an LTC system which gives the statement of the proposition. □

Proposition 1 shows that LTCs are causal models in their forward pass, and by manipulation of their parameters, one can gain insights into the underlying systems. We next show that LTCs' backward path also gives rise to a causal model. A core finding of our paper is that LTCs trained from demonstration via reverse-mode automatic differentiation [62] can give rise to a causal model.

4.2 Training LTCs via Gradient Descent Yields Causal Models

The uniqueness of the solution of LTCs allows any ODE solver to reproduce a forward pass trajectory with backward computations from the end point of the forward pass. This property enables us to use the backpropagation algorithm, either by using the adjoint sensitivity [54] method or backpropagation through time (BPTT) to train an LTC system. Formally, for a forward pass trajectory of the system states $\mathbf{x}(t)$ of an LTC network from t_0 to t_n, we can compute a scalar value loss, L at t_n by:

$$L(\mathbf{x}(t_n)) = L\Big(\mathbf{x}(t_0) + \int_{t_0}^{t_n} \frac{d\mathbf{x}}{dt} dt\Big). \tag{8}$$

Suppose we use the adjoint method for training the network, then the augmented state is computed by $a(t) = \frac{\partial L}{\partial \mathbf{x}(t)}$, whose kinetics are determined by $\frac{da}{dt} = -a^T(t) \frac{\partial LTC_{rhs}}{\partial \mathbf{x(t)}}$ [8]. To compute the loss gradients in respect to the parameters, we run the adjoint dynamics backwards from gradients $\frac{dL}{d\mathbf{x}(t_n)}$ and states $\mathbf{x}(t_n)$, while solving a third integral in reverse as follows: $\frac{dL}{d\theta} = -\int_{t_n}^{t_0} a^T(t) \frac{\partial LTC_{rhs}}{\partial \theta} dt$. All integrals can be solved in reverse-mode by a call to the ODE solver.

Based on Proposition 1, for an LTC network at each training iteration, the internal and external interventions not only help the system learn statistical dependencies from data but also facilitate the learning of the causal mapping. These results bridge the gap between pure physical models and causal structural models to obtain better learning systems.

Real-world implications. Consider a simple drone navigation task in which the objective is for the drone to navigate from position A to a target position B while simultaneously avoiding obstacles. In this setting, the agent is asked to infer high-level control signals directly from visual inputs to reach the target goal. Even the simplest vision-based deep model can accomplish this task in passive open-loop settings [35]. However, completion in a closed-loop environment while inferring the true causal structure of the task is significantly more challenging. For instance, where should the agent attend to for taking the next action? With what mechanisms does the system infer the objective of the task? And accordingly, how robust are the decisions of the learned policy?

In the next section, we perform an extensive experimental evaluation to find answers to the above questions. We also aim to validate our theoretical results on the capability of LTC models to capture the true causal structure of a high-dimensional visual reasoning task, where other models fail.

5 Experiments

We designed photorealistic visual navigation tasks with varying memory horizons including (1) navigating to a static target, (2) chasing a moving target, and (3) hiking with guide markers (Fig. 3).

Experimental setup. We designed tasks in Microsoft's AirSim [38] and Unreal Engine. To create data for imitation learning, we use greedy path search with a Euclidean heuristic over unoccupied voxels to obtain knot points for cubic spline interpolation, which is then followed via a pure pursuit controller. This strategy is modified for each specific task (details in the supplements).

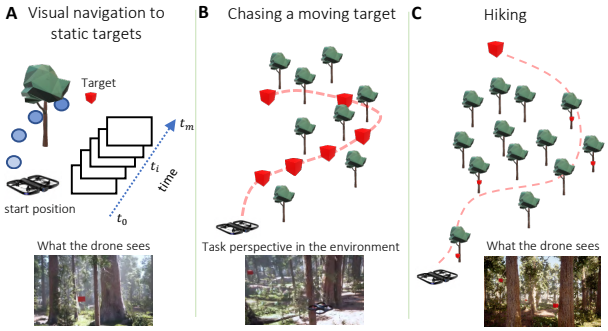

Figure 3: Visual drone navigation tasks. A) Navigation to a static target, B) Chasing a moving target, C) Hiking with a set of markers in the environment

Baselines. We evaluate NCP networks [35] against a set of baseline models. This includes ODE-RNNs [59] which are the recurrent network version of Neural ODEs [8], long short-term memory networks (LSTMs) [26], and CT-GRU networks [43], which are the continuous equivalent of GRUs [9]. These baselines are chosen to validate our theoretical results: Not all CT models are causal models. Similarly, discretized RNN models, such as LSTMs, perform well on tasks with long-term dependencies, yet they are not causal models.

