

# Inducing Causal Structure for Interpretable Neural Networks

Atticus Geiger\* Zhengxuan Wu\* Hanson Lu\* Josh Rozner Elisa Kreiss Thomas Icard Noah D. Goodman  
Christopher Potts

## Abstract

In many areas, we have well-founded insights about causal structure that would be useful to bring into our trained models while still allowing them to learn in a data-driven fashion. To achieve this, we present the new method of *interchange intervention training* (IIT). In IIT, we (1) align variables in the causal model with representations in the neural model and (2) train a neural model to match the counterfactual behavior of the causal model on a base input when aligned representations in both models are set to be the value they would be for a second source input. IIT is fully differentiable, flexibly combines with other objectives, and guarantees that the target causal model is a *causal abstraction* of the neural model when its loss is minimized. We evaluate IIT on a structured vision task (MNIST-PVR) and a navigational instruction task (ReaSCAN). We compare IIT against multi-task training objectives and data augmentation. In all our experiments, IIT achieves the best results and produces neural models that are more interpretable in the sense that they realize the target causal model.

## 1. Introduction

In many domains, we have well-founded insights about causal structure that we can express in symbolic terms, ranging from commonsense intuitions about how the world works to advanced scientific knowledge. These insights have the potential to make up for gaps in available data, or more generally to provide useful inductive biases. Can we bring these insights into our models while still allowing them to learn in a data-driven fashion?

In this paper, we present *interchange intervention training* (IIT), a new method that trains a neural network to realize the abstract structure of a causal model. In IIT, we (1) align

the variables in a causal model  $\mathcal{C}$  with the representations in a neural model  $\mathcal{N}$  and (2) train  $\mathcal{N}$  to have the counterfactual behavior of  $\mathcal{C}$  by performing aligned *interchange interventions* (swapping of internal states created for different inputs) on  $\mathcal{N}$  using  $\mathcal{C}$ 's counterfactual output as the gold label for the counterfactual prediction of  $\mathcal{N}$ . IIT objectives are differentiable and guarantee that, where the loss is minimized, the target causal model is a *causal abstraction* of the neural network in the sense of Beckers & Halpern (2019).

IIT is an extension of the causal abstraction analysis of Geiger et al. (2021), which can be placed under the broader rubric of *structural evaluations* of neural models, which includes probing and many kinds of feature attribution. Our central point of differentiation from this prior work is that we go beyond passive study of static models, by pushing them to learn specific causal structures as part of optimization. This allows for a productive interplay between model analysis and model improvement: we not only assess whether models have systematic, interpretable internal structure but also push them to acquire such structure.

We evaluate IIT in two very different contexts: (1) ResNet trained on a vision task where one part of an image “points” to another (MNIST-PVR) and (2) a CNN-LSTM model trained to produce action sequences in a grid world given a natural language command (ReaSCAN). In both cases, we define high-level causal models that capture aspects of the problem and that are aligned with our neural models, and we define IIT training objectives using those causal models.

For both case studies, we report two kinds of evaluation: traditional behavioral evaluations using systematic generalization tasks that assess whether a model has learned a truly general solution, and structural evaluations that directly assess the interpretability of our models by evaluating whether they realize the target causal model. We compare IIT against multi-task training objectives and data augmentation methods defined to make use of our causal models, finding that IIT leads to models that both perform better on systematic generalization benchmarks and are more interpretable.<sup>1</sup>

\*Equal contribution. Correspondence to: Atticus Geiger <atticusg@stanford.edu>, Zhengxuan Wu <wuzhengx@stanford.edu>.

<sup>1</sup>We release our code at <https://github.com/frankaging/Interchange-Intervention-Training>.

## 2. Related Work

**Probes** Probes are supervised or unsupervised models that can be used to gain an understanding of what is encoded in the internal representations of neural networks (Hupkes et al., 2018; Peters et al., 2018; Tenney et al., 2019; Clark et al., 2019). Probes have yielded important insights about what models learn to encode. However, probes are fundamentally limited in a way that is central to our present goals: there is no guarantee that probed information plays a causal role in the network’s behavior (Ravichander et al., 2020; Elazar et al., 2020; Geiger et al., 2021; 2020).

**Feature Attribution** In contrast to probes, gradient-based feature attribution methods (Zeiler & Fergus, 2014; Springenberg et al., 2014; Shrikumar et al., 2016; Binder et al., 2016) generally do measure causal properties (Chattopadhyay et al., 2019). For example, Geiger et al. (2021) note that the integrated gradients method of Sundararajan et al. (2017) computes the *individual causal effect* of neurons (Imbens & Rubin, 2015). In comparison with our proposal, the main limitation of these methods is that (by definition) they passively study trained networks rather than allowing for active improvements of them (though see Erion et al. 2021 for a path from attribution to improved optimization).

**Intervention-Based Analyses** Intervention-Based analyses seek to provide rich characterizations of model representations (like probing) while also supporting inferences about the causal role that those representations play (like feature attribution). In intervention-based analysis, one actively changes the values of model representations in systematic ways and studies the effects. Such interventions can be applied to input representations in order to measure the effect on the output representation (Feder et al., 2021; Pryzant et al., 2021), or on network internal representations to characterize how these representations mediate the causal relationships between inputs and outputs (Vig et al., 2020; Soulos et al., 2020; Ravfogel et al., 2020; Elazar et al., 2020; Giulianelli et al., 2018; Geiger et al., 2020; 2021). In the context of neural network analysis, this provides a very powerful tool-kit for understanding a model’s causal structure, since an enormous number of diverse and finely controlled intervention experiments can be performed. We build directly on these methods by extending them to the optimization process.

**Multi-Task Training** Multi-task training is the practice of jointly training a model against a set of learning tasks to improve data efficiency and increase model robustness (Ruder, 2017; Zhang & Yang, 2017; Crawshaw, 2020). This can be thought of in terms of supervised probing. In standard supervised probing, one trains the probe using internal representations from the target model while keeping the

target model frozen. In multi-task training, we allow the target model’s parameters to be changed by the probing process. This provides a natural point of comparison with our proposal for IIT, where we use our target symbolic causal model to define multi-task training objectives.

**Data Augmentation** Data augmentation is the practice of enhancing training sets by modifying existing examples to generate new ones (Perez & Wang, 2017; Shorten & Khoshgoftaar, 2019; Kaushik et al., 2019; Liu et al., 2021). For us, data augmentation is another natural comparison point because we can use a target symbolic causal model to generate additional data. Crucially, IIT involves interchanging internal network representations, while our data augmentation method only involves interchanging parts of inputs.

## 3. Interchange Intervention Training

Our goal is to train a neural network to have an internal causal structure that realizes a high-level causal model. To concretize this goal, we draw on two strands of work on causality: (1) formal interventionist theories of causality (Spirtes et al., 2001; Pearl, 2001), in which causal processes are identified with the effect of interventions, and (2) theories of abstraction (Beckers & Halpern, 2019; Beckers et al., 2020; Chalupka et al., 2016; Rubenstein et al., 2017), where relationships between two causal processes are determined by the presence of systematic correspondences between the effects of interventions. The key insight is that having a particular causal structure is a matter of satisfying a number of counterfactual statements about the effect of interventions (Hitchcock, 2001). The present section defines this process formally, and Figure 1 illustrates all the concepts with a self-contained example.

**Structured Causal Models** We introduce a minimal notation for structured causal models here. We define a structural causal model  $\mathcal{M}$  to consist of variables  $\mathcal{V}$ , and, for each variable  $V \in \mathcal{V}$ , a set of values  $\text{Val}(V)$ , a set of parents  $PA_V$ , and a structured equation  $F_V$  that sets the value of  $V$  based on the setting of its parents. We denote the set of variables with no parents as  $\mathbf{V}_{in}$  and those with no children  $\mathbf{V}_{out}$ . A structural causal model  $\mathcal{M} = (\mathcal{V}, PA, \text{Val}, F)$  can represent both symbolic computations and neural networks.

Given a setting of an **input**  $\in \text{Val}(\mathbf{V}_{in})$  and variables  $\mathbf{V} \subseteq \mathcal{V}$ , we define  $\text{GETVALS}(\mathcal{M}, \text{input}, \mathbf{V}) \in \text{Val}(\mathbf{V})$  to be the setting of  $\mathbf{V}$  determined by the setting **input** and model  $\mathcal{M}$ . For example,  $\mathbf{V}$  could correspond to a layer in a neural network, and  $\text{GETVALS}(\mathcal{M}, \text{input}, \mathbf{V})$  then denotes the particular values that  $\mathbf{V}$  takes on when the model  $\mathcal{M}$  processes **input**.

