Inducing Causal Structure for Interpretable Neural Networks

Atticus Geiger [*1] **Zhengxuan Wu** [*1] **Hanson Lu** [*1] **Josh Rozner** [1] **Elisa Kreiss** [1]
Thomas Icard [1] **Noah D. Goodman** [1] **Christopher Potts** [1]

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 a causal model (e.g., a deterministic program or Bayesian network) with representations in a neural model and (2) train the 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 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 zero. We evaluate IIT on a structural vision task (MNIST-PVR), a navigational language task (ReaSCAN), and a natural language inference task (MQNLI). 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 more successfully 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, when the loss is zero, 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 three contexts: (1) ResNet trained on a vision task where one part of an image "points" to another (MNIST-PVR), (2) a CNN-LSTM model trained to produce action sequences in a grid world given a natural language command (ReaSCAN), and (3) a pretrained BERT model fine-tuned to label the semantic relation between two sentences (MQNLI). For each context, we define a high-level causal model that capture aspects of the task. We then align high-level causal variables to low-level neural representations to define IIT training objectives.

For the three 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

[*] Equal contribution [1] Stanford University, Stanford, California. Correspondence to: Atticus Geiger <atticusg@stanford.edu>, Zhengxuan Wu <wuzhengx@stanford.edu>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

generalization benchmarks and are more interpretable.[1]

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 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 (Giulianelli et al., 2018; Bau et al., 2019; Vig et al., 2020; Soulos et al., 2020; Ravfogel et al., 2020; Elazar et al., 2020; Besserve et al., 2020; Geiger et al., 2020; Csordás et al., 2021; Geiger et al., 2021; Meng et al., 2022). In the context of neural network analysis, this provides a 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 on these methods, 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 data augmentation methods only involve the creation 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, 2009), in which causal processes are associated 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.

Structural Causal Models We introduce a minimal notation for structural 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 $\mathsf{Val}(V)$, a set of parents PA_V, and a structural 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 $\mathcal{M} = (\mathcal{V}, PA, \mathsf{Val}, F)$ can represent both symbolic computations and neural networks.

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

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

[1] We release our code at `https://github.com/frankaging/Interchange-Intervention-Training`.

are set to constant values $\mathbf{v}$. Because we overwrite neurons with $\mathbf{v}$ in-place, gradients can back-propagate through $\mathbf{v}$. This corresponds closely to the *do* operator of Pearl (2009), which characterizes interventions on models in the service of exploring hypothetical or counterfactual states.

Interchange Interventions With the above definitions in place, we can straightforwardly characterize the *interchange interventions* of Geiger et al. (2020), in which a model $\mathcal{M}$ is used to process two different inputs, $\mathbf{source}$ and $\mathbf{base}$, and then a particular internal state obtained by processing $\mathbf{source}$ is used in place of the corresponding internal state obtained by $\mathbf{base}$. For a given set of variables $\mathbf{V}$,

$$\mathcal{M}_{\mathbf{V}\leftarrow \textsc{GetVals}(\mathcal{M},\mathbf{source},\mathbf{V})}$$

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

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

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

$$\textsc{IntInv}(\mathcal{M}, \mathbf{base}, \mathbf{source}, \mathbf{V}) \overset{\text{def}}{=} \textsc{GetVals}(\mathcal{M}_{\mathbf{V}\leftarrow \textsc{GetVals}(\mathcal{M},\mathbf{source},\mathbf{V})}, \mathbf{base}, \mathbf{V}_{\text{Out}}) \quad (1)$$

In short, the interchange intervention provides the output of the model $\mathcal{M}$ for the input $\mathbf{base}$, except the variables $\mathbf{V}$ are set to the values they would have if $\mathbf{source}$ 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, κ (for example, if the low level model produces a probability distribution over output classes, then κ could be the $\arg\max$ function, which selects the highest probability class). Further suppose we have an alignment Π 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 $\mathbf{V}_{\mathit{In}}$, $\mathbf{V}_{\mathit{Out}}$, 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 $\mathbf{b}, \mathbf{s} \in \mathbf{V}_{\mathit{In}}$:

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

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 have the same effect. This is especially suited for situations in which we seek to relate small symbolic models with large neural models with 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* ($\textsc{IntInvAcc}$):

$$\textsc{IntInvAcc}(\mathcal{M}_{\mathcal{H}}, \mathcal{M}_{\mathcal{L}}, V_H, \Pi) \overset{\text{def}}{=} \frac{1}{|\mathsf{Val}(\mathbf{V}_{\mathit{In}})|^2} \sum_{\mathbf{b},\mathbf{s}\in\mathsf{Val}(\mathbf{V}_{\mathit{In}})} \mathbb{I}\Big[\textsc{IntInv}(\mathcal{M}^*_{\mathcal{H}}, \mathbf{b}, \mathbf{s}, V_H) = \kappa\big(\textsc{IntInv}(\mathcal{M}_{\mathcal{L}}, \mathbf{b}, \mathbf{s}, \Pi(V_H))\big)\Big] \quad (3)$$

Where every pair of inputs $\mathbf{b}$ and $\mathbf{s}$ is considered, and $\textsc{IntInvAcc}$ 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.