In Section 4, we showed that NCPs, which are sparse neural networks built based on LTC neurons [24], are dynamical causal models. Therefore, they can learn the true causation of a given task. In our experiments, camera images are perceived by convolutional layers and are fed into the RNN networks which act as a controller. For a fair comparison, the number of trainable parameters of all models is within the same range, and they are all trained by Adam optimizer [33] with a cosine similarity loss (See more details in the supplements).

I) Navigation to Static Target with Occlusion. In this task, the drone navigates to a target marker that is less than 25 meters away and visible to it. We place a red cube on a random unoccupied voxel in the environment to function as the target marker. We constrain the target marker to appear in the drone's viewing frustum for most cases. Though there may be occlusion of the target upon random respond of the drone in the environment (more details in the supplements). Occlusions create temporal dependencies. We tested static CNN networks and observed that they fail to perform scenarios in which the target is occluded. Table 1 shows that in both neighborhood and forest environments, all agents learn the task with a reasonable validation loss in a passive imitation learning setting. However, once these agents are deployed with closed-loop control, we observed that the success rates for LSTM and ODE-RNNs drop to only 24% and 18% of 50 attempted runs. CT-GRU managed to complete this task in 40% of the runs, whereas NCPs completed the tasks in 48% of the runs. Why is this the case?

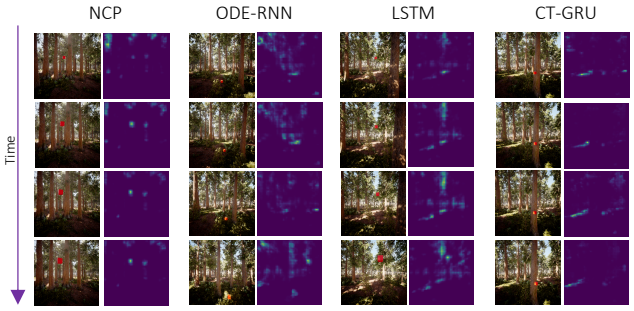

Figure 4: Navigation to a static target in closed-loop environments. NCPs are the only models that can capture the causal structure of the tasks directly from visual data.

To understand these results better, we used the Visual-Backprop algorithm [3] to compute the saliency maps of the learned features in the input space. Saliency maps would show us where the attention of the network was when taking the next navigation decision. As shown in Fig. 4, we observe that NCP has learned to attend to the static target within its field of view to make a future decision. This attention profile was not present in the saliency maps of the other agents. LSTM agents are sensitive to lighting conditions compared to the CT models. This experiment supports our theoretical results on the ability of LTC-based models to learn causal representations.

Table 1: Validation on short-term navigation. Cosine similarity loss [-1, 1] (smaller is better), n=5.

Algorithms	**Environments**	
	RedWood Forest	Neighborhood
LSTM	-0.823 $\pm$ 0.006	-0.838 $\pm$ 0.019
ODE-RNN	-0.815 $\pm$ 0.043	-0.855 $\pm$ 0.019
CT-GRU	-0.855 $\pm$ 0.001	**-0.877** $\pm$ **0.002**
NCP (ours)	**-0.859** $\pm$ **0.035**	-0.855 $\pm$ 0.008

Table 4: Closed-loop evaluation of trained policies on various navigation and interaction tasks. Agents and policies are reinitialized randomly at the beginning of each trial (n=50). Values correspond to success rates (higher is better).

	Static Target					Chasing				Hiking
Model	**Clear**	**Fog**	**Light Rain**	**Heavy Rain**	**Occlusion**	**Clear**	**Fog**	**Light Rain**	**Heavy Rain**	**Clear**
LSTM	24%	22%	22%	4%	20%	66%	**62%**	56%	44%	2%
ODE-RNN	18%	10%	18%	2%	24%	52%	42%	62%	44%	4%
CT-GRU	40%	8%	**60%**	32%	28%	38%	36%	48%	42%	0%
NCP (ours)	**48%**	**40%**	52%	**60%**	**32%**	**78%**	52%	**84%**	**54%**	**30%**

II) Chasing a Moving Target. In this task, the drone follows a target marker along a smooth spline path. Using a generate and test method, we create a path for the drone to follow by using a random walk with momentum to find knot points for fitting a spline (details in the supplements).

