Ignorance is Bliss: Robust Control via Information Gating

Manan Tomar[1, 2], **Riashat Islam**[2], **Sergey Levine**[3] **and Philip Bachman**[2]
[1]University of Alberta, [2]Microsoft Research Montreal, [3]UC Berkeley

Informational parsimony — i.e., using the minimal information required for a task, — provides a useful inductive bias for learning representations that achieve better generalization by being robust to noise and spurious correlations. We propose information gating in the pixel space as a way to learn more parsimonious representations. Information gating works by learning masks that capture only the minimal information required to solve a given task. Intuitively, our models learn to identify which visual cues actually matter for a given task. We gate information using a differentiable parameterization of the signal-to-noise ratio, which can be applied to arbitrary values in a network, e.g. masking out pixels at the input layer. We apply our approach, which we call InfoGating, to various objectives such as: multi-step forward and inverse dynamics, Q-learning, behavior cloning, and standard self-supervised tasks. Our experiments show that learning to identify and use minimal information can improve generalization in downstream tasks — e.g., policies based on info-gated images are considerably more robust to distracting/irrelevant visual features.

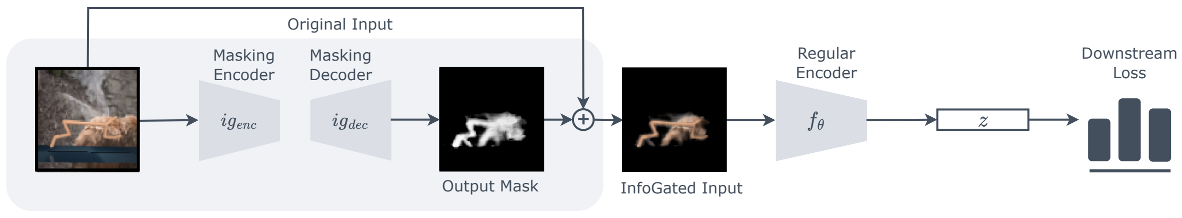


Figure 1: InfoGating. We learn information masks conditioned on the original input and then process the masked input to learn representations corresponding to a downstream loss. The information masks and the representation are trained together. The mask is encouraged to remove as much of the original input as possible without hurting performance on the downstream task.

1. Introduction

Pretraining models on large, diverse datasets and transferring their representations to downstream applications is becoming common practice in deep learning. Such representations should capture useful information from the available data while ignoring irrelevant features and noise (Efroni et al., 2021; Dietterich et al., 2018). For instance, an object recognition model may achieve high accuracy on its training set while relying strongly on "spurious" correlations between background features and the actual objects. When transferred to new data where objects are the same but the distribution of backgrounds is different, performance may suffer. Similarly, a policy for vision-based robot control (Levine et al., 2016; Finn et al., 2016) may perform well in its original training environment but fail catastrophically when transferred to a new setting (Nagabandi et al., 2019) where all task-relevant features are the same but minor background features differ from training.

When data for a domain is limited, various approaches to regularization have proven useful for improving the robustness and generalization of representations. Though these methods are helpful, the means by which they improve model robustness are often unclear. For instance, dropout (Srivastava et al., 2014) encourages representations which distribute useful information more broadly across a network's internal features/activations, but does not explicitly reduce the amount of information from the input which is present in those representations. One can seek "disentangled" representations by forcing dimensions of the representations to be independent. While the set of disentangled distributions over features is strictly

Corresponding author(s): Manan Tomar (manan.tomar@gmail.com)

smaller, and thus in some sense simpler, than the set of all distributions over features, disentanglement does not say much about the salience of the information captured in the features.

A promising principle for encouraging robustness to out-of-distribution observations is informational parsimony, i.e. learning features which contain minimal information about the input while still containing enough information to solve the task(s) of interest. Such regularization is usually based on imposing some sort of *information bottleneck* (IB) on the network which tries to minimize the amount of information flowing from the input, through the bottleneck, to the output prediction. Although IB approaches have been beneficial in some cases, adding an information bottleneck at the penultimate step of computation does little to prevent overfitting in the preceding steps of computation, which typically comprise the overwhelming bulk of the model. To that end, we consider measuring the information used throughout a computation rather than just measuring the information which comes out of a computation. For instance, restricting information in the input itself leads to minimizing information used throughout the computation model.