For a set of variables  $\mathbf{V}$  and a setting for those variables  $\mathbf{v} \in \text{Val}(\mathbf{V})$ , we define  $\mathcal{M}_{\mathbf{V} \leftarrow \mathbf{v}}$  to be the causal model identical

to  $\mathcal{M}$ , except that the structured equations for  $\mathbf{V}$  are set to a constant value  $\mathbf{v}$ .<sup>2</sup> This corresponds closely to the *do* operator of Pearl (2001), which characterizes interventions on models in the service of exploring hypothetical states.

**Interchange Interventions** With the above definitions in place, we can straightforwardly characterize the *interchange interventions* of Geiger et al. (2019), in which a model  $\mathcal{M}$  is used to process two different inputs, **source** and **base**, and then a particular internal state obtained by processing **source** is used in place of the corresponding internal state obtained by **base**. For a given set of variables  $\mathbf{V}$ ,

$$\mathcal{M}_{\mathbf{V} \leftarrow \text{GETVALS}(\mathcal{M}, \text{source}, \mathbf{V})}$$

is a version of  $\mathcal{M}$  with the values of  $\mathbf{V}$  set to those obtained by processing **source**. In addition,

$$\text{GETVALS}(\mathcal{M}, \text{base}, \mathbf{V}_{\text{Out}})$$

is the setting of the outputs  $\mathbf{V}_{\text{Out}}$  obtained by processing **base** with model  $\mathcal{M}$ . When we put these two steps together, we obtain the interchange intervention:

$$\text{INTINV}(\mathcal{M}, \text{base}, \text{source}, \mathbf{V}) \stackrel{\text{def}}{=} \text{GETVALS}(\mathcal{M}_{\mathbf{V} \leftarrow \text{GETVALS}(\mathcal{M}, \text{source}, \mathbf{V})}, \text{base}, \mathbf{V}_{\text{Out}}) \quad (1)$$

In short, the interchange intervention provides the output of the model  $\mathcal{M}$  for the input **base**, except the variables  $\mathbf{V}$  are set to the values they would have if **source** were the input.

**Causal Abstraction Relationships** Suppose we have a high-level model  $\mathcal{M}_{\mathcal{H}}$  and a low-level model  $\mathcal{M}_{\mathcal{L}}$  with identical input spaces and a predetermined mapping of output values from the low to high level,  $\kappa$  (for example, if the low level model produces a probability distribution over output classes, then  $\kappa$  could be the  $\arg \max$  function, which selects the highest probability class). Further suppose we have an alignment  $\Pi$  mapping intermediate variables in  $\mathcal{V}_{\mathcal{H}}$  to non-overlapping subsets of variables in  $\mathcal{V}_{\mathcal{L}}$ . Consider some intermediate variable  $V_H$  and define  $\mathcal{M}_{\mathcal{H}}^*$  to be  $\mathcal{M}_{\mathcal{H}}$  with every variable marginalized other than  $\mathbf{V}_{\text{In}}$ ,  $\mathbf{V}_{\text{Out}}$ , and  $V_H$ . We can use the definition of interchange interventions to define what it means for  $\mathcal{M}_{\mathcal{L}}$  and  $\mathcal{M}_{\mathcal{H}}$  to be in a causal abstraction relationship, namely, for all  $\mathbf{b}, \mathbf{s} \in \mathbf{V}_{\text{In}}$ :

$$\text{INTINV}(\mathcal{M}_{\mathcal{H}}^*, \mathbf{b}, \mathbf{s}, V_H) = \kappa(\text{INTINV}(\mathcal{M}_{\mathcal{L}}, \mathbf{b}, \mathbf{s}, \Pi(V_H))) \quad (2)$$

This is in fact a *constructive* abstraction relationship in the sense of Beckers & Halpern (2019), in which aligned interventions on the low-level model and high-level model

<sup>2</sup>For neural models trained with IIT, this is not just a simple constant, but rather a computation graph, as discussed below.

have the same effect. This is especially suited for situations in which we seek to relate small symbolic models with large neural models consisting of numerous high-dimensional representations.

**Abstraction and Interpretability** Causal abstraction analysis is not a story about the reasoning a neural network *might* use to achieve its behavior, but instead is an intervention-based method that determines how it *does*, in fact, achieve its behavior. We can interpret the semantic content of neural representations using the high-level variables they are aligned with, and understand how those neural representations are composed using the high-level parenthood relation. Simply put, when a high-level causal model is an abstraction of a neural network, it is a faithful interpretation (Lipton, 2018; Jacovi & Goldberg, 2020) of the network.

**Interchange Intervention Accuracy** To quantify partial success when it comes to causal abstraction relationships, we measure the percentage of aligned interchange interventions that produce the same output, reporting this as the *interchange intervention accuracy* (INTINVACC):

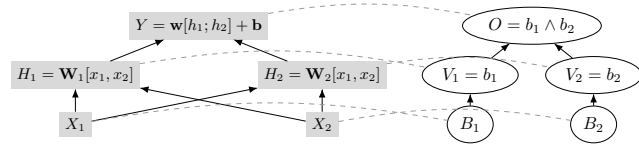
$$\text{INTINVACC}(\mathcal{M}_{\mathcal{H}}, \mathcal{M}_{\mathcal{L}}, V_H, \Pi) \stackrel{\text{def}}{=} \frac{1}{|\text{Val}(\mathbf{V}_{\text{In}})|^2} \sum_{\mathbf{b}, \mathbf{s} \in \text{Val}(\mathbf{V}_{\text{In}})} \mathbb{I} \left[ \text{INTINV}(\mathcal{M}_{\mathcal{H}}^*, \mathbf{b}, \mathbf{s}, V_H) = \kappa(\text{INTINV}(\mathcal{M}_{\mathcal{L}}, \mathbf{b}, \mathbf{s}, \Pi(V_H))) \right] \quad (3)$$

Where every pair of inputs  $\mathbf{b}$  and  $\mathbf{s}$  is considered, and INTINVACC is 1, the two models are in the causal abstraction relationship. However, we often only approximate this by evaluating a set of randomly sampled pairs of inputs, due to the enormous space of input pairs.

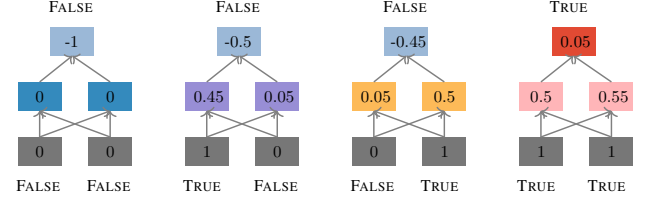
INTINVACC provides a natural metric for quantifying the interpretability of a neural network in the following sense: when INTINVACC is 1, the causal model is an explanation of how the network behaves, providing a very clear window into the network itself. In practice, we rarely observe perfect INTINVACC in complex networks, but we can still say that the higher the value of INTINVACC, the more we have license to reason about the high-level causal model instead of reasoning directly about the low-level network. In this way, the causal model provides an interpretable proxy for the network itself.

**IIT Loss Functions** The definition of IIT for high-level models with one intermediate variable falls out directly from the causal abstraction definition:

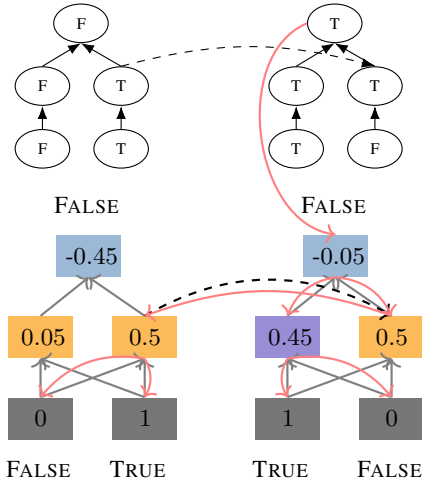
$$\sum_{\mathbf{b}, \mathbf{s} \in \mathbf{V}_{\text{In}}} \text{LOSS} \left( \text{INTINV}(\mathcal{C}, \mathbf{b}, \mathbf{s}, V), \text{INTINV}(\mathcal{N}^{\theta}, \mathbf{b}, \mathbf{s}, \Pi(V)) \right) \quad (4)$$



(a) A linear network with unspecified weights (left) and a symbolic causal model that computes boolean conjunction (right). An alignment between the two is denoted by dashed lines. The causal model is an abstraction of the network when, for both  $V_1$  and  $V_2$ , aligned interchange interventions on network and causal model result in the same output on all 16 ordered pairs of inputs. (The aligned intervention pair (b, s) in general differs from (s, b).)