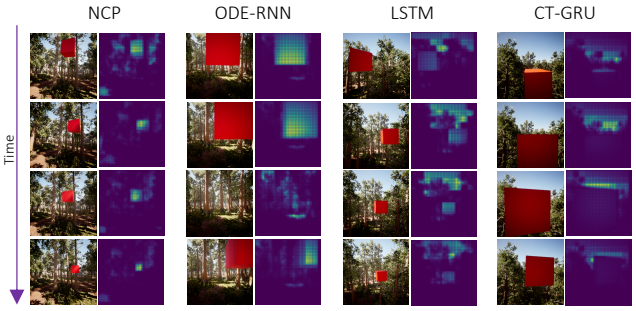

Figure 5: Chasing a moving target in closed-loop environments. NCPs are the only models that can capture the causal structure of the tasks directly from visual data.

Table 2 shows that all agents were able to learn to follow their targets in a passive open-loop case. However, similar to the previous experiment, we witnessed that not all models can successfully complete the task in a closed-loop setting where interventions play a big role. NCPs were 78% successful at completing their task, while LSTM in 66%, ODE-RNNs in 52%, and CT-GRU in 38% were successful. Once again, we looked into the attention maps of the models, illustrated in Fig. 5. We see that CT-GRU networks did not learn to attend to the target they follow. LSTMs again show sensitivity to lighting conditions. ODE-RNNs keep a close distance to the target, but they occasionally lose the target. In contrast, NCPs have learned to attend to the target and follow them as they move in the environment.

Table 2: Chasing objects, Validation performance with co-sine similarity loss (smaller is better), n=5

Algorithms	**Environments**	
	RedWood Forest	Neighborhood
LSTM	-0.943 $\pm$ 0.028	-0.947 $\pm$ 0.008
ODE-RNN	**-0.967** $\pm$ **0.009**	-0.953 $\pm$ 0.011
CT-GRU	-0.958 $\pm$ 0.017	**-0.979** $\pm$ **0.003**
NCP (ours)	-0.936 $\pm$ 0.022	**-0.975** $\pm$ **0.012**

Table 3: Hiking, Validation performance with co-sine similarity loss (smaller is better), n=5

Algorithms	**Environments**	
	RedWood Forest	Neighborhood
LSTM	-0.273 $\pm$ 0.388	**-0.781** $\pm$ **0.030**
ODE-RNN	**-0.896** $\pm$ **0.026**	-0.710 $\pm$ 0.003
CT-GRU	-0.359 $\pm$ 0.073	-0.725 $\pm$ 0.086
NCP (ours)	-0.676 $\pm$ 0.192	-0.711 $\pm$ 0.013

III) Hiking Through an Environment. In this task, the drone follows multiple target markers which are placed on the surface of obstacles within the environment (Fig. 3C) (Experimental details in Supplements). This task is significantly more complex than the previous tasks, especially when agents are deployed directly in the environment. This is because the agents have to learn to follow a much longer time-horizon task, by visual cues, in a hierarchical fashion.

Interestingly, we see most agents learn a reasonable degree of validation loss during the learning process as depicted by Table 3. Even ODE-RNNs realize excellent performance in the passive setting. However, when deployed in the environment, none of the models other than NCP could perform the task completely in 50 runs. NCPs could perform 30% successfully thanks to their causal structure.

6 Discussions, Scope and Conclusions

We provided theoretical evidence for the capability of different representations of continuous-time neural networks in learning causal structures. We then performed a set of experiments to confirm the effectiveness of continuous-time causal models in high-dimensional and visual drone navigation tasks

with different memory-horizons compared to other methods. We conclude that the class of liquid time-constant networks has the great potential of learning causal structures in closed-loop reasoning tasks where other advanced RNN models cannot perform well.

Model performances drop significantly in closed-loop. Table 4 shows a summary of success episodes when the agents were deployed in various navigation tasks. As the memory-horizon and the complexity of the task increases, models that are not causal struggle with close-loop interactions with the environment. Continuous-time models with causal structures, such as NCPs, considerably improve the real-world performance of agents.

Robustness to environmental perturbations.

We next investigated how robust are these models under heavy environmental perturbations such as rain. We used the AirSim weather API to include heavy rain and fog in closed-loop tests of the static target and chasing a moving target task (Fig. 6).