In this paper, we propose learning input-conditioned functions that *gate* the flow of information between the input and the rest of the model. Intuitively, we consider what happens when one inserts an IB near the beginning of a model's computation rather than near the end. We train the information gating functions to minimize information flow from the input into the rest of the model while still permitting the model to solve the task(s) of interest. We implement this gating using a differentiable parameterization of the signal-to-noise ratio. As the noise level goes up, the signal level goes down, and hence the model displays reduced dependencies with respect to the input signal. Such a parameterization can be learnt in conjunction to any downstream loss function corresponding to the task of interest. For example, the downstream loss could be a standard contrastive loss for self-supervised learning or an inverse dynamics loss when learning representations for reinforcement learning (RL). In both cases, the information gates should *reveal minimal information* which is needed for optimizing the loss. We also explore an alternative formulation, i.e. learning gating functions which seek to *remove any information* which can be used to optimize the loss. We primarily focus on InfoGating representations used for learning control policies. Using background distractors and multi-object interactions as a form of noise/irrelevant features, we see that InfoGating is able to remove almost all of such irrelevant information from the pixels, thus leading to better out-of-distribution generalization performance in the presence of noise and better in-distribution generalization performance in the presence of multiple objects.

Our main contributions are as follows: **1.** A general purpose practical framework for informational parsimony called InfoGating which can restrict information flow throughout a computation to learn robust representations. **2.** Qualitative analyses on the properties of the gating functions learned with InfoGating that show they are semantically meaningful and enhance intrepretability. **3.** Quantitative analyses of applying InfoGating in the context of various downstream objectives which show clear benefits with regards to improved generalization performance.

2. Related Work

For our purposes, and simplifying aggressively, we can group methods which pursue minimal representations using two bits. For the first bit, the two possibilities are: methods which augment the available set of samples by applying perturbations directly to the data (e.g. standard data augmentation or training with adversarial samples), and methods which inject noise directly into the features produced by the model (e.g. dropout or Variational IB). For the second bit, the two possibilities are: methods where the objective is cooperative (e.g. Variational IB), and methods where the objective is adversarial (e.g. training with adversarial samples). Roughly speaking, cooperative approaches involve min-min optimization between the noise and the predictions while adversarial approaches involve min-max optimization. In this paper, we primarily consider a cooperative training regime with noise injected directly in the input space, which we believe is an under-explored regime.

Information Bottleneck. Much prior work in learning robust representations has stemmed from the idea of imposing an Information Bottleneck (IB) (Tishby and Zaslavsky, 2015; Tishby et al., 2000). Imposing an IB involves maximizing performance on the downstream task while removing as much information about the input as possible from a network's internal representations. Typical approaches based on Variational IB (Alemi et al., 2016) achieve this by learning stochastic representations which are constrained to be close to a standard Gaussian prior. Variants of dropout (Srivastava et al., 2014; Kingma et al., 2015) can be seen as having a similar effect, i.e. adding noise to representations used in the network, but without the explicit aim of maximizing noise/minimizing information in the representations. Contrary to applying an IB on representations used in the later stages of a network, InfoGating can be seen as enforcing an IB at the input itself, which is less explored in the IB literature. Like Variational IB methods, and unlike typical dropout-based methods, InfoGating learns how much noise can be added without hurting performance on the downstream task(s). Information Dropout (Achille and Soatto, 2018) is perhaps the most similar prior work to InfoGating. Information Dropout imposes IBs at multiple points in a network's computation and the IBs are implemented using techniques based on variational dropout (Kingma et al., 2015). A key distinction between InfoGating and Information Dropout is that InfoGating works post-hoc. With InfoGating a model can consider a computation it has already performed and predict ways in which it could produce the same result more efficiently (w.r.t. information from the input, FLOPs, etc.).

Finally, IB has also been studied in RL literature, often based on maximizing mutual information (MI) objectives (Kim et al., 2019; Bai et al., 2021), including stochastic VIB (Yingjun et al., 2021; Pei and Hou, 2019), and more recently for removal of exogenous/distractor information (Islam et al., 2022b).