(b) We define a network with initial parameters  $W_1 = [0.45, 0.05]$ ,  $W_2 = [0.05, 0.5]$ , output bias  $b = -1$ , and output weights  $w = [1, 1]$ . Input values are  $-1$  for False and  $1$  for True. With the initial weights, the network has perfect behavioral accuracy, predicting true (red) iff both its inputs are 1, otherwise it predicts false (blue). Although correct when run on the four inputs (T, T), (T, F), (F, T), (F, F), the *interchange intervention accuracy* is 81.25%: between the two high-level variables  $V_1$  and  $V_2$ , there are six ordered pairs of inputs where performing aligned interchange interventions in the causal model and neural network producing different outputs (see figure 1c for one such pair).

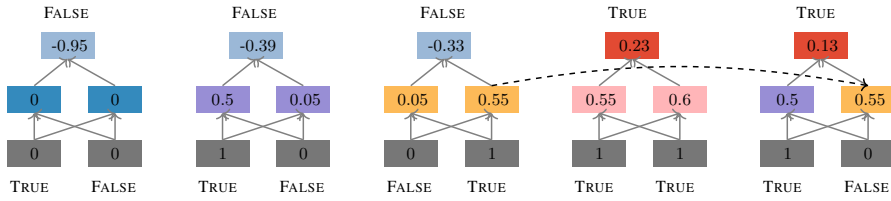


(c) An illustration of an interchange intervention training update, where an intervened network is trained to predict the intervened output of the causal model. It can be seen that the intervention puts the network in a state that could not be achieved with any input representation.

**Require:** High-level and low-level models  $\mathcal{M}^H$  and  $\mathcal{M}^L$  with variables  $\mathcal{V}_H$  and  $\mathcal{V}_L$ , an alignment  $\Pi$  that maps a  $V_H \in \mathcal{V}_H$  to a  $\mathbf{V}_L \subseteq \mathcal{V}_L$ , training dataset  $\mathcal{D}$

- 1:  $\mathcal{M}^H.\text{eval}()$
- 2:  $\mathcal{M}^L.\text{train}()$
- 3: **while** not converged **do**
- 4:   **for** (b, s) in enumerate( $\mathcal{D}$ ) **do** // base and source
- 5:      $V_H \sim \mathcal{V}_H$  // sample a high-level variable
- 6:      $\mathbf{V}_L = \Pi(V_H)$  // aligned low-level variables
- 7:     **with** no\_grad:
- 8:        $a_H = \text{GETVALS}(\mathcal{M}^H, \mathbf{s}, V_H)$
- 9:        $o_H = \text{GETVALS}(\mathcal{M}_{V_H \leftarrow a_H}^H, \mathbf{b}, \mathbf{V}_{\text{Out}})$  // label
- 10:        $a_L = \text{GETVALS}(\mathcal{M}^L, \mathbf{s}, V_L)$
- 11:        $o_L = \text{GETVALS}(\mathcal{M}_{V_L \leftarrow a_L}^L, \mathbf{b}, \mathbf{V}_{\text{Out}})$  // pred
- 12:        $\mathcal{L}_{\text{IIT}} = \text{LOSS}(o_H, o_L)$
- 13:        $\mathcal{L} = \mathcal{L}_{\text{IIT}} + \mathcal{L}_{\text{Others}}$  // combine with other losses
- 14:        $\mathcal{L}.\text{backward}()$
- 15:     Update model parameters with gradients

(d) Pseudocode for interchange intervention training.



(e) The network defined in figure 1b after the IIT training update from figure 1c has been applied, resulting in a network with 100% interchange intervention accuracy (though still nonzero loss), while maintaining the same behavior. The new network has parameters  $W_1 = [0.5012, 0.05]$ ,  $W_2 = [0.5512, 0.05]$ , bias  $b = -0.9488$ , and output weights  $w = [1.0231, 1.0256]$ .

Figure 1. Interchange intervention training example. Network  $\mathcal{N}_\wedge$  performs boolean conjunction with perfect accuracy, or, equivalently, it agrees with  $\mathcal{C}_\wedge$  on the four possible inputs (figure 1b). However,  $\mathcal{C}_\wedge$  is not a causal abstraction of  $\mathcal{N}_\wedge$  under this alignment, because there are aligned interchange interventions that result in  $\mathcal{N}_\wedge$  and  $\mathcal{C}_\wedge$  producing different outputs, meaning that the internal dynamics of the network do not realize the structure of the causal model. To quantify this, we note that the interchange intervention accuracy (Eqn. 3) is 81.25%. After a single interchange intervention training update (figure 1c, figure 1d), this is fixed: all aligned interchange interventions result in the same output (the interchange intervention accuracy is now 1), so  $\mathcal{C}_\wedge$  has become a causal abstraction of  $\mathcal{N}_\wedge$  (figure 1e).



where  $\mathcal{C}$  is the high-level causal model,  $V$  is a high-level variable,  $\mathcal{N}^\theta$  is the low-level neural network with learned parameters  $\theta$ ,  $\Pi(V)$  is a set of low-level variables (neurons) that are aligned with  $V$ , and  $\text{Loss}$  is some loss function  $\mathbf{V}_{\text{Out}} \times \mathbf{V}_{\text{Out}} \rightarrow \mathbb{R}^+$ . Observe that we do not apply the output map  $\kappa$ , because the loss function takes in the network logits directly.

The crucial feature of an IIT update is that the interchange intervention intertwines two computation graphs, one generated by the forward pass for the base input and one by the forward pass for the source input. This means that when backpropagation is performed with the IIT loss objective, updates are applied as they are in regular training, starting from the output representation and proceeding towards the input representations. However, when the intervention site is reached, this process bifurcates, and weights receive two updates, once from  $\mathcal{N}^\theta$  processing the input **base**, and once from  $\mathcal{N}^\theta$  processing **source**. In our toy example (figure 1c), the network is too small to observe this double update, but the networks in our two case studies are not. (See figure 2, which exemplifies such a process.)

An important formal property of IIT is that, if Eqn. 4 is minimized, then  $\mathcal{C}$  and  $\mathcal{N}^\theta$  stand in the causal abstraction relation Eqn. 2. See Appendix A for a brief proof of this result. (The reverse does not hold;  $\mathcal{C}$  can be a causal abstraction of  $\mathcal{N}^\theta$  without the loss being minimized. Figure 1 is an example. This is a desirable property of the method, since we do not expect our loss functions to be minimized in general.)

**Example** Figure 1 provides an example of interchange intervention training, in which a causal model  $\mathcal{C}_\wedge$  of boolean conjunction is aligned with a one-layer linear network  $\mathcal{N}_\wedge^\theta$ , where  $\theta = \{W_1, W_2, b, w\}$ , as in figure 1b.

At the start,  $\mathcal{N}_\wedge^\theta$  is perfect in terms of its input–output behavior but does not conform to the counterfactual behavior of  $\mathcal{C}_\wedge$ . In other words, the regular behavioral learning objective is met, but the interchange intervention training objective is not; interchange intervention accuracy (Eqn. 3) is 81.25.

The figure depicts one interchange intervention training update (figure 1c), which results in a network that satisfies both objectives (figure 1e):  $\mathcal{N}_\wedge^\theta$  now stands in the causal abstraction relation to  $\mathcal{C}_\wedge$  (interchange intervention accuracy is now 1).

## 4. MNIST Pointer-Value Retrieval

Our first benchmark is MNIST Pointer-Value Retrieval (MNIST-PVR; Zhang et al. 2021), a visual reasoning task constructed using the MNIST dataset (LeCun et al., 2010). An input  $i = (i_{\text{TL}}, i_{\text{TR}}, i_{\text{BL}}, i_{\text{BR}})$  consists of four MNIST images (handwritten digits) arranged in a grid. The top left

image  $i_{\text{TL}}$  acts as a pointer that picks out one of the three other images.