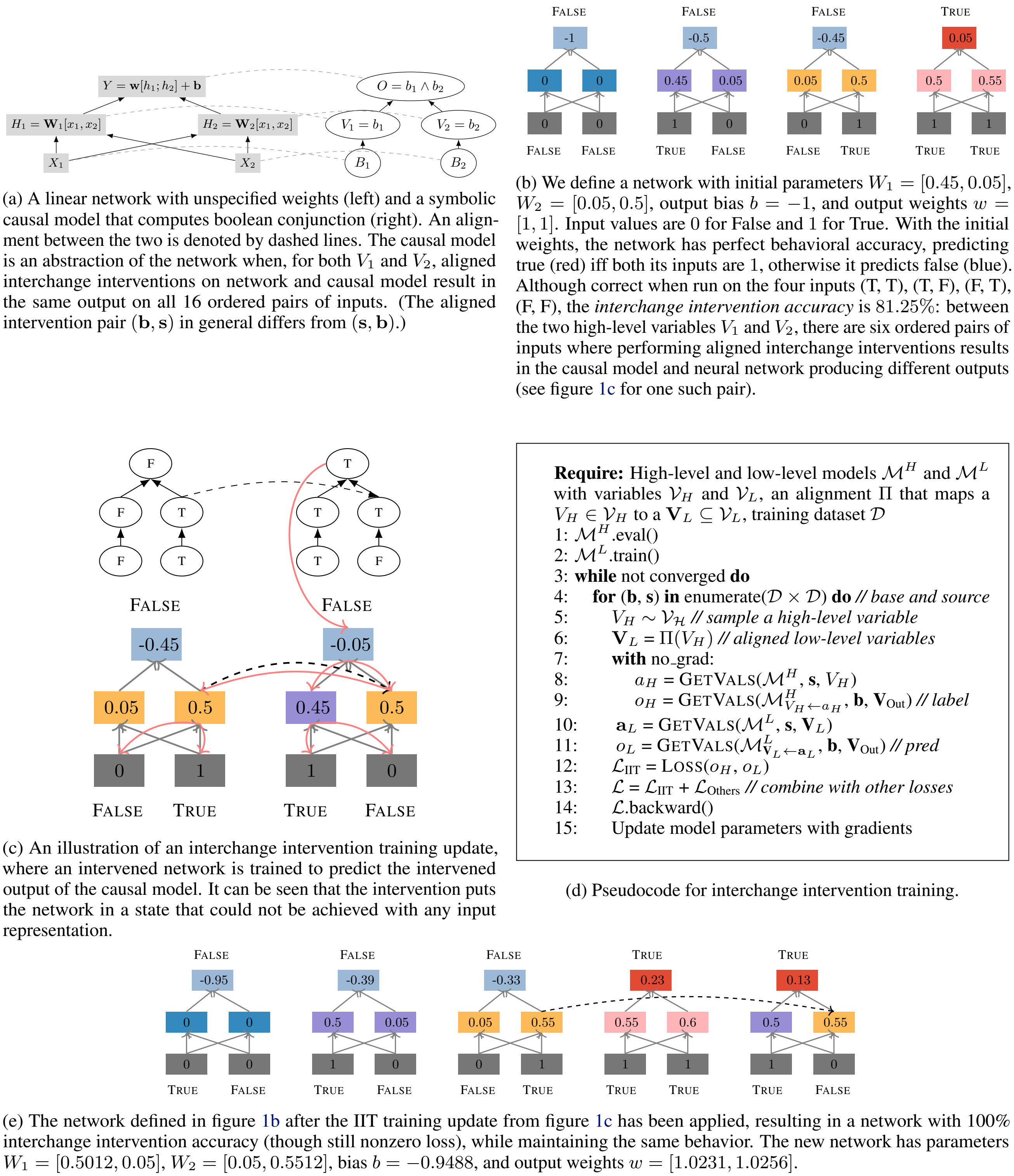
$\textsc{IntInvAcc}$ provides a natural metric for quantifying the interpretability of a neural network in the following sense: when $\textsc{IntInvAcc}$ is 1, the causal model is an explanation of how the network behaves, providing a clear window into the network itself. In practice, we rarely observe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

$$\sum_{\mathbf{b},\mathbf{s}\in\mathbf{V}_{\mathit{In}}} \textsc{Loss}\Big(\textsc{IntInv}(\mathcal{C}, \mathbf{b}, \mathbf{s}, V), \textsc{IntInv}(\mathcal{N}^{\theta}, \mathbf{b}, \mathbf{s}, \Pi(V))\Big) \quad (4)$$

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 θ, $\Pi(V)$ is a set of low-level variables (neurons) that are aligned with V, and $\textsc{Loss}$ is some loss function. Observe that we do not apply the output map κ, because the loss function takes in the logits directly.

The crucial feature of an IIT update is that the interchange intervention intertwines two computation graphs, one gener-



(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 $(\mathbf{b}, \mathbf{s})$ in general differs from $(\mathbf{s}, \mathbf{b})$.)