We observed that under high-intensity rain, all models have a performance drop, but NCPs show more resiliency to these perturbations. High-intensity rain and fog (Table 4) confuses LSTM and ODE RNNs the most where they are not able to complete the task.

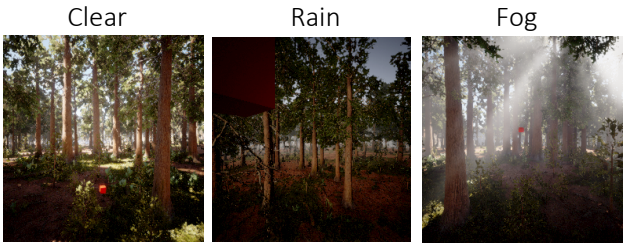

Figure 6: Sample input images from the forest environment. While agents are trained on offline, passive datasets from the Clear environment, testing is performed in an active control setup under various levels of environmental perturbations.

When shall we use temporal convolutions and attention-based methods? Both temporal convolution [1] and attention-based architectures [71] are reasonable choices for many spatiotemporal tasks [37, 31]. However, they require the user to explicitly set the temporal horizon of the observation memory, whereas RNNs learn the temporal memory component, implicitly. Consequently, RNNs are usually the preferred choice when the temporal horizon is not known or varies, which is the case in our experiments.

Handling Occlusions and temporal dependencies. All tasks discussed require temporal dependencies because of environmental line-of-sight occlusions between the drone and the target. We specifically observed that NCPs are capable of handling more complex runs (where occlusions are maximal) and other models fail. We have also tested these scenarios with (single frame) CNNs and observed that the model could not complete the task successfully in all cases where the marker is occluded. CNNs achieved only a 14% success rate in static target runs with an occlusion which is at least 50% lower than the results of the other recurrent models shown in Table 4.

If temporal samples arrive at a constant rate, why a continuous-time model performs better? The use of continuous models (implemented by ODEs) for continuous problems can potentially lead to the design of better causal mechanisms (getting closer to physical modeling in the taxonomy of causal models). Moreover, many recent works show advanced CT models can outperform advanced discretized RNNs even when the incoming samples are equidistant [34, 63, 35, 24, 15]. Nonetheless, CT-models realize a novel category of functions which was not possible to realize otherwise (i.e., the vector fields realized by complex ODE solvers cannot be achieved by discretized models) [8].

It is not trivial to show causal properties for gated RNNs. In gated recurrent networks (e.g., LSTMs), the gating mechanisms can be counted as an implicit intervention to the state representation. However, a direct theoretical framework to prove this is not feasible as there is no direct relation between dynamic causal models and LSTMs, for instance. Besides, our empirical study in visual navigation tasks suggests that the gating mechanism does not help causality (See for instance the attention maps presented in Figures. 1, 4, and 5.)

In summary, we showed that special presentation of CT models realize causal models and can significantly enhance decision making in real-world applications.

References

[1] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[2] Nir Baram, Oron Anschel, Itai Caspi, and Shie Mannor. End-to-end differentiable adversarial imitation learning. In *International Conference on Machine Learning*, pages 390–399. PMLR, 2017.

[3] Mariusz Bojarski, Anna Choromanska, Krzysztof Choromanski, Bernhard Firner, Larry Jackel, Urs Muller, and Karol Zieba. Visualbackprop: efficient visualization of cnns. *arXiv preprint arXiv:1611.05418*, 2016.

[4] Michael Breakspear. Dynamic models of large-scale brain activity. *Nature neuroscience*, 20(3):340–352, 2017.

[5] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *International Conference on Learning Representations*, 2019.

[6] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological slam for visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12875–12884, 2020.

[7] Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning exploration policies for navigation. In *International Conference on Learning Representations*, 2018.

[8] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems*, pages 6571–6583, 2018.

[9] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[10] Pim de Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. *arXiv preprint arXiv:1905.11979*, 2019.

[11] Siddharth Desai, Ishan Durugkar, Haresh Karnan, Garrett Warnell, Josiah Hanna, Peter Stone, and AI Sony. An imitation from observation approach to transfer learning with dynamics mismatch. *Advances in Neural Information Processing Systems*, 33, 2020.

[12] Yan Duan, Marcin Andrychowicz, Bradly C Stadie, Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. *arXiv preprint arXiv:1703.07326*, 2017.

[13] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. In *Advances in Neural Information Processing Systems*, pages 3134–3144, 2019.

[14] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. In *Advances in Neural Information Processing Systems*, pages 7509–7520, 2019.