**Symbolic Causal Structure** Our target causal model will abstract away from the details of how to identify the handwritten digit in an image, focusing just on the reasoning about pointers. Formally, we define a causal model  $\mathcal{C}_{\text{PVR}} = (\mathcal{V}, PA, \text{Val}, F)$  that computes the label for each of the four MNIST images using an oracle  $O_{\text{MNIST}}$  with a look-up table to select the correct label based on the pointer. The variables are  $\mathcal{V} = \{I_{\text{TL}}, I_{\text{TR}}, I_{\text{BL}}, I_{\text{BR}}, Y_{\text{TL}}, Y_{\text{TR}}, Y_{\text{BL}}, Y_{\text{BR}}, O\}$  and the values assigned by  $\text{Val}$  are the MNIST training images for the four input variables  $I_{\text{TL}}, I_{\text{TR}}, I_{\text{BL}}, I_{\text{BR}}$ , and the set of numbers 0–9 for all other variables. The parents are defined such that  $PA_{I_w} = \emptyset$  and  $PA_{Y_w} = \{I_w\}$  for all  $w \in \{\text{TR}, \text{TL}, \text{BR}, \text{BL}\}$ , and  $PA_O = \{Y_{\text{TL}}, Y_{\text{TR}}, Y_{\text{BL}}, Y_{\text{BR}}\}$ . The structured equations are

$$\begin{aligned} F_{Y_{\text{TL}}}(i_{\text{TL}}) &= O_{\text{MNIST}}(i_{\text{TL}}) \\ F_{Y_{\text{TR}}}(i_{\text{TR}}) &= O_{\text{MNIST}}(i_{\text{TR}}) \\ F_{Y_{\text{BL}}}(i_{\text{BL}}) &= O_{\text{MNIST}}(i_{\text{BL}}) \\ F_{Y_{\text{BR}}}(i_{\text{BR}}) &= O_{\text{MNIST}}(i_{\text{BR}}) \end{aligned} \quad \begin{cases} F_O(y_{\text{TL}}, y_{\text{TR}}, y_{\text{BL}}, y_{\text{BR}}) = \\ \begin{aligned} &y_{\text{TR}} && y_{\text{TL}} \in \{0, 1, 2, 3\} \\ &y_{\text{BL}} && y_{\text{TL}} \in \{4, 5, 6\} \\ &y_{\text{BR}} && y_{\text{TL}} \in \{7, 8, 9\} \end{aligned} \end{cases}$$

**Systematic Generalization** The train/test split designed by Zhang et al. (2021) creates a distributional shift between the training and testing data by removing training examples where either  $O_{\text{MNIST}}(i_{\text{TR}}) \in \{1, 2, 3\}$ ,  $O_{\text{MNIST}}(i_{\text{BL}}) \in \{4, 5, 6\}$ , or  $O_{\text{MNIST}}(i_{\text{BR}}) \in \{0, 7, 8, 9\}$ . This evaluates where models can systematically generalize, learning the general structure of the problem rather than memorizing many special cases.

**Neural Network** We trained ResNet18 using the model from PyTorch vision. This is the deep residual network (He et al., 2016) baseline used by Zhang et al. (2021) on the MNIST-PVR dataset, and we adopt their hyperparameters. We call this model  $\mathcal{N}_{\text{PVR}}^\theta$ , where  $\theta$  abbreviates the parameters.

**Alignments** In our experiments, we align the neural representations of  $\mathcal{N}_{\text{PVR}}^\theta$  with the symbolic variables of  $\mathcal{C}_{\text{PVR}}$  by partitioning the layer resulting from the first application of max-pooling into quadrants  $\mathbf{Q}_{\text{TL}}, \mathbf{Q}_{\text{TR}}, \mathbf{Q}_{\text{BL}}, \mathbf{Q}_{\text{BR}}$  which are aligned with the variables  $Y_{\text{TL}}, Y_{\text{TR}}, Y_{\text{BL}}, Y_{\text{BR}}$ . In initial experimentation, we found that the layers must be partitioned such that each quadrant is directly above its corresponding input. This is likely due to the locality of convolution operators. We also found that aligning layers closer to the classifier head was ineffective.

**Interchange Intervention Training** For each intermediate variable  $Y_w \in \{Y_{\text{TL}}, Y_{\text{TR}}, Y_{\text{BL}}, Y_{\text{BR}}\}$ , we introduce an IIT objective that optimizes for  $\mathcal{N}_{\text{PVR}}^\theta$  implementing the sub-model of  $\mathcal{C}_{\text{PVR}}$  where the three intermediate variables that

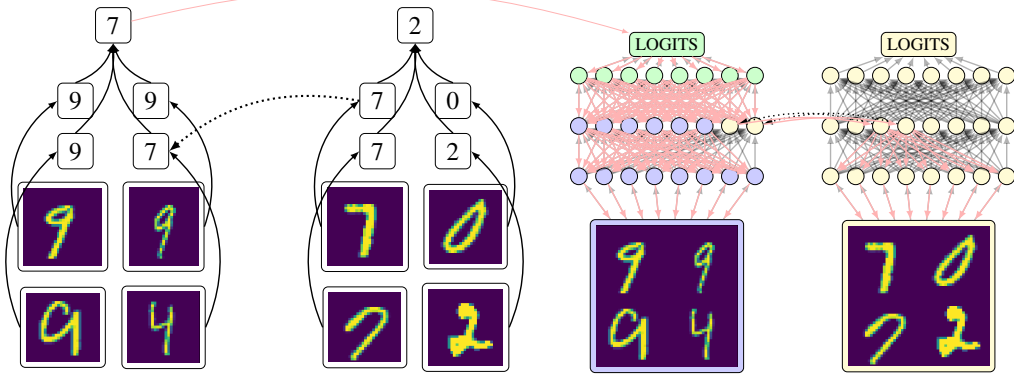


Figure 2. An illustration of an IIT update where a neural network (right) is trained to realize a causal model (left) that solves the PVR-MNIST task. Solid lines are feed-forward connections, dashed lines are interchange interventions, red lines are the flow of backpropagation. Observe that when backpropagation reaches the interchange intervention, it flows into both the source input’s computation graph and the base input’s graph, updating the weights below the interchange intervention twice.

aren’t  $Y_w$  are marginalized out:

$$\sum_{\mathbf{b}, \mathbf{s} \in \text{MNIST-PVR}} \text{CE} \left( \text{INTINV}(\mathcal{C}_{\text{PVR}}, \mathbf{b}, \mathbf{s}, Y_w), \text{INTINV}(\mathcal{N}_{\text{PVR}}^\theta, \mathbf{b}, \mathbf{s}, Q_w) \right) \quad (5)$$

where CE is the cross-entropy loss and MNIST-PVR is the dataset. We visualize an IIT update to  $\mathcal{N}_{\text{PVR}}^\theta$  in figure 2.

**Typed Interchange Intervention Training** We make further use of the causal model by observing that the intermediate variables  $Y_{\text{TL}}, Y_{\text{TR}}, Y_{\text{BL}}, Y_{\text{BR}}$  can be treated as the same type. They all share a value space, as do the neural representations  $Q_{\text{TL}}, Q_{\text{TR}}, Q_{\text{BL}}, Q_{\text{BR}}$ . This means we can perform interchange interventions between different variables and extend our training objective to these interventions as well:

$$\text{T-INTINV}(\mathcal{M}, \mathbf{b}, \mathbf{s}, \mathbf{V}, \mathbf{V}') \stackrel{\text{def}}{=} \text{GETVALS}(\mathcal{M}_{\mathbf{V}' \leftarrow \text{GETVALS}(\mathcal{M}, \mathbf{s}, \mathbf{V})}, \mathbf{b}, \mathbf{V}_{\text{Out}}) \quad (6)$$

$$\sum_{\substack{w, w' \in \{\text{TL}, \text{TR}, \text{BL}, \text{BR}\} \\ \mathbf{b}, \mathbf{s} \in \text{PVR-MNIST}}} \text{CE} \left( \text{T-INTINV}(\mathcal{C}_{\text{PVR}}, \mathbf{b}, \mathbf{s}, Y_w, Y_{w'}), \text{T-INTINV}(\mathcal{N}_{\text{PVR}}^\theta, \mathbf{b}, \mathbf{s}, Q_w, Q_{w'}) \right) \quad (7)$$

**Multi-Task Objectives** To compare against multi-task objectives, we train models to predict the value of intermediate variables from the aligned neural representations, backpropagating into the weights of the target model. Specifically, we train four linear classifiers  $\mathcal{P}^{\varphi_w}$  on the loss

$$\sum_{\substack{\text{input} \in \text{PVR-MNIST} \\ w \in \{\text{TL}, \text{TR}, \text{BL}, \text{BR}\}}} \text{CE}(\mathcal{P}^{\varphi_w}(\text{GETVALS}(\mathcal{N}_{\text{PVR}}^\theta, \text{input}, Q_w)), \text{GETVALS}(\mathcal{C}_{\text{PVR}}, \text{input}, w)) \quad (8)$$

where the trained parameters are  $\theta$  the parameters of ResNet and  $\varphi_w$  the parameters of the linear classifiers.