(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 0 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 results 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 $\mathcal{D}$
$\mathcal{M}^H$.eval()
$\mathcal{M}^L$.train()
while not converged do
    for ($\mathbf{b}$, $\mathbf{s}$) in enumerate($\mathcal{D} \times \mathcal{D}$) do // base and source
        $V_H \sim \mathcal{V}_\mathcal{H}$ // sample a high-level variable
        $\mathbf{V}_L = \Pi(V_H)$ // aligned low-level variables
        with no_grad:
            $a_H$ = GETVALS($\mathcal{M}^H$, $\mathbf{s}$, $V_H$)
            $o_H$ = GETVALS($\mathcal{M}^H_{V_H \leftarrow a_H}$, $\mathbf{b}$, $\mathbf{V}_{\text{Out}}$) // label
        $\mathbf{a}_L$ = GETVALS($\mathcal{M}^L$, $\mathbf{s}$, $\mathbf{V}_L$)
        $o_L$ = GETVALS($\mathcal{M}^L_{\mathbf{V}_L \leftarrow \mathbf{a}_L}$, $\mathbf{b}$, $\mathbf{V}_{\text{Out}}$) // pred
        $\mathcal{L}_{\text{IIT}}$ = LOSS($o_H$, $o_L$)
        $\mathcal{L}$ = $\mathcal{L}_{\text{IIT}}$ + $\mathcal{L}_{\text{Others}}$ // combine with other losses
        $\mathcal{L}$.backward()
        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 $W_1 = [0.5012, 0.05]$, $W_2 = [0.05, 0.5512]$, 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).

ated 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 $\mathbf{base}$, and once from $\mathcal{N}^\theta$ processing $\mathbf{source}$. In our toy example (figure 1c), the network is too small to observe this double update, but the networks in our three case studies are not. (See figure 2, which exemplifies such a process.)

An important property of IIT is that, if Eqn. 4 is zero, 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 zero. Figure 1 is an example. This is a desirable property of the method, since we do not expect our loss functions to be zero 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 $\mathcal{N}^\theta_\wedge$, where $\theta = \{W_1, W_2, b, w\}$, as in figure 1b.

At the start, $\mathcal{N}^\theta_\wedge$ 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.

One interchange intervention training update (figure 1c) results in a network that satisfies both objectives (figure 1e): $\mathcal{N}^\theta_\wedge$ 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_{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 $\mathcal{C}_{\text{PVR}} = (\mathcal{V}, \mathit{PA}, \mathsf{Val}, F)$ that computes the label for each of the four MNIST images using an oracle O_{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 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 $\mathit{PA}_{I_w} = \emptyset$ and $\mathit{PA}_{Y_w} = \{I_w\}$ for all $w \in \{\text{TR}, \text{TL}, \text{BR}, \text{BL}\}$, and $\mathit{PA}_O = \{Y_{\text{TL}}, Y_{\text{TR}}, Y_{\text{BL}}, Y_{\text{BR}}\}$. The structural equations are

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

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 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 $\mathcal{N}^\theta_{\text{PVR}}$, where θ abbreviates the parameters.

Alignments In our experiments, we align the neural representations of $\mathcal{N}^\theta_{\text{PVR}}$ 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 $\mathcal{N}^\theta_{\text{PVR}}$ implementing $\mathcal{C}^w_{\text{PVR}}$ the submodel of $\mathcal{C}_{\text{PVR}}$ where the three intermediate variables that aren't Y_w are marginalized out:

$$\sum_{\mathbf{b},\mathbf{s} \in \text{MNIST-PVR}} \text{CE}\Big(\textsc{IntInv}(\mathcal{C}^w_{\text{PVR}}, \mathbf{b}, \mathbf{s}, Y_w), \textsc{IntInv}(\mathcal{N}^\theta_{\text{PVR}}, \mathbf{b}, \mathbf{s}, \mathbf{Q}_w)\Big) \quad (5)$$

where CE is the cross-entropy loss and MNIST-PVR is the dataset. We visualize an IIT update to $\mathcal{N}^\theta_{\text{PVR}}$ 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 $\mathbf{Q}_{\text{TL}}, \mathbf{Q}_{\text{TR}}, \mathbf{Q}_{\text{BL}}, \mathbf{Q}_{\text{BR}}$. This means we can perform interchange interventions between different variables and