Masked Image Modelling. There has been a lot of recent work in self-supervised learning where the input is masked and the model is tasked at reconstructing the missing pixels (He et al., 2022; Vincent et al., 2010; Assran et al., 2022). Most of these works operate with a fixed, random masking scheme, instead of learning it directly through the downstream loss. The amount of information masked is also fixed *a priori* whereas we aim at maximizing masking as much of the input as possible. Some related ideas include learning adversarial augmentations for producing new views of an input during contrastive learning (Tamkin et al., 2020) while also learning masks that essentially segment the input space (Shi et al., 2022).

Representation Learning in Reinforcement Learning. There is much prior work on learning representations for RL. This includes temporally contrastive learning (Mazoure et al., 2020), one-step inverse models (Pathak et al., 2017), learning through next state reconstruction (Hafner et al., 2019), bisimulation metrics (Ferns et al., 2011) and many more. Although some of these objectives have been shown to be useful in learning robust representations (Tomar et al., 2021), we see our method as being complementary to these approaches since InfoGating works with any downstream loss function, without needing knowledge of specific values like reward, next state etc. For instance, bisimulation and task-informed abstractions (Fu et al., 2021) learn to compress the observation space using reward information, while recent work on learning Denoised MDPs (Wang et al., 2022) aims at regularizing next state reconstruction using a variational objective.

3. Background

In this section, we describe the working of the primary downstream losses we use with InfoGating, namely an InfoNCE based contrastive loss, and a multi-step inverse dynamics loss.

3.1. Mutual Information Estimation via InfoNCE

Given $\mathbf{x}$ and $\mathbf{z}$ as two random variables, their mutual information (MI) can be defined as the decrease in uncertainty in observing $\mathbf{x}$ given $\mathbf{z}$ compared to just observing $\mathbf{x}$: $I(\mathbf{x}, \mathbf{z}) = H(\mathbf{x}) - H(\mathbf{x} \mid \mathbf{z})$, where H is the Shannon entropy. InfoNCE (Oord et al., 2018), based on Noise Contrastive Estimation (Gutmann and Hyvärinen, 2010), is a well known method to compute a lower bound on $I(\mathbf{z}^1, \mathbf{z}^2)$ where $\mathbf{z}^1$ and $\mathbf{z}^2$ are two representations of the input $\mathbf{x}$ produced by some encoder $f(x)$. Specifically, the bound is optimized by discriminating "positive" from "negative" pairs:

$$\mathcal{L}_{\text{InfoNCE}} = \mathbb{E}_{Z^-}\left[\log \frac{e^{\psi(\mathbf{z}^1,\, \mathbf{z}^2)}}{e^{\psi(\mathbf{z}^1,\, \mathbf{z}^2)} + \sum_{\mathbf{z}^- \in Z^-} e^{\psi(\mathbf{z}^1,\, \mathbf{z}^-)}}\right], \tag{1}$$

where ψ is a pairwise function of the representations and Z^- is a batch of "negative samples". Given two different views of an observation, self-supervised contrastive learning uses the InfoNCE objective to encourage the representations of two views of the same input $\mathbf{x}$ to be closer while pushing apart the representations of views of different inputs (Bachman et al., 2019; Chen et al., 2020). Different views of the input are typically generated through data augmentation such as random cropping and color jittering.

3.2. Multi-Step Inverse Dynamics Models

Multi-step inverse dynamics models (Efroni et al., 2021) predict the action(s) which took an agent from some observation $\mathbf{x}_t$ to some future observation $\mathbf{x}_{t+k}$ in the hope to learn a useful representation of the observations. This resembles learning goal-conditioned policies through relabelling future observations as achieved goals. Although the trivial case of $k = 1$, i.e. a one-step model, does not capture long term dependencies, recent work has shown that multi-step models are able to capture more information which may be useful for controlling the agent (Lamb et al., 2022). Multi-step inverse models can be used to learn representations by simply predicting the actions $(\mathbf{a}_t, ..., \mathbf{a}_{t+k-1})$ conditioned on $\mathbf{x}_t$ and $\mathbf{x}_{t+k}$. Actions can be predicted using a standard max likelihood objective or with a contrastive reconstruction objective which maximizes the InfoNCE-based lowerbound on $I((\mathbf{x}_t, \mathbf{x}_{t+k}), (\mathbf{a}_t, ..., \mathbf{a}_{t+k-1}))$. The learning objective in this case looks as follows:

$$\mathcal{L} = \text{InfoNCE}\big((\mathbf{z}_t, \mathbf{z}_{t+k}), \mathbf{y}_{t:t+k-1}\big),$$

where $\mathbf{z}_t$ and $\mathbf{z}_{t+k}$ are representations computed from $\mathbf{x}_t$ and $\mathbf{x}_{t+k}$ and $\mathbf{y}_{t:t+k-1}$ is a representation computed from $(\mathbf{a}_t, ..., \mathbf{a}_{t+k-1})$. The negative samples in this case come from sampling random actions instead of the actual actions taken by the agent. In the simplest case, the representation $\mathbf{y}_{t:t+k-1}$ may depend only on $\mathbf{a}_t$.

4. Method

Overview. In this section, we first describe the basic components required to understand InfoGating. We then describe three different instantiations of InfoGating. Specifically, we first provide **1)** a description of applying InfoGating in the pixel input, which forms the main version of InfoGating we study in this paper. We then describe **2)** a similar variant but where InfoGating is done in the feature space, which helps provide similarities to previous work like Dropout and VIB. Finally, **3)** we describe an adversarial version (instead of a cooperative one as in **1)** and **2)**) where we turn the InfoGating objective on its head, so as to minimize masking the input while *not* being able to do the downstream task well.

We consider two components in order to learn parsimonious representations, first being a regular encoder of the input $\mathbf{x}$, denoted by $f(\mathbf{x})$, and second being an info gating function, denoted by $ig(\mathbf{x})$ (see Figure 1). We wish to gate information passing through any node in the computation graph that computes the encoding $\mathbf{z} = f(\mathbf{x})$. The info gating function $ig(\mathbf{x}) \in [0, 1]$ provides continuous-valued masks which describe where to erase information from the input or the internal representations of $f(\mathbf{x})$. In general, the goal of $ig(\mathbf{x})$ is to erase as much information from $f(\mathbf{x})$ as possible without hurting task performance.

InfoGating in Input Space. For this paper, we focus mainly on erasing information from the input. We apply info gating to an input $\mathbf{x}$ by taking a simple convex combination of $\mathbf{x}$ and random Gaussian noise ϵ, where the weighting for the combination is given by $ig(\mathbf{x})$:

$$\mathbf{x}^{ig} = ig(\mathbf{x}) \odot \mathbf{x} + (1 - ig(\mathbf{x})) \odot \epsilon, \tag{2}$$

where $\mathbf{x}^{ig}$ denotes the info gated input and $\odot$ denotes element-wise multiplication. An all-zero mask hence corresponds to complete noise (i.e. erasing all information from the input) while an all-one mask corresponds to keeping the original input as is. The function $ig(x)$ is learnt using the same downstream objective that is used to learn $f(\mathbf{x})$. We encourage the masks to remove information from the input by minimizing their L1 norm. This tends to produce sparse masks due to properties of the L1 norm. The overall objective for learning with InfoGating looks like:

$$\mathcal{L}_{ig,f} = \mathcal{L}_{\text{task}}\big(f(\mathbf{x}^{ig})\big) + \lambda \, \|ig(\mathbf{x})\|_1, \tag{3}$$

where $\mathcal{L}_{\text{task}}$ refers to any objective through which a useful representation $\mathbf{z}$ can be learnt. Note that when doing InfoGating, we minimize the downstream loss for the masked input $\mathbf{x}^{ig}$ (first term), instead of the original input $\mathbf{x}$ while masking out as much of the input as possible through the L1 penalty (second term). The λ coefficient controls how much of the input is masked. Since in principle any loss function can be used in conjunction with InfoGating, we consider multiple downstream objectives, such as contrastive learning, Q-learning, and behavior cloning. We describe the overall procedure in Algorithm 1, which assumes contrastive multi-step inverse dynamics modeling as the downstream objective.