Training Regime	Behavioral Accuracy		Interchange Intervention Accuracy	
	Train	Test	Train	Test
STANDARD	99.10	0.00	88.80	20.60
IIT	99.60	93.93	99.00	94.85
MULTI	99.64	0.00	89.35	20.50
IIT + MULTI	99.60	<b>96.01</b>	99.10	<b>96.64</b>
AUGMENT	99.40	90.90	98.90	92.00
TYPING ABLATION	99.41	0.09	99.47	16.88

Table 1. Results for  $\mathcal{N}_{\text{PVR}}^\theta$  (ResNet18) trained on the PVR-MNIST dataset. Behavioral accuracy is the percentage of inputs that  $\mathcal{N}_{\text{PVR}}^\theta$  agrees with  $\mathcal{C}_{\text{PVR}}$  on. Interchange intervention accuracy quantifies the extent to which the interpretable causal model is a proxy for the network (section 3). IIT delivers the best results, especially when combined with multi-task objectives.

**Data Augmentation** We perform data augmentation by randomly sampling two examples and swapping a random quadrant of the base input with a random quadrant of the source input to produce a new example that is then labeled with  $\mathcal{C}_{\text{PVR}}$ . This procedure is guided by the same causal structure used by our other models, but it is by definition restricted to input manipulations.

**Results** Our results are shown in Table 1. The standard accuracy captures whether we have achieved our behavioral objective and the interchange intervention accuracy captures whether the symbolic causal model is an abstraction of the neural network.

Neither the standard nor multi-task models learned the behavioral objective in a way that generalizes, with total failure

on the testing data (0%). On the other hand, IIT solves the generalization task (93.93%). However, multi-task training does synergize with IIT, producing the model with the best performance (96.01%). Data augmentation lessens the distributional shift; however, the distributions remain skewed and model performance is stuck at 90.90%.

Our interchange intervention test set accuracies tell a similar story. Neither the standard nor multi-task models learned the IIT objective in a way that generalizes, with total failure on the testing data (20.60% and 20.50%, respectively). On the other hand, IIT learns a general solution to the interchange intervention objectives, achieving accuracy on the test data (94.85%). Again, multi-task training synergizes with IIT, producing the model with the best performance on the IIT objective (96.64%). The causal model  $\mathcal{C}_{PVR}$  is a near perfect abstraction of our best model, meaning the seemingly opaque and complex network dynamics have an interpretable and faithful abstract structure given by  $\mathcal{C}_{PVR}$ .

We can see that Resnet has an inherently modular architecture from the fact that standard training produces a model with quite high (88.80%) interchange intervention accuracy on the training data. However, without any structural training objectives, ResNet does not generalize this modular solution to test data (20.60%). We believe this modularity is the result of convolutions being operations that preserve locality of information across layers. When the distributional shift between training and testing is lessened by data augmentation, the ResNet model produces a model with near perfect (98.90%) interchange intervention accuracy on the training data, which generalizes better to test data (92.00%) (but is still out performed by IIT).

When we ablate our typed interchange intervention objectives, behavioral and interchange intervention accuracy plummets on the test data. Typing our variables is crucial for generalization.

## 5. Grounded Navigation and Language Reasoning (ReaSCAN)

Our second benchmark is ReaSCAN (Wu et al., 2021), a synthetic command-based navigation task that builds off the SCAN (Lake & Baroni, 2018) and gSCAN (Ruis et al., 2020) benchmarks. The goal is to predict an action sequence for the agent to reach the referred target and operate on it given a command and a grid world. For simplicity, we experiment with the simplest command structure included in ReaSCAN, which excludes any relative clauses.<sup>3</sup>

<sup>3</sup>Complex command structures may be accompanied by a different symbolic causal structure, which is not the focus of this paper, and is left for future research.

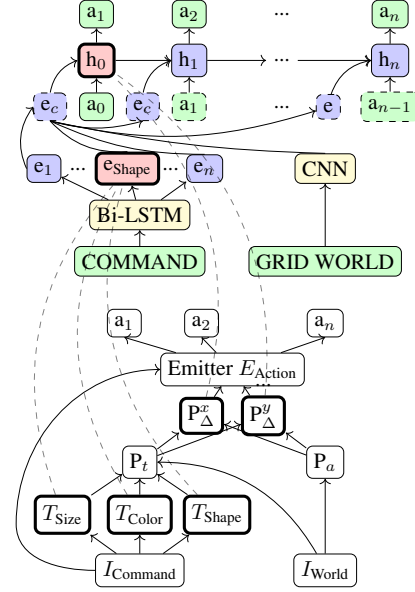


Figure 3. A schematic of the causal model that solves ReaSCAN (bottom) and the neural CNN-LSTM model trained on ReaSCAN (top). Dashed lines align variables in the causal model with neural representations in the CNN-LSTM.

**Symbolic Causal Structure** Our causal model  $\mathcal{C}_{\text{ReaSCAN}} = (\mathcal{V}, PA, \text{Val}, F)$  (see figure 3 bottom) is an oracle solver for ReaSCAN that (1) parses the language command, identifying *size*, *color*, and *shape* properties of the target shape, (2) computes the location of the target object from these properties and the grid world, (3) calculates the horizontal and vertical distances from the agent to the target, and, finally, (4) emits an action sequence that brings the agent to the target. Formally, we define variables and values

$$\begin{aligned} \mathcal{V} &= \{I_{\text{Com}}, I_{\text{World}}, T_{\text{Size}}, T_{\text{Color}}, T_{\text{Shape}}, P_t, P_a, P_{\Delta}^x, P_{\Delta}^y, O\} \\ \text{Val}(T_{\text{Shape}}) &= \{\text{circle, square, cylinder}\} \\ \text{Val}(T_{\text{Color}}) &= \{\text{red, green, blue, yellow}\} \\ \text{Val}(T_{\text{Size}}) &= \{\text{small, big}\} \\ \text{Val}(P_t) &= \text{Val}(P_a) = \text{Val}(P_{\Delta}) = \{-5, \dots, 5\} \end{aligned}$$

with the values  $\text{Val}(I_{\text{Com}})$ ,  $\text{Val}(I_{\text{World}})$ , and  $\text{Val}(O)$  being equal to the command space, world space, and action sequence space. The parents are defined according to the topology of directed arrows pointing from parents to children in figure 3.

The structured equations for object properties,  $F_{T_{\text{Size}}}(i_{\text{Com}})$ ,  $F_{T_{\text{Color}}}(i_{\text{Com}})$ , and  $F_{T_{\text{Shape}}}(i_{\text{Com}})$ , are determined by parsing and interpreting the input language command. The structured equations for position look-ups  $F_{P_t}(t_{\text{Size}}, t_{\text{Color}}, t_{\text{Shape}}, i_{\text{World}})$  and  $F_{P_a}(i_{\text{World}})$  determine the target object and agent location from the target object prop-

Training Regime	Behavioral Exact Match %			
	Novel color	Novel size	Novel direction	Novel length
STANDARD	55.98 (6.31)	41.67 (6.24)	0.00 (0.00)	5.72 (3.44)
MULTI	76.91 (5.02)	39.46 (7.68)	0.00 (0.00)	9.05 (5.28)
IIT	74.12 (6.00)	65.65 (4.26)	0.26 (0.14)	10.20 (6.08)
IIT+ MULTI	<b>80.37</b> (0.88)	<b>74.84</b> (0.04)	<b>14.72</b> (3.54)	<b>25.82</b> (0.37)
Training Regime	Interchange Intervention Accuracy (Exact Match %)			
	Novel color	Novel size	Novel direction	Novel length
STANDARD	44.26 (2.76)	35.57 (2.64)	0.00 (0.00)	0.30 (0.21)
MULTI	68.42 (0.20)	45.83 (2.45)	0.00 (0.00)	0.19 (0.05)
IIT	70.63 (9.33)	65.18 (2.84)	5.24 (3.07)	4.75 (2.06)
IIT+ MULTI	<b>70.73</b> (6.86)	<b>75.34</b> (0.91)	<b>11.79</b> (2.57)	<b>8.49</b> (1.53)

Table 2. Results for the CNN-LSTM on the ReaSCAN systematic generalization tasks. Only models that use IIT are able to consistently get traction on these tasks, and once again we see that IIT combines effectively with multi-task objectives, in both standard behavioral evaluations and evaluations that seek to quantify the extent to which the high-level causal model serves as an interpretable proxy for the network.

erties and the input world. The position deltas  $F_{P_{\Delta}^x}(P_t, P_a)$  and  $F_{P_{\Delta}^y}(P_t, P_a)$  are determined to be the horizontal and vertical distance between the target object and agent, respectively. Finally, the structured equations for the output  $F_O(P_{\Delta}^x, P_{\Delta}^y)$  is the action sequence that takes the agent to the target object, as determined by the vertical and horizontal distances between the two. We visualize this model in figure 3 (bottom).