[15] N. Benjamin Erichson, Omri Azencot, Alejandro Queiruga, Liam Hodgkinson, and Michael W. Mahoney. Lipschitz recurrent neural networks. In *International Conference on Learning Representations*, 2021.

[16] Karl J Friston, Lee Harrison, and Will Penny. Dynamic causal modelling. *Neuroimage*, 19(4):1273–1302, 2003.

[17] Ken-ichi Funahashi and Yuichi Nakamura. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural networks*, 6(6):801–806, 1993.

[18] Amir Gholami, Kurt Keutzer, and George Biros. Anode: Unconditionally accurate memory-efficient gradients for neural odes. *arXiv preprint arXiv:1902.10298*, 2019.

[19] Karthik Gopalratnam, Henry Kautz, and Daniel S Weld. Extending continuous time bayesian networks. In *AAAI*, pages 981–986, 2005.

[20] Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

[21] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, 2017.

[22] Saurabh Gupta, David Fouhey, Sergey Levine, and Jitendra Malik. Unifying map and landmark based representations for visual navigation. *arXiv preprint arXiv:1712.08125*, 2017.

[23] YAN Hanshu, DU Jiawei, TAN Vincent, and FENG Jiashi. On robustness of neural ordinary differential equations. In *International Conference on Learning Representations*, 2020.

[24] Ramin Hasani, Mathias Lechner, Alexander Amini, Daniela Rus, and Radu Grosu. Liquid time-constant networks. *arXiv preprint arXiv:2006.04439*, 2020.

[25] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *arXiv preprint arXiv:1606.03476*, 2016.

[26] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[27] Philipp Holl, Vladlen Koltun, and Nils Thuerey. Learning to control pdes with differentiable physics. *arXiv preprint arXiv:2001.07457*, 2020.

[28] Junteng Jia and Austin R Benson. Neural jump stochastic differential equations. In *Advances in Neural Information Processing Systems*, pages 9843–9854, 2019.

[29] Harang Ju and Danielle S Bassett. Dynamic representations in networked neural systems. *Nature Neuroscience*, 23(8):908–917, 2020.

[30] Amir-Hossein Karimi, Bernhard Schölkopf, and Isabel Valera. Algorithmic recourse: from counterfactual explanations to interventions. *arXiv preprint arXiv:2002.06278*, 2020.

[31] Elia Kaufmann, Antonio Loquercio, René Ranftl, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Deep drone acrobatics. *arXiv preprint arXiv:2006.05768*, 2020.

[32] Charles Kemp, Noah D Goodman, and Joshua B Tenenbaum. Learning to learn causal models. *Cognitive Science*, 34(7):1185–1243, 2010.

[33] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[34] Mathias Lechner and Ramin Hasani. Learning long-term dependencies in irregularly-sampled time series. *arXiv preprint arXiv:2006.04418*, 2020.

[35] Mathias Lechner, Ramin Hasani, Alexander Amini, Thomas A Henzinger, Daniela Rus, and Radu Grosu. Neural circuit policies enabling auditable autonomy. *Nature Machine Intelligence*, 2(10):642–652, 2020.

[36] Mathias Lechner, Ramin Hasani, Manuel Zimmer, Thomas A Henzinger, and Radu Grosu. Designing worm-inspired neural networks for interpretable robotic control. In *International Conference on Robotics and Automation (ICRA)*, pages 87–94, 2019.

[37] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47), 2020.

[38] Ratnesh Madaan, Nicholas Gyde, Sai Vemprala, Matthew Brown, Keiko Nagami, Tim Taubner, Eric Cristofalo, Davide Scaramuzza, Mac Schwager, and Ashish Kapoor. Airsim drone racing lab. In *NeurIPS 2019 Competition and Demonstration Track*, pages 177–191. PMLR, 2020.

[39] Stefano Massaroli, Michael Poli, Jinkyoo Park, Atsushi Yamashita, and Hajime Asma. Dissecting neural odes. In *34th Conference on Neural Information Processing Systems, NeurIPS 2020*. The Neural Information Processing Systems, 2020.

[40] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[41] Joris M Mooij, Dominik Janzing, and Bernhard Schölkopf. From ordinary differential equations to structural causal models: the deterministic case. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pages 440–448, 2013.

[42] Arsalan Mousavian, Alexander Toshev, Marek Fišer, Jana Košecká, Ayzaan Wahid, and James Davidson. Visual representations for semantic target driven navigation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8846–8852. IEEE, 2019.