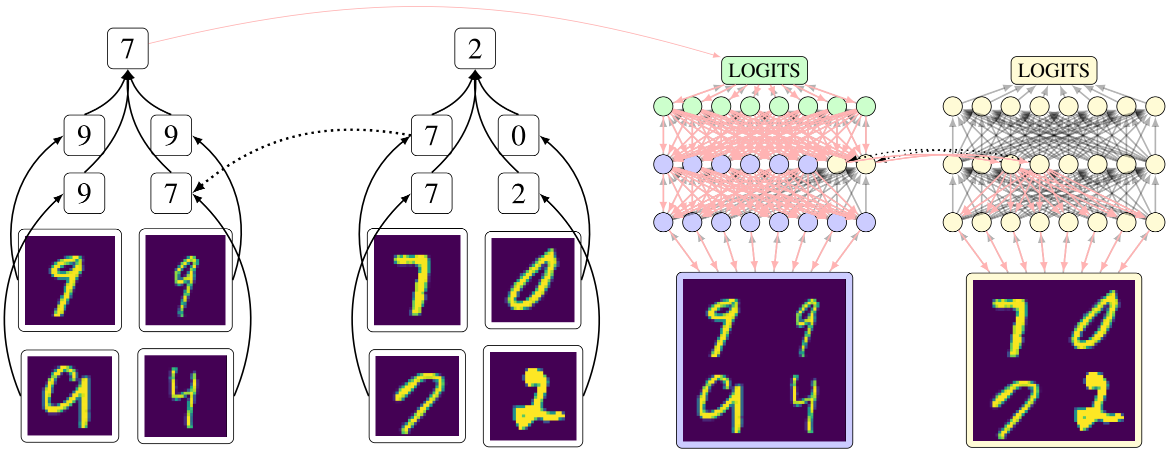


Figure 2. An illustration of an IIT update where a neural network (right) is trained to realize a causal model (left) that solves the PVR-MNIST task. Solid lines are feed-forward connections, dashed lines are interchange interventions, red lines are the flow of backpropagation. Observe that when backpropagation 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.

extend our training objective to these interventions as well:

$$\text{T-I}\textsc{nt}\text{I}\textsc{nv}(\mathcal{M}, \mathbf{b}, \mathbf{s}, \mathbf{V}, \mathbf{V}') \overset{\text{def}}{=} \text{G}\textsc{et}\text{V}\textsc{als}(\mathcal{M}_{\mathbf{V}' \leftarrow \text{G}\textsc{et}\text{V}\textsc{als}(\mathcal{M}, \mathbf{s}, \mathbf{V})}, \mathbf{b}, \mathbf{V}_{\text{Out}}) \quad (6)$$

$$\sum_{\substack{w, w' \in \{\text{TL, TR, BL, BR}\} \\ \mathbf{b}, \mathbf{s} \in \text{PVR-MNIST}}} \text{CE}\Big(\text{T-I}\textsc{nt}\text{I}\textsc{nv}(\mathcal{C}_{\text{PVR}}, \mathbf{b}, \mathbf{s}, Y_w, Y_{w'}), \text{T-I}\textsc{nt}\text{I}\textsc{nv}(\mathcal{N}^{\theta}_{\text{PVR}}, \mathbf{b}, \mathbf{s}, \mathbf{Q}_w, \mathbf{Q}_{w'})\Big) \quad (7)$$

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

$$\sum_{\substack{\mathbf{input} \in \text{PVR-MNIST} \\ w \in \{\text{TL,TR,BL,BR}\}}} \text{CE}(\mathcal{P}^{\varphi_w}(\text{G}\textsc{et}\text{V}\textsc{als}(\mathcal{N}^{\theta}_{\text{PVR}}, \mathbf{input}, \mathbf{Q}_w)), \text{G}\textsc{et}\text{V}\textsc{als}(\mathcal{C}_{\text{PVR}}, \mathbf{input}, Y_w)) \quad (8)$$

where the trained parameters are θ, the parameters of ResNet and φ_w, the parameters of the linear classifiers.

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.

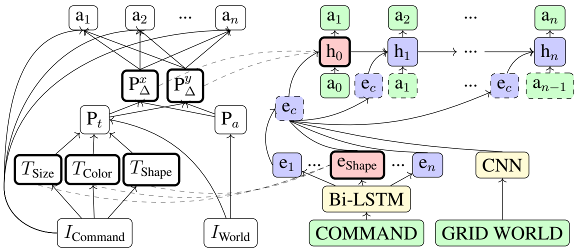
Results Our results are in Table 1. The behavioral accuracy is the standard metric, while the interchange intervention accuracy captures whether the symbolic causal model is an abstraction of the neural network.

Training	**Behavioral Accuracy**		**IIT 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	**96.01**	99.10	**96.64**
AUGMENT	99.40	90.90	98.90	92.00
NO TYPING	99.41	0.09	99.47	16.88

Table 1. Results for $\mathcal{N}^{\theta}_{\text{PVR}}$ (ResNet18) trained on the PVR-MNIST dataset. Behavioral accuracy is the percentage of inputs that $\mathcal{N}^{\theta}_{\text{PVR}}$ 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.

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 does not exceed 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



(a) The causal model that solves ReaSCAN (left) and the neural CNN-LSTM model trained on ReaSCAN (right). Dashed lines align variables in the causal model with neural representations.

Training	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	**80.37** (0.88)	**74.84** (0.04)	**14.72** (3.54)	**25.82** (0.37)
	Interchange Intervention Exact Match %			
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	**70.73** (6.86)	**75.34** (0.91)	**11.79** (2.57)	**8.49** (1.53)

(b) 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.

Figure 3.

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 $\mathcal{C}_{\text{PVR}}$ 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 $\mathcal{C}_{\text{PVR}}$.

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).

Without our typed IIT objectives, behavioral and interchange intervention accuracy plummets on the test data. Typing our variables is crucial for generalization.