**Systematic Generalization** ReaSCAN includes testing examples that are systematically different from training examples. Performance on those test sets provides insights into a model’s capabilities to generalize to unseen composites of seen concepts in a zero-shot fashion. In this experiment, we generate four unseen testing splits investigating two distinct generalization patterns by adapting ReaSCAN’s data generation framework.<sup>4</sup> We investigate two splits focusing on novel attribute compositions in input commands (**Novel color** and **Novel size**), and two splits focusing on novel compositions in output action sequences (**Novel direction** and **Novel length**). See Appendix B for details about these splits.

**Neural Network** We use the original baseline models for ReaSCAN (Wu et al., 2021) as our neural model  $\mathcal{N}_{\text{CNN-LSTM}}^{\theta}$ .  $\mathcal{N}_{\text{CNN-LSTM}}^{\theta}$  is a multimodal sequence-to-sequence model which takes in a command and a grid world, and predicts an action sequence as shown in figure 3. We include details about the model and experimental set-up in Appendix B.

**Alignments** In our experiments, we align neural representations of  $\mathcal{N}_{\text{CNN-LSTM}}^{\theta}$  with the variables  $T_{\text{Size}}, T_{\text{Color}}, T_{\text{Shape}}$

and  $P_{\Delta}$ , in  $\mathcal{C}_{\text{ReaSCAN}}$ . We choose the neural representation  $\mathbf{e}_{\text{Shape}}$  output by the LSTM encoder above the noun token (e.g., “circle”), which has 75 dimensions, to be evenly partitioned into three chunks of 25 dimensions, which are aligned with the target properties  $T_{\text{Size}}, T_{\text{Color}}$ , and  $T_{\text{Shape}}$ . The recurrent nature of the LSTM model may encode target properties by the hidden representation of the  $T_{\text{Shape}}$  property, which is at the last word in a target expression. For the position deltas, we choose the initial hidden representation  $\mathbf{h}_1$  of the decoder LSTM, which has 100 dimensions, to be sliced into two evenly partitioned 50 dimension chunks where the first chunk represents the position difference by row  $P_{\Delta}^y$ , and the second chunk represents the position difference by column  $P_{\Delta}^x$ . We hypothesize that  $\mathbf{h}_1$  encodes information about the target position with respect to the agent position derived from the shape world and the command, and our action emitter generated output sequence accordingly.

**Interchange intervention training** For each possible aligned pair of variables and neurons in  $\mathcal{C}_{\text{ReaSCAN}}$  and  $\mathcal{N}_{\text{CNN-LSTM}}^{\theta}$  (i.e.,  $C_W, \mathbf{N}_W$ , where  $C_W \in \{T_{\text{Size}}, T_{\text{Color}}, T_{\text{Shape}}, P_{\Delta}^x, P_{\Delta}^y\}$ , and  $\mathbf{N}_W \in \{\mathbf{e}_{\text{Shape}}, \mathbf{h}_1\}$  where  $i$  is the index of the token for the shape descriptor), we introduce an IIT objective that optimizes for  $\mathcal{N}_{\text{CNN-LSTM}}^{\theta}$  implementing the submodel of  $\mathcal{C}_{\text{ReaSCAN}}$ :

$$\sum_{\mathbf{b}, \mathbf{s} \in \text{ReaSCAN}} \text{CE}_{\text{Action}} \left( \text{INTINV}(\mathcal{N}_{\text{CNN-LSTM}}^{\theta}, \mathbf{b}, \mathbf{s}, \mathbf{N}_W), \text{INTINV}(\mathcal{C}_{\text{ReaSCAN}}, \mathbf{b}, \mathbf{s}, C_W) \right) \quad (9)$$

where  $\text{CE}_{\text{Action}}$  is the cross-entropy loss over each action token prediction over the complete action sequence.

**Multi-task Objectives** Similar to MNIST-PVR, we train small models to predict the position offsets between the

<sup>4</sup>To better facilitate our evaluation pipeline, we regenerate specialized datasets using ReaSCAN’s data generation framework. Details about the datasets can be found in Appendix B.



target and the agent from the aligned neural representations. Specifically, for all  $W \in \{T_{\text{Size}}, T_{\text{Color}}, T_{\text{Shape}}, P_{\Delta}^x, P_{\Delta}^y\}$ , we train a single-layer linear classifier  $\mathcal{P}^{\varphi_W}$  on the loss

$$\sum_{i \in \text{ReaSCAN}} \text{CE}_{\text{Position}} \left( \mathcal{P}^{\varphi_W} \left( \text{GETVALS}(\mathcal{N}_{\text{CNN-LSTM}}^{\theta+i}, \mathbf{i}, \mathbf{P}_{\Delta}) \right), \text{GETVALS}(\mathcal{C}_{\text{ReaSCAN}}, \mathbf{i}, h_1) \right) \quad (10)$$

where the trained parameters are  $\theta$ , the parameters of the CNN-LSTM, and  $\varphi_W$ , the parameters of the linear classifiers.

**Results** Our results are shown in Table 2. We use exact matches of action sequences as our evaluation metric for the behavioral and interchange intervention tasks.

We begin with our results on the behavioral task. Standard training produces models that fail to generalize across all four tasks. IIT alone out-performs multi-task training on novel sizes and lengths, and performs similarly on novel colors and lengths. Again, we observe that IIT and multi-task synergize, producing the models that best generalize across all tasks. Overall, IIT is essential to achieving state-of-the-art results on this systematic generalization task.

Our interchange intervention accuracy results suggest that IIT delivers models that best conform to the interpretable causal model. Without any IIT objectives, both the standard and multi-task models achieve non-zero interchange intervention accuracy for only the two easier splits: novel colors and novel size. IIT achieves significant improvements over these two tasks and gets traction on the two more difficult ones, novel direction and novel length. And, once again, combining IIT with multi-task training delivers the best model by wide margins on all four tasks.

## 6. Conclusion

We introduced interchange intervention training as a method to imbue neural networks with interpretable, systematic causal structure, and we conducted two quite different case studies with IIT: a vision task (MNIST-PVR) and a grounded language understanding task (ReaSCAN). In both settings, models trained with IIT perform best in standard (but very challenging) behavioral evaluations and prove to be the most interpretable in the sense that they conform best to our high-level causal models of the tasks. In addition, our results show that IIT is easily combined with multi-task objectives that further strengthen the results. These initial findings suggest that IIT is a flexible and powerful way to bring high-level insights about causal structure into a data-driven learning process.

## References

- Beckers, S. and Halpern, J. Y. Abstracting causal models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):2678–2685, Jul. 2019. doi: 10.1609/aaai.v33i01.33012678. URL <https://ojs.aaai.org/index.php/AAAI/article/view/4117>.
- Beckers, S., Eberhardt, F., and Halpern, J. Y. Approximate causal abstractions. In Adams, R. P. and Gogate, V. (eds.), *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, volume 115 of *Proceedings of Machine Learning Research*, pp. 606–615, Tel Aviv, Israel, 22–25 Jul 2020. PMLR. URL <http://proceedings.mlr.press/v115/beckers20a.html>.
- Binder, A., Montavon, G., Bach, S., Müller, K., and Samek, W. Layer-wise relevance propagation for neural networks with local renormalization layers. *CoRR*, abs/1604.00825, 2016. URL <http://arxiv.org/abs/1604.00825>.
- Chalupka, K., Eberhardt, F., and Perona, P. Multi-level cause-effect systems. In Gretton, A. and Robert, C. C. (eds.), *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pp. 361–369, Cadiz, Spain, 09–11 May 2016. PMLR. URL <http://proceedings.mlr.press/v51/chalupka16.html>.
- Chattopadhyay, A., Manupriya, P., Sarkar, A., and Balasubramanian, V. N. Neural network attributions: A causal perspective. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 981–990, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL <http://proceedings.mlr.press/v97/chattopadhyay19a.html>.
- Clark, K., Khandelwal, U., Levy, O., and Manning, C. D. What does BERT look at? an analysis of BERT’s attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 276–286, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4828. URL <https://www.aclweb.org/anthology/W19-4828>.
- Crawshaw, M. Multi-task learning with deep neural networks: A survey. *CoRR*, abs/2009.09796, 2020. URL <https://arxiv.org/abs/2009.09796>.
- Elazar, Y., Ravfogel, S., Jacovi, A., and Goldberg, Y. Amnesic probing: Behavioral explanation with amnesic counterfactuals. In *Proceedings of the 2020 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, November 2020. doi: 10.18653/v1/W18-5426.