[43] Michael C Mozer, Denis Kazakov, and Robert V Lindsey. Discrete event, continuous time rnns. *arXiv preprint arXiv:1710.04110*, 2017.

[44] Olavi Nevanlinna. Remarks on picard-lindelöf iteration. *BIT Numerical Mathematics*, 29(3):535–562, 1989.

[45] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[46] Uri Nodelman, Christian R Shelton, and Daphne Koller. In *Proceedings of the Eighteenth International Conference on Uncertainty in Artificial Intelligence*, 2002.

[47] Uri Nodelman, Christian R Shelton, and Daphne Koller. Learning continuous time bayesian networks. In *Proceedings of the Nineteenth International Conference on Uncertainty in Artificial Intelligence*, 2003.

[48] Uri D Nodelman. *Continuous time Bayesian networks*. PhD thesis, Stanford University, 2007.

[49] Judea Pearl. *Causality*. Cambridge university press, 2009.

[50] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 2014.

[51] Judea Pearl et al. Causal inference in statistics: An overview. *Statistics surveys*, 3:96–146, 2009.

[52] Will Penny, Zoubin Ghahramani, and Karl Friston. Bilinear dynamical systems. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 360(1457):983–993, 2005.

[53] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

[54] Lev Semenovich Pontryagin. *Mathematical theory of optimal processes*. Routledge, 2018.

[55] Alessio Quaglino, Marco Gallieri, Jonathan Masci, and Jan KoutnÃk. Snode: Spectral discretization of neural odes for system identification. In *International Conference on Learning Representations*, 2020.

[56] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. In *international conference on machine learning*, pages 2847–2854. PMLR, 2017.

[57] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[58] Yulia Rubanova, Ricky T. Q. Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 5320–5330. Curran Associates, Inc., 2019.

[59] Yulia Rubanova, Ricky TQ Chen, and David Duvenaud. Latent odes for irregularly-sampled time series. *arXiv preprint arXiv:1907.03907*, 2019.

[60] Paul K Rubenstein, Stephan Bongers, Bernhard Schölkopf, and Joris M Mooij. From deterministic odes to dynamic structural causal models. *arXiv preprint arXiv:1608.08028*, 2016.

[61] F Ruggeri, F Faltin, and R Kenett. Bayesian networks. encyclopedia of statistics in quality and reliability, 2007.

[62] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[63] T. Konstantin Rusch and Siddhartha Mishra. Coupled oscillatory recurrent neural network (co{rnn}): An accurate and (gradient) stable architecture for learning long time dependencies. In *International Conference on Learning Representations*, 2021.

[64] Stuart Russell and Peter Norvig. *Artificial intelligence: a modern approach*. 2002.

[65] Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. In *International Conference on Learning Representations*, 2018.

[66] Stefan Schaal. Is imitation learning the route to humanoid robots? *Trends in cognitive sciences*, 3(6):233–242, 1999.

[67] Bernhard Schölkopf. Causality for machine learning. *arXiv preprint arXiv:1911.10500*, 2019.

[68] Yannick Schroecker and Charles Isbell. State aware imitation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 2915–2924, 2017.

[69] Peter Spirtes, Clark N Glymour, Richard Scheines, and David Heckerman. *Causation, prediction, and search*. MIT press, 2000.

[70] Mingfei Sun and Xiaojuan Ma. Adversarial imitation learning from incomplete demonstrations. *arXiv preprint arXiv:1905.12310*, 2019.

[71] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[72] Sebastian Weichwald, Martin E Jakobsen, Phillip B Mogensen, Lasse Petersen, Nikolaj Thams, and Gherardo Varando. Causal structure learning from time series: Large regression coefficients may predict causal links better in practice than small p-values. In *NeurIPS 2019 Competition and Demonstration Track*, pages 27–36. PMLR, 2020.

[73] Yueh-Hua Wu, Nontawat Charoenphakdee, Han Bao, Voot Tangkaratt, and Masashi Sugiyama. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, pages 6818–6827. PMLR, 2019.

[74] Tianhe Yu, Pieter Abbeel, Sergey Levine, and Chelsea Finn. One-shot hierarchical imitation learning of compound visuomotor tasks. *arXiv preprint arXiv:1810.11043*, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...

(a) Did you state the full set of assumptions of all theoretical results? [Yes]
(b) Did you include complete proofs of all theoretical results? [Yes]

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]