InfoGating in Feature Space. Having considered gating the pixel-level input, we can also consider a

Algorithm 1 InfoGating Pseudocode

```
# f: regular encoder
# ig: masking net

for x, g, a in loader: # load a minibatch x with n
    samples
   x, g = aug(x), aug(g) # random crop aug

   ig_x, ig_g = ig(x), ig(g) # get info-gates

   x_ig = ig_x * x + (1 - ig_x) * nz_x # info-gate
        current state
   g_ig = ig_g * g + (1 - ig_g) * nz_g # info-gate
        goal state

   zx, zg = f(x_ig), f(g_ig) # encodings

   L = InfoNCE(zx, zg, a) + |ig_x| + |ig_g| # loss
   L.backward() # back-propagate
   update(f, ig) # SGD update
```

variant where $f(\mathbf{x})$ is masked instead of the input. Consider the following masking:

$$\mathbf{z}^{ig} = ig(\mathbf{x}) \odot \mathbf{z} + (1 - ig(\mathbf{x})) \odot \epsilon, \qquad \mathbf{z} = f(\mathbf{x}) \quad (4)$$

The training objective remains the same as in the pixel-level case, i.e. minimize the downstream loss $\mathcal{L}_{\text{task}}(\mathbf{z}^{ig})$ for the masked representation $\mathbf{z}^{ig}$ while masking as much of $\mathbf{z}$ as possible (Figure 2). This version is closely related to the deep variational information bottleneck (VIB), which minimizes KL divergence between the (stochastic) representation $\mathbf{z}$ and a prior distribution, typically chosen to be a standard Gaussian. Roughly, the values in $ig(\mathbf{x})$ can be seen as specifying how many steps of forward diffusion to run in a Gaussian diffusion process initiated at $\mathbf{z}$, where values near zero correspond to running more steps of diffusion and thus sampling from a distribution which is KL closer to a standard Gaussian than the original input. Thus, like the VIB objective, we aim at minimizing $I(\mathbf{x}, \mathbf{z})$ while maximizing the task performance of $\mathbf{z}$. We can similarly consider other variations of InfoGating where arbitrary edges in a compute graph are gated[*].

Adversarial InfoGating. Note that the above versions of InfoGating are cooperative objectives, where the gating network and the regular encoder work together to lower the overall loss. This in turn leads to finding representations that capture the minimal sufficient information for a given task. We can also turn this formulation on its head, giving rise to an adversarial objective. In this case, the gating network is tasked with outputting masks that, when used by the

[*]Constructing differentiable upper bounds on compute cost is one potential application of this idea, similar to DARTS (Liu et al., 2019).

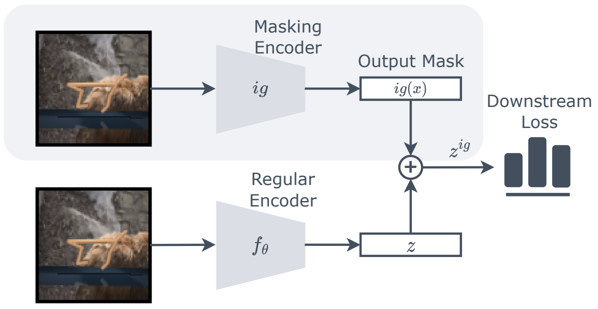


Figure 2: Feature InfoGating. performs gating in the representation space. Both the regular encoder and the gating encoder are trained using the downstream loss. The gating encoder is further encouraged to mask as much of the representation **z** as possible through an L1 penalty on the mask (not shown in figure).

regular encoder, lead to *maximizing* its loss. Instead of encouraging the masks to erase as much of the input as possible, the adversarial objective encourages the masks to remove as little information as possible while minimizing the regular encoder's performance on the downstream task when using the masked inputs. We can write the adversarial objective for $ig(x)$ as follows:

$$\mathcal{L}_{ig} = -\mathcal{L}_{\text{task}}\left(f(\mathbf{x}^{ig})\right) - \lambda \, \|ig(\mathbf{x})\|_1, \quad (5)$$

while the regular encoder $f(x)$ is optimized as before:

$$\mathcal{L}_f = \mathcal{L}_{\text{task}}\left(f(\mathbf{x}^{ig})\right) \quad (6)$$

This gives rise to a min-max objective w.r.t. the regular encoder and masking encoder. Note that the cooperative InfoGating version corresponds to a min-min objective instead. We describe a similar working of the adversarial objective with the multi-step inverse dynamics downstream loss in Algorithm 2.