5. Navigation and Language (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.

Symbolic Causal Structure Our causal model $\mathcal{C}_{\text{ReaSCAN}} = (\mathcal{V}, \mathit{PA}, \mathsf{Val}, F)$ (see figure 3a 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 (We condense the action sequence to a single output variable). Formally, we define variables and values

$$\mathcal{V} = \{I_{\text{Com}}, I_{\text{World}}, T_{\text{Size}}, T_{\text{Color}}, T_{\text{Shape}}, \mathrm{P}_t, \mathrm{P}_a, \mathrm{P}^x_\Delta, \mathrm{P}^y_\Delta, O\}$$
$$\mathsf{Val}(T_{\text{Shape}}) = \{\text{circle}, \text{square}, \text{cylinder}\}$$
$$\mathsf{Val}(T_{\text{Color}}) = \{\text{red}, \text{green}, \text{blue}, \text{yellow}\}$$
$$\mathsf{Val}(T_{\text{Size}}) = \{\text{small}, \text{big}\}$$
$$\mathsf{Val}(\mathrm{P}_t) = \mathsf{Val}(\mathrm{P}_a) = \mathsf{Val}(\mathrm{P}^{x/y}_\Delta) = \{-5, \ldots, 5\}$$

with the values $\mathsf{Val}(I_{\text{Com}})$, $\mathsf{Val}(I_{\text{world}})$, and $\mathsf{Val}(O)$ 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 3a.

The structural 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 structural equations for position look-ups $F_{\mathrm{P}_t}(t_{\text{Size}}, t_{\text{Color}}, t_{\text{Shape}}, i_{\text{World}})$ and $F_{\mathrm{P}_a}(i_{\text{World}})$ determine the target object and agent location from the target object properties and the input world. The position deltas $F_{\mathrm{P}^x_\Delta}(\mathrm{p}_t, \mathrm{p}_a)$ and $F_{\mathrm{P}^y_\Delta}(\mathrm{p}_t, \mathrm{p}_a)$ are determined to be the horizontal and vertical distance between the target object and agent, respectively. Finally, the equation for the output $F_O(\mathrm{P}^x_\Delta, \mathrm{P}^y_\Delta, i_{command})$ is the action sequence that takes the agent to the target object in the correct manner of movement, as determined by the vertical and horizontal distances between the two and the adverb in the command.

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. 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 on splits.

Neural Network We use the original baseline model for ReaSCAN (Wu et al., 2021) as our neural model $\mathcal{N}^{\theta}_{\text{CNN-LSTM}}$. $\mathcal{N}^{\theta}_{\text{CNN-LSTM}}$ 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 3a. We include details about the model and experimental set-up in Appendix B.

Alignments In our experiments, we align neural representations of $\mathcal{N}^{\theta}_{\text{CNN-LSTM}}$ with the variables T_{Size}, T_{Color}, T_{Shape}, $\text{P}^{x}_{\triangle}$, and $\text{P}^{y}_{\triangle}$, 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_{Size}, T_{Color}, and T_{Shape}. For the position deltas, we choose the initial hidden representation $\mathbf{h}_0$ 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 $\text{P}^{y}_{\triangle}$, and the second chunk represents the position difference by column $\text{P}^{x}_{\triangle}$. We train the network to derive from the world and the command the horizontal and vertical distances between the target and agent, storing the horizontal distance in one half of $\mathbf{h}_0$ and the vertical distance in the other.

Interchange Intervention Training For each variable V in $\mathcal{C}_{\text{ReaSCAN}}$ aligned with neurons $\mathbf{N}_V$ in $\mathcal{N}^{\theta}_{\text{CNN-LSTM}}$, we introduce an IIT objective that optimizes for $\mathcal{N}^{\theta}_{\text{CNN-LSTM}}$ implementing the marginalized submodel $\mathcal{C}^{V}_{\text{ReaSCAN}}$:

$$\sum_{\mathbf{b},\mathbf{s}\in\text{ReaSCAN}} \text{CE}_{\text{Action}}\Big(\text{IntInv}(\mathcal{N}^{\theta}_{\text{CNN-LSTM}}, \mathbf{b}, \mathbf{s}, \mathbf{N}_V),\\ \text{IntInv}(\mathcal{C}^{V}_{\text{ReaSCAN}}, \mathbf{b}, \mathbf{s}, V)\Big) \quad (9)$$

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 target and the agent from the aligned neural representations. Specifically, for each $V \in \{T_{\text{Size}}, T_{\text{Color}}, T_{\text{Shape}}, \text{P}^{x}_{\triangle}, \text{P}^{y}_{\triangle}\}$, we train a single-layer linear classifier $\mathcal{P}^{\varphi_V}$ on the loss

$$\sum_{\mathbf{i}\in\text{ReaSCAN}} \text{CE}_{\text{Position}}\Big(\mathcal{P}^{\varphi_V}(\text{GetVals}(\mathcal{N}^{\theta}_{\text{CNN-LSTM}}, \mathbf{i}, \mathbf{N}_V),\\ \text{GetVals}(\mathcal{C}_{\text{ReaSCAN}}, \mathbf{i}, V)\Big) \quad (10)$$

where the trained parameters are θ, the parameters of the CNN-LSTM, and φ_V, the parameters of the classifiers.