- Erion, G., Janizek, J. D., Sturmfels, P., Lundberg, S. M., and Lee, S.-I. Improving performance of deep learning models with axiomatic attribution priors and expected gradients. *Nature Machine Intelligence*, 3(7):620–631, 2021. doi: 10.1038/s42256-021-00343-w. URL <https://doi.org/10.1038/s42256-021-00343-w>.
- Feder, A., Oved, N., Shalit, U., and Reichart, R. CausalLM: Causal Model Explanation Through Counterfactual Language Models. *Computational Linguistics*, pp. 1–54, 05 2021. ISSN 0891-2017. doi: 10.1162/coli\_a.00404. URL [https://doi.org/10.1162/coli\\_a.00404](https://doi.org/10.1162/coli_a.00404).
- Fukushima, K. and Miyake, S. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pp. 267–285. Springer, 1982.
- Geiger, A., Cases, I., Karttunen, L., and Potts, C. Posing fair generalization tasks for natural language inference. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4475–4485, Stroudsburg, PA, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1456. URL <https://www.aclweb.org/anthology/D19-1456>.
- Geiger, A., Richardson, K., and Potts, C. Neural natural language inference models partially embed theories of lexical entailment and negation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pp. 163–173, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.blackboxnlp-1.16. URL <https://www.aclweb.org/anthology/2020.blackboxnlp-1.16>.
- Geiger, A., Lu, H., Icard, T., and Potts, C. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*, 2021. URL <https://arxiv.org/abs/2109.08994>.
- Giulianelli, M., Harding, J., Mohnert, F., Hupkes, D., and Zuidema, W. Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 240–248, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5426. URL <https://www.aclweb.org/anthology/W18-5426>.
- He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.
- Hitchcock, C. The intransitivity of causation revealed in equations and graphs. *Journal of Philosophy*, 98(6):273–299, 2001.
- Hudson, D. A. and Manning, C. D. Compositional attention networks for machine reasoning. In *International Conference on Learning Representations*, 2018.
- Hupkes, D., Bouwmeester, S., and Fernández, R. Analysing the potential of seq-to-seq models for incremental interpretation in task-oriented dialogue. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 165–174, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5419. URL <https://www.aclweb.org/anthology/W18-5419>.
- Imbens, G. W. and Rubin, D. B. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press, 2015.
- Jacovi, A. and Goldberg, Y. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4198–4205, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.386. URL <https://www.aclweb.org/anthology/2020.acl-main.386>.
- Kaushik, D., Hovy, E. H., and Lipton, Z. C. Learning the difference that makes a difference with counterfactually-augmented data. *CoRR*, abs/1909.12434, 2019. URL <http://arxiv.org/abs/1909.12434>.
- Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.
- Lake, B. and Baroni, M. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pp. 2873–2882. PMLR, 2018.
- LeCun, Y., Cortes, C., and Burges, C. Mnist handwritten digit database. *ATT Labs [Online]*. Available: <http://yann.lecun.com/exdb/mnist>, 2, 2010.
- Lipton, Z. C. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, jun 2018. ISSN 1542-7730. doi: 10.1145/3236386.3241340. URL <https://doi.org/10.1145/3236386.3241340>.
- Liu, Q., Kusner, M., and Blunsom, P. Counterfactual data augmentation for neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 187–197,

- Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.18. URL <https://aclanthology.org/2021.naacl-main.18>.
- Pearl, J. Direct and indirect effects. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, UAI'01, pp. 411–420, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1558608001.
- Perez, L. and Wang, J. The effectiveness of data augmentation in image classification using deep learning. *CoRR*, abs/1712.04621, 2017. URL <http://arxiv.org/abs/1712.04621>.
- Peters, M., Neumann, M., Zettlemoyer, L., and Yih, W.-t. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1499–1509, Brussels, Belgium, October–November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1179. URL <https://www.aclweb.org/anthology/D18-1179>.
- Pryzant, R., Card, D., Jurafsky, D., Veitch, V., and Sridhar, D. Causal effects of linguistic properties. In *NAACL*, 2021.
- Ravfogel, S., Elazar, Y., Gonen, H., Twiton, M., and Goldberg, Y. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7237–7256, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.647. URL <https://www.aclweb.org/anthology/2020.acl-main.647>.
- Ravichander, A., Belinkov, Y., and Hovy, E. Probing the probing paradigm: Does probing accuracy entail task relevance?, 2020.
- Rubenstein, P. K., Weichwald, S., Bongers, S., Mooij, J. M., Janzing, D., Grosse-Wentrup, M., and Schölkopf, B. Causal consistency of structural equation models. In *Proceedings of the 33rd Conference on Uncertainty in Artificial Intelligence (UAI)*. Association for Uncertainty in Artificial Intelligence (AUAI), August 2017. URL <http://auai.org/uai2017/proceedings/papers/11.pdf>. \*equal contribution.
- Ruder, S. An overview of multi-task learning in deep neural networks. 06 2017.
- Ruis, L., Andreas, J., Baroni, M., Bouchacourt, D., and Lake, B. M. A benchmark for systematic generalization in grounded language understanding. *Advances in Neural Information Processing Systems*, 33, 2020.
- Schuster, M. and Paliwal, K. K. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.
- Shorten, C. and Khoshgoftaar, T. M. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6:1–48, 2019.
- Shrikumar, A., Greenside, P., Shcherbina, A., and Kundaje, A. Not just a black box: Learning important features through propagating activation differences. *CoRR*, abs/1605.01713, 2016. URL <http://arxiv.org/abs/1605.01713>.
- Soulos, P., McCoy, R. T., Linzen, T., and Smolensky, P. Discovering the compositional structure of vector representations with role learning networks. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pp. 238–254, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.blackboxnlp-1.23. URL <https://www.aclweb.org/anthology/2020.blackboxnlp-1.23>.
- Spirtes, P., Glymour, C. N., and Scheines, R. *Causation, Prediction, and Search*. MIT Press, 2nd edition, 2001.
- Springenberg, J., Dosovitskiy, A., Brox, T., and Riedmiller, M. Striving for simplicity: The all convolutional net. *CoRR*, 12 2014.
- Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3319–3328, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL <http://proceedings.mlr.press/v70/sundararajan17a.html>.
- Tenney, I., Das, D., and Pavlick, E. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4593–4601, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1452. URL <https://www.aclweb.org/anthology/P19-1452>.
- Vig, J., Gehrmann, S., Belinkov, Y., Qian, S., Nevo, D., Singer, Y., and Shieber, S. Causal mediation analysis for interpreting neural nlp: The case of gender bias, 2020.
- Wu, Z., Kreiss, E., Ong, D. C., and Potts, C. ReaSCAN: Compositional reasoning in language grounding. *NeurIPS 2021 Datasets and Benchmarks Track*, 2021. URL <https://arxiv.org/abs/2109.08994>.

Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T. (eds.), *Computer Vision – ECCV 2014*, pp. 818–833, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10590-1.

Zhang, C., Raghu, M., Kleinberg, J. M., and Bengio, S. Pointer value retrieval: A new benchmark for understanding the limits of neural network generalization. *CoRR*, abs/2107.12580, 2021. URL <https://arxiv.org/abs/2107.12580>.

Zhang, Y. and Yang, Q. A survey on multi-task learning. *CoRR*, abs/1707.08114, 2017. URL <http://arxiv.org/abs/1707.08114>.

## A. Minimized Loss Entails Causal Abstraction

**Claim** Suppose we have a loss function  $\text{LOSS}$  that outputs a non-negative value. If  $\text{LOSS}(x, y) = 0 \Rightarrow x = \kappa(y)$ , then the interchange intervention loss being zero guarantees that then causal model  $\mathcal{C}$  is a causal abstraction of the neural network  $\mathcal{N}^\theta$ .