5. Experiment Setup

We test InfoGating with multiple different downstream objectives, namely with: **1)** contrastive dynamics models (both inverse and forward models), **2)** self-supervised models like SimSiam (Chen and He, 2021), **3)** Q-learning (particularly TD-based critic updates), and **4)** behavior cloning. Since the multi-step inverse model produces the best results, we use the same for testing the feature InfoGating and adversarial InfoGating versions as well. We test **1)** and **3)** on the offline visual D4RL domain (Lu et al., 2022), **2)** on STL-10 (qualitatively) and corrupted CIFAR-10 (quantitatively), and **4)** on Kitchen (Gupta et al.,

Algorithm 2 Adversarial InfoGating Pseudocode

```
# f: regular encoder
# ig: gating encoder

for x, g, a in loader: # load a minibatch x with n
    samples
   x, g = aug(x), aug(g) # random crop aug

   ig_x, ig_g = ig(x), ig(g) # get info-gates

   x_ig = ig_x * x + (1 - ig_x) * nz_x # info-gate
        current state
   g_ig = ig_g * g + (1 - ig_g) * nz_g # info-gate
        goal state

   zx, zg = f(x_ig), f(g_ig) # encodings

   L = - InfoNCE(zx, zg, a) - |ig_x| - |ig_g| #
       masking loss
   L.backward() # back-propagate
   update(ig) # SGD update gating encoder

   L = InfoNCE(zx, zg, a) # encoder loss
   L.backward() # back-propagate
   update(f) # SGD update regular encoder
```

2019) manipulation domains. We now describe the workings of each InfoGating version and the corresponding experimental results. All hyper-parameter details are listed in the Appendix.

Inverse Dynamics Models. We use multi-step inverse models as our primary downstream loss due to its ability to recover the latent state effectively (Lamb et al., 2022; Islam et al., 2022a). Consider the contrastive loss based on InfoNCE as described in Eq. 1. To apply InfoGating, we mask both the current observation $\mathbf{x}_t$ and the goal observation $\mathbf{x}_{t+k}$ using the $ig(\mathbf{x}_t)$ and $ig(\mathbf{x}_{t+k})$ info-gates respectively. We then process both info-gated observations through the regular encoder to compute the corresponding representations $\mathbf{z}_t^{ig} = f(\mathbf{x}_t^{ig})$ and $\mathbf{z}_{t+k}^{ig} = f(\mathbf{x}_{t+k}^{ig})$. These are then optimized just as in Eq. 3 as follows:

$$\mathcal{L} = \text{InfoNCE}((\mathbf{z}_t^{ig}, \mathbf{z}_{t+k}^{ig}), \mathbf{a}_t) + \lambda \left(\|ig(\mathbf{x}_t)\|_1 + \|ig(\mathbf{x}_{t+k})\|_1 \right) \tag{7}$$

Note that both current and future observation masks are penalized through the L1 term. We test the inverse model InfoGating version on datasets consisting of offline observation-action pairings. The datasets contain pixel-based observations with video distractors playing in the background (Lu et al., 2022). We first pretrain the representations with the InfoGating objective in Eq. 7 and then perform behavior cloning (BC) using a 2-layer MLP over the frozen representations. We can replace BC-based evaluation with, e.g., Q-Learning-based evaluation, which leads to similar relative scores. We report scores for BC-based evaluation for easier reproducibility and reduced dependence on hyperparameters.

Table 1: Multi-step Inverse Dynamics with InfoGating. Comparing performance drops in the presence of background distractors. We report rewards achieved by a policy produced by behavior cloning on top of pretrained representations in **cheetah-run**. The "inv" model is our baseline with pretraining via multi-step inverse dynamics. The "w/ rand" model adds random info gating during pretraining and the "w/ IG" model adds learned info gating during pretraining (this is our method). The easy/medium/hard settings correspond to different levels of background distractor noise.