Results Our results are shown in Table 3b. 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 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 only for 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. Natural Language Inference (MQNLI)

Our final benchmark is MQNLI (Geiger et al., 2019), a synthetic natural language inference dataset where the task is to label the semantic relation between two sentences as **enailment**, **contradiction**, or **neutral**. Here is an example:

ε every ε baker ε ε happily eats ε some stale bread
contradiction
ε some angry baker does not ε eat ε some ε bread

where an ε denotes the absence of a word and is used to align corresponding words in the two sentences.

Geiger et al. (2020) fine-tuned a BERT model on MQNLI, achieving state-of-the-art results (≈90% test accuracy). Their interchange intervention analysis revealed that this model learns to partially represent the relation between aligned subphrases in the two sentences (e.g. *stale bread* and ε *bread* in the example above) We hypothesize that if we *teach* BERT to fully represent this information using IIT, the task will be solved perfectly.

Symbolic causal structure

Geiger et al. (2020) define a causal model that computes the relation between aligned phrases in order to compute the relation between two sentences. We narrow our focus to the submodel $\mathcal{C}_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}$, which (1) contains a single intermediate variable QP_{Obj} that computes the relation between the *quantified verb phrase* (= adverb + verb + quantified object noun phrase) of each sentence, and (2) uses this to infer the relation between the two sentences. To label the MQNLI example above, $\mathcal{C}_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}$ would compute that $\text{QP}_{\text{Obj}} = \sqsubset$ because "*happily eats* ε *some stale bread*" **entails** "*eat* ε *some* ε *bread*". Then, this information is used infer that the relation between the sentences is **contradiction**.

Systematic Generalization The train-test split of MQNLI is constructed to be as difficult as possible while still being solved by a compositional memorization-based learning model. This makes the task hard, but fair.

Neural Network We fine-tune a pretrained BERT model $\mathcal{N}_{\text{NLI}}^{\theta}$ on MQNLI. The architecture consists of 12 transformer layers that create a neural representations for each token in the input; the grid of neural representations has a column for each token and a row for each layer.

Alignments In our experiments, we align QP_{Obj} with the neural representations above the verb in the first sentence from BERT layers $\{0, 2, 4, 6, 8, 10\}$.

Interchange intervention training For QP_{Obj} in $\mathcal{C}_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}$ aligned with neurons $\mathbf{N}$ in $\mathcal{N}_{\text{NLI}}^{\theta}$, we introduce an IIT objective that optimizes for $\mathcal{N}_{\text{NLI}}$ implementing the marginalized submodel $\mathcal{C}_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}$:

$$\sum_{\mathbf{b},\mathbf{s} \in \text{MQNLI}} \text{CE}\Big(\textsc{IntInv}(\mathcal{N}_{\text{NLI}}^{\theta}, \mathbf{b}, \mathbf{s}, \mathbf{N}), \textsc{IntInv}(\mathcal{C}_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}, \mathbf{b}, \mathbf{s}, \text{QP}_{\text{Obj}})\Big) \tag{11}$$

Multi-task Objectives We train a linear classifier $\mathcal{P}^{\varphi}$ to predict the value of QP_{Obj} with loss:

$$\sum_{\mathbf{i} \in \text{MQNLI}} \text{CE}\Big(\mathcal{P}^{\varphi}(\textsc{GetVals}(\mathcal{N}_{\text{NLI}}^{\theta}, \mathbf{i}, \mathbf{N}), \textsc{GetVals}(\mathcal{C}_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}, \mathbf{i}, \text{QP}_{\text{Obj}})\Big) \tag{12}$$

where the trained parameters are θ, the parameters of BERT, and φ, the parameters of the classifier.

Data Augmentation We perform data augmentation by randomly sampling two examples and replacing the quantified verb phrase from the first example with those from the second in order to produce a new example that is then labeled with $\mathcal{C}_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}$. This procedure is guided by the same causal structure used by our other models, but it is by definition restricted to input manipulations.

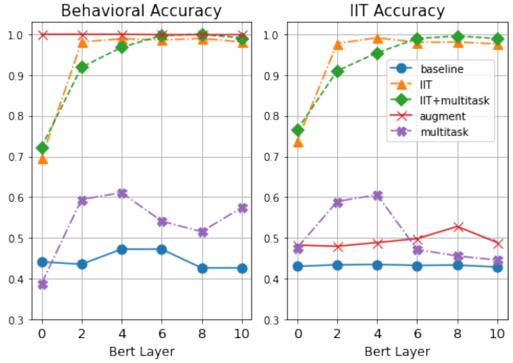


Figure 4. Performance of a pretrained BERT natural language inference model fine-tuned on the MQNLI dataset with the causal model $\mathcal{C}_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}$ from Geiger et al. (2020). We report the results on the evaluation set. While data augmentation leads to consistently excellent behavior accuracy (left) panel, it has very low interchange intervention accuracy. In other words, IIT is necessary for an interpretable model with high-performance.