**Proof** Suppose that

$$\sum_{\mathbf{b}, \mathbf{s} \in \mathbf{V}_m} \text{LOSS}(\text{INTINV}(\mathcal{C}, \mathbf{b}, \mathbf{s}, V), \text{INTINV}(\mathcal{N}^\theta, \mathbf{b}, \mathbf{s}, \Pi(V))) = 0 \quad (11)$$

Because our loss function outputs non-negative numbers, we know that, if the sum Eqn. 4 is 0, then each addend in the sum is 0:

$$\forall \mathbf{b}, \mathbf{s} \in \mathbf{V}_m : \text{LOSS}(\text{INTINV}(\mathcal{C}, \mathbf{b}, \mathbf{s}, V), \text{INTINV}(\mathcal{N}^\theta, \mathbf{b}, \mathbf{s}, \Pi(V))) = 0 \quad (12)$$

Because our loss function is such that  $\text{LOSS}(x, y) = 0 \Rightarrow x = \kappa(y)$ , we conclude:

$$\forall \mathbf{b}, \mathbf{s} \in \mathbf{V}_m : \text{INTINV}(\mathcal{C}, \mathbf{b}, \mathbf{s}, V) = \kappa(\text{INTINV}(\mathcal{N}^\theta, \mathbf{b}, \mathbf{s}, \Pi(V))) \quad (13)$$

This is exactly the condition for abstraction in Eqn. 2.

## B. ReaSCAN

**Dataset Generation** Table 3 shows dataset statistics. For the novel color and novel size splits, we only train a single model which uses the same training set, but test on different testing sets, as discussed in section 5. For the novel color, novel size and novel length splits, we use the ReaSCAN

Split	#Train	#Dev	#Test	#Zero-shot
A1: novel color	76,102	3,816	3,774	7,195
A2: novel size	76,102	3,816	3,774	7,227
B1: novel direction	34,343	1,201	357	8,282
B2: novel length	52,662	4,250	4,250	1,338

Table 3. Statistics of all splits in our ReaSCAN dataset.

framework<sup>5</sup> to generate Simple commands without any relative clause as discussed in its original paper (Wu et al., 2021). For the novel color and novel size splits, we have allowed verbs = {"walk to", "push", "pull"}, and allowed adverbs = {"while zigzagging", "while spinning", "cautiously", "hesitantly"}. For the novel direction and novel length splits, we have allowed verbs = {"walk to"}, and we disallow adverbs, as we are focusing on action length generalization, not command generalization.

Our split B1 is derived from gSCAN with its novel direction testing split (Ruis et al., 2020), as the ReaSCAN framework cannot partition splits by relative agent-to-target direction. We set 200 grid worlds per command for the novel color and novel size splits, and we set 1200 grid worlds per command for the novel length split, as the allowed command pattern is much smaller for this split, since we exclude all other verbs except "walk to".

The ReaSCAN dataset generation procedure leads to some artifacts, which are discussed in its original paper in detail. These are not especially relevant for our experiments.

The data generation process takes approximately 30 minutes on a multi-CPU cluster. Although we generate our own datasets from an existing data generation engine, our training paradigm can be extended to solve existing datasets.

**Experiment Set-up** For our CNN-LSTM, we adapt code from the original repository.<sup>6</sup> For all training objectives, we optimize for cross-entropy loss using Adam with default parameters (Kingma & Ba, 2015). The learning rate starts at  $1e^{-4}$  and decays by 0.9 every 20,000 steps. We train the model for a fixed number of epochs (100,000) before stopping. The best model is picked by performance on a smaller development set of 2,000 examples, which is consistent with the training pipeline proposed in Ruis et al. (2020) for gSCAN. The training time is about 1 day on a Standard GeForce RTX 2080 Ti GPU with 11GB memory. To foster reproducibility, we release our adapted evaluation scripts in our code repository. We repeat each experiment with three distinct random seeds to ensure a fair comparison.

<sup>5</sup>The implementation is adapted from ReaSCAN’s public code repository: <https://github.com/frankaging/Reason-SCAN>.

<sup>6</sup>[https://github.com/LauraRuis/multimodal\\_seq2seq\\_gSCAN](https://github.com/LauraRuis/multimodal_seq2seq_gSCAN)



**Training Procedure** We release implementations for our neural models with our symbolic causal structures in our code repository. Our released symbolic causal structures for solving ReaSCAN is not unique, and may not be the optimal one for improving generalizability. Additionally, our variable mappings between two models are not unique. Ideally, a chosen casual variable can be mapped into any hidden states in the neural model. However, we find that the specific mapping chosen substantially affects model performance and generalizability.

In contrast to a standard training pipeline, which takes in a single input, our IIT takes pairs of examples as inputs. We found that the formulation of the pairs affects performance. We leave this for future research into the effects of example pairing on model performance.

**Generalization Splits** To evaluate the generalization power of models, ReaSCAN includes testing examples that are systematically different from training examples. Specifically, ReaSCAN generates unseen testing patterns to assess whether models can generalize to unseen composites of seen concepts in a zero-shot setting. We now describe each split in detail.

**Novel Color Attribute (Novel color)** allows models to see “yellow circle” (6,127 examples) and “red square” (6,111 examples) during training but never allows any commands containing “yellow square” during training, and evaluates models with commands containing “yellow square” during test time.

**Novel Size Attribute (Novel size)** holds out all commands referring to small cylinders in any color, meaning that models have not seen commands containing phrases such as “small cylinder” or “small yellow cylinder” during training. On the other hand, models have seen commands containing “big cylinder” (2,020 examples) or “small square” (2,093 examples). At test time, models need to generalize to the hold-out examples.

**Novel Direction (Novel direction)** holds out any command and grid world pair where the referred target is initially located at the south west (SW) of the agent. At test time, models need to generate action sequence to reach to the target which is located SW of the agent. As our agent is always facing to the right (i.e., east) in the beginning, models need to generate action sequences containing three “turn left” actions in order to reach any target positioning at SW of the agent.

**Novel Action Sequence Length (Novel length)** holds out any command and grid world pair that requires models to predict action sequence that contains more than 10 actions. At test time, models need to generalize to examples that require 11, 12, or 13 actions to reach at the target.

**CNN-LSTM** The encoder contains two parts, a convolutional network (CNN) (Fukushima & Miyake, 1982) for encoding the grid world and a bi-directional LSTM (Schuster & Paliwal, 1997; Hudson & Manning, 2018) for encoding the command. The decoder is a LSTM with cross-modality attention weights over the command and the grid world.

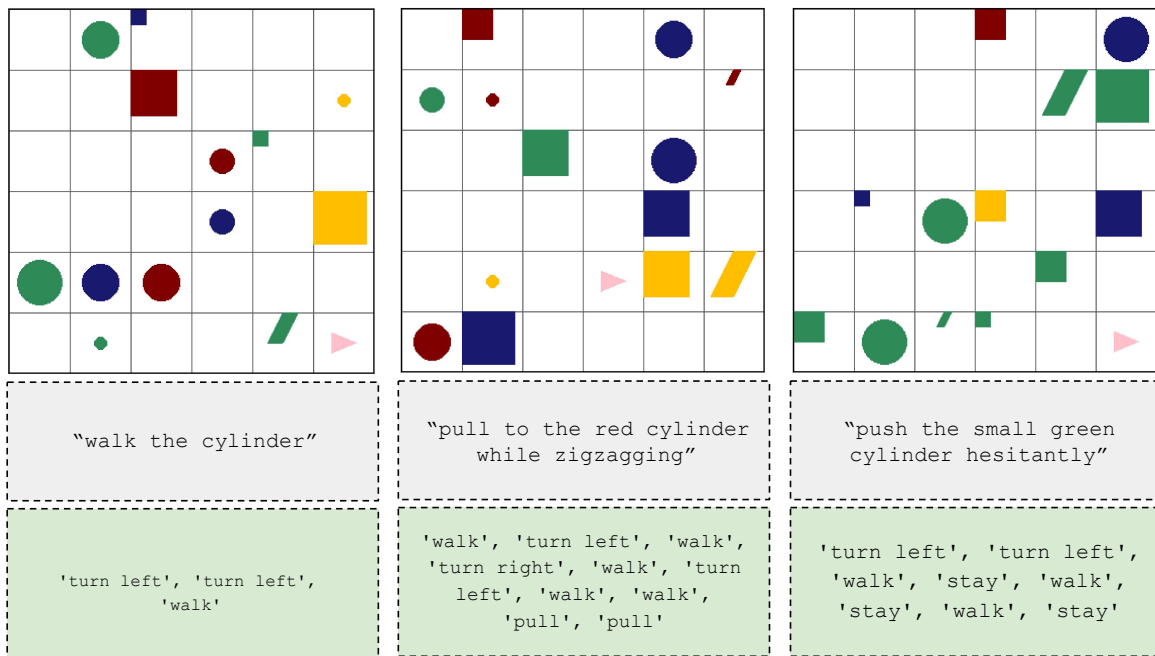


Figure 4. ReaSCAN examples with varying command patterns. The navigation commands and the target action sequences are in the grey boxes and green boxes respectively.