Results We compute the accuracy on the basic behavioral task of predicting MQNLI labels, and interchange intervention (IIT) accuracy, where we compute the percentage of cases where performing an intervention on the neural model produced a same change in output as the submodel $\mathcal{C}_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}$.

Our results, shown in figure 4, demonstrate that IIT training on $\mathcal{C}_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}$ solves MQNLI with near perfect accuracy and IIT accuracy (≈100%). Furthermore, we see that aligning QP_{Obj} to the first few layers of BERT results in lower accuracy on MQNLI, with layer 6 and onward resulting in near perfect accuracy and interchange intervention accuracy. When QP_{Obj} is aligned with layer 6 or later, we again see multi-task training synergizing with IIT to produce the best models.

Data augmentation results in models with perfect behavioral performance. This is unsurprising, as data augmentation removes the out-of-domain generalization problem. IIT is needed to produce a model with an interpretable solution.

7. Conclusion

We introduced interchange intervention training as a method to imbue neural networks with interpretable, systematic causal structure, and we conducted three case studies with IIT: a vision task (MNIST-PVR), a grounded language understanding task (ReaSCAN), and a natural language inference task (MQNLI). In all 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

Bau, D., Zhu, J., Strobelt, H., Zhou, B., Tenenbaum, J. B., Freeman, W. T., and Torralba, A. GAN dissection: Visualizing and understanding generative adversarial networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Hyg_X2C5FX.

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.

Besserve, M., Mehrjou, A., Sun, R., and Schölkopf, B. Counterfactuals uncover the modular structure of deep generative models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=SJxDDpEKvH.

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.

Csordás, R., van Steenkiste, S., and Schmidhuber, J. Are neural nets modular? inspecting functional modularity through differentiable weight masks. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=7uVcpu-gMD.

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. CausaLM: 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.

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 Analyz-*

ing 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.

Hewitt, J. and Liang, P. Designing and interpreting probes with control tasks. 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. 2733–2743, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1275. URL https://www.aclweb.org/anthology/D19-1275.

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., Veldhoen, S., and Zuidema, W. H. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *J. Artif. Intell. Res.*, 61:907–926, 2018. doi: 10.1613/jair.1.11196. URL https://doi.org/10.1613/jair.1.11196.

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.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in gpt, 2022. URL https://arxiv.org/abs/2202.05262.

Pearl, J. *Causality: Models, Reasoning and Inference*. Cambridge University Press, USA, 2nd edition, 2009. ISBN 052189560X.

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. *CoRR*, abs/1706.05098, 2017. URL http://arxiv.org/abs/1706.05098.

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. Zero Loss Entails Causal Abstraction

Claim Suppose we have a loss function $\textsc{Loss}$ that outputs a non-negative value. If $\textsc{Loss}(x, y) = 0 \Rightarrow x = \kappa(y)$, 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

$$\sum_{\mathbf{b},\mathbf{s}\in\mathbf{V}_{\mathit{In}}} \textsc{Loss}\Big(\textsc{IntInv}(\mathcal{C}, \mathbf{b}, \mathbf{s}, V), \textsc{IntInv}(\mathcal{N}^{\theta}, \mathbf{b}, \mathbf{s}, \Pi(V))\Big) = 0 \quad (13)$$

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:

$$\forall\, \mathbf{b}, \mathbf{s} \in \mathbf{V}_{\mathit{In}} : \textsc{Loss}\Big(\textsc{IntInv}(\mathcal{C}, \mathbf{b}, \mathbf{s}, V), \textsc{IntInv}(\mathcal{N}^{\theta}, \mathbf{b}, \mathbf{s}, \Pi(V))\Big) = 0 \quad (14)$$

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

$$\forall\, \mathbf{b}, \mathbf{s} \in \mathbf{V}_{\mathit{In}} : \textsc{IntInv}(\mathcal{C}, \mathbf{b}, \mathbf{s}, V) = \kappa(\textsc{IntInv}(\mathcal{N}^{\theta}, \mathbf{b}, \mathbf{s}, \Pi(V))) \quad (15)$$

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

B. ReaSCAN

Dataset Generation

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 2. Statistics of all splits in our ReaSCAN dataset.

Table 2 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 framework[2] 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.[3] 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.

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.

[2] The implementation is adapted from ReaSCAN’s public code repository: `https://github.com/frankaging/Reason-SCAN`.

[3] `https://github.com/LauraRuis/multimodal_seq2seq_gSCAN`

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-modalitiy attention weights over the command and the grid world.

C. Experiment Details of Multiply Quantified Natural Language Inference (MQNLI)

Dataset For our experiments, we used a train set with 500K examples, a dev set with 60k examples, and a test set with 10K examples – the most difficult generalization scheme of Geiger et al. (2019). Unlike Geiger et al. (2020) we do not include extra augmented examples consisting of subphrases with labeled relations, but only include complete sentences. Only the QP$_{Obj}$ labels are used to introduce IIT and multi-task training objectives.

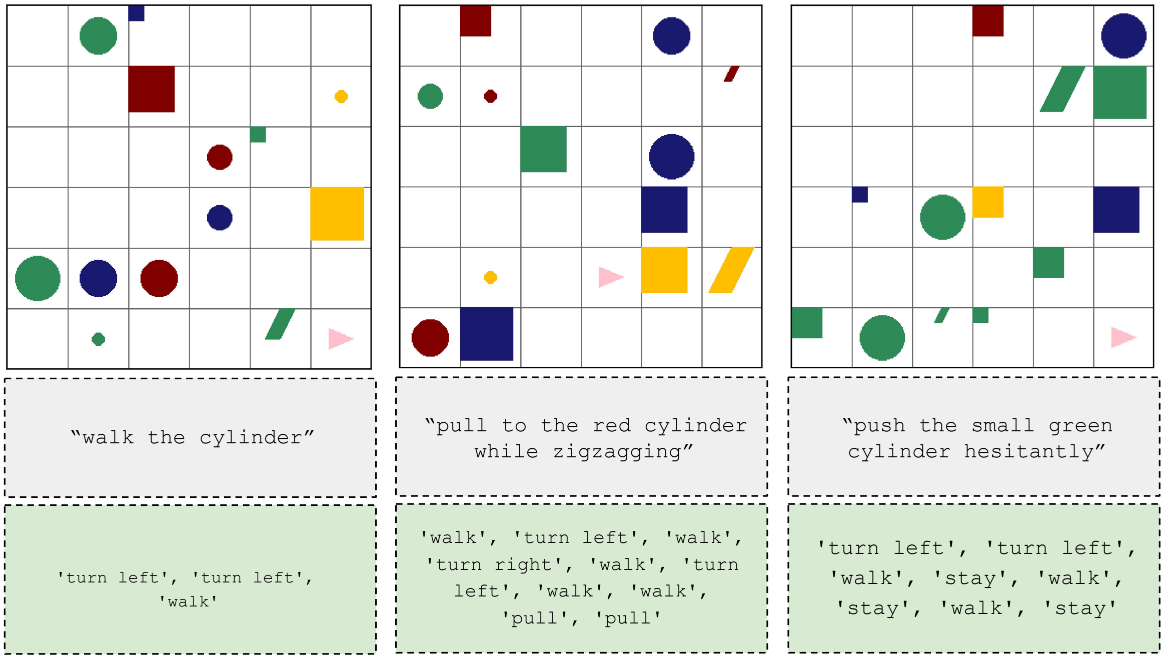
Models For BERT, we use the same model architecture for the `uncased-base` variant, with 12 layers and a hidden size of 768.

For the linear classifier that predicts the value of QP_{Obj} given a neural representation, we use a similar model as the probes used in Geiger et al. (2020). It is a single-layer softmax classifier: $y \propto \text{softmax}(Ah + b)$ where h is a hidden representation and y is the predicted probability distribution of each class of QP_{Obj}. Following Hewitt & Liang (2019), to control the dimensionality of A, we factorize it in the form $A = LR$ where $L \in \mathbb{R}^{|\text{QP}_{\text{Obj}}| \times \ell}$ and $R \in \mathbb{R}^{\ell \times d}$ where d is the dimensionality of h. We choose $\ell = 32$.

Training Procedure We use a batch size of 32. We use 5.0×10^{-5} as our learning rate, and use `adamw` optimization. We train for a maximum of 5 epochs. We warm up the learning rate linearly from 0 to the specified value in the first 50% of steps of the first epoch, and linearly decrease the learning rate to 0 following that until the end of training.

To construct each batch sample during training, we randomly pick two samples from the MQNLI dataset. We construct the augmented input by replacing the quantified verb phrase in the first example with that in the second one. For counterfactual training, we use `antra` package, which is built off of the implementation in Geiger et al. (2020), which we construct an intervention with the first example as the base, and the second example as the intervention source. For multi-task training, we only use the hidden representation of the first example as input to the linear classifier. We sample 50000 base examples from the train set of MQNLI. For each base example we sample 20 intervention sources, and try to ensure that 10 of them will be able to construct *impactful* interventions with the base example, i.e. the logical model $C_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}$ computes that doing an intervention will change the final label. We use the same $50000 \times 20 = 1\text{M}$ pairs for each training epoch.

We use the logical model $C_{\text{NatLog}}^{\text{QP}_{\text{Obj}}}$ as our oracle to obtain all of the labels for the base output, counterfactual output, augmented output, and the value of QP_{Obj} used for the linear classifier. For each of the experiment settings (base, IIT, IIT+multitask, augment, multitask), we always assign a weight of 1.0 to the base objective plus each additional objective, e.g. IIT + multitask uses a weight of 1.0 for each of base, IIT and multitask.



"walk the cylinder"

'turn left', 'turn left', 'walk'

"pull to the red cylinder while zigzagging"

'walk', 'turn left', 'walk', 'turn right', 'walk', 'turn left', 'walk', 'walk', 'pull', 'pull'

"push the small green cylinder hesitantly"

'turn left', 'turn left', 'walk', 'stay', 'walk', 'stay', 'walk', 'stay'

Figure 5. ReaSCAN examples with varying command patterns. The navigation commands and the target action sequences are in the grey boxes and green boxes respectively.