case	easy	medium	hard	overall
		expert		
inv	42.6 ± 36.3	29.5 ± 31.9	2 ± 0.6	24.7
w/ rand	21.0 ± 15.6	7.2 ± 6.12	4.2 ± 0.7	10.8
w/ IG	**176.2** ± **9.1**	**97.0** ± **5.7**	**44.8** ± **18.4**	**106.0**
		medium		
inv	45.6 ± 23.2	2 ± 0.8	10.7 ± 6.0	19.4
w/ rand	42.0 ± 31.8	85.4 ± 13.2	5.3 ± 1.9	44.2
w/ IG	**133.0** ± **12.8**	92.0 ± 35.2	3.0 ± 1.43	**76.0**
		med-expert		
inv	25.5 ± 18.6	14.2 ± 11.2	3.4 ± 4.8	14.3
w/ rand	10.4 ± 9.9	24.8 ± 34.4	2.9 ± 1.2	12.7
w/ IG	**158.0** ± **21.8**	**89.2** ± **7.0**	**38.8** ± **37.5**	**95.3**

We work with three levels of varying distractor difficulties, ranging from *easy*, *medium*, and *hard*. These variations correspond to the amount of noise added to the observations. At evaluation time, a noise-free plain background is used, thus creating an out-of-distribution shift in the distractor. All comparisons use random cropping as a pre-processing step for the observations. We compare the inverse InfoGating version with two baselines, 1) without InfoGating, i.e. the standard inverse model formulation from Eq. 1, and 2) a random masking strategy that follows the same setup as the InfoGating version but instead uses random masks in place of learnable masks. See Table 1 for results.

We observe that InfoGating leads to considerable performance gains over the standard inverse model and the random masking inverse model. Although adding random masking helps, learnable masks lead to much better performance. Since random masking provides similar benefits to data augmentation, some performance improvement over the standard inverse model is expected. However, learning masks clearly does more than just data augmentation. As can be visualized in Figure 1, the InfoGating masks explicitly remove background noise from the input observa-

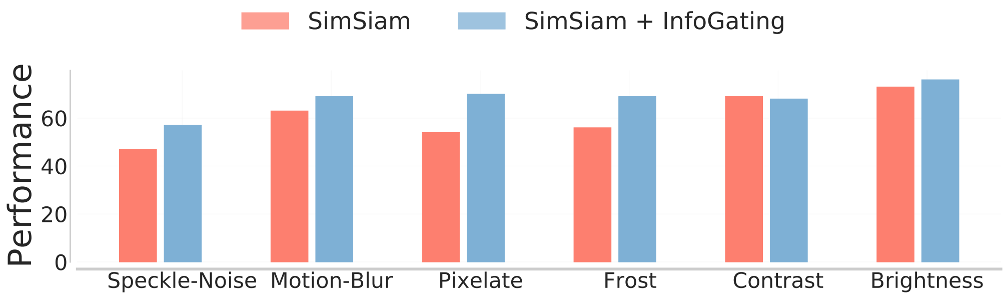


Figure 3: Generalizing to Corruptions through InfoGating. We directly compare SimSiam performance w/ and w/o InfoGating on different subsets of the corrupted CIFAR-10 dataset. Adding InfoGating improves robustness to test-time corruptions.

tions. When processing inputs info-gated by these masks, the encoder is unlikely to be distracted by any background noise. Overall, these results indicate that minimizing the downstream objective (in this case the inverse model) is not always sufficient for learning robust representations. By incorporating the InfoGating objective, we add an inductive bias which encourages focusing on the most useful information for the task without requiring additional supervision. Empirically, this inductive bias towards informational parsimony seems to produce more robust representations.

Forward Dynamics Models. Similar to the multi-step inverse dynamics model, we can use InfoGating on a multi-step forward dynamics model. The forward model is usually not enough to encode sufficient information for control and therefore lacks the guarantees that a multi-step inverse model has. We test this by info-gating both the current and future observation pair, just as in the inverse model. Next we process both of the masked observations through the regular encoder, and then concatenate the action $\mathbf{a}_t$ with the $\mathbf{z}_t^{ig}$ representation, before projecting it to the same dimension as the goal representation $\mathbf{z}_{t+k}^{ig}$. Finally, we penalize the info-gates for both current and goal observations as done previously to obtain the following objective:

$$\mathcal{L} = \text{InfoNCE}(\bar{\mathbf{z}}_t^{ig}, \mathbf{z}_{t+k}^{ig}) + \lambda \left(\|ig(\mathbf{x}_t)\|_1 + \|ig(\mathbf{x}_{t+k})\|_1 \right), \tag{8}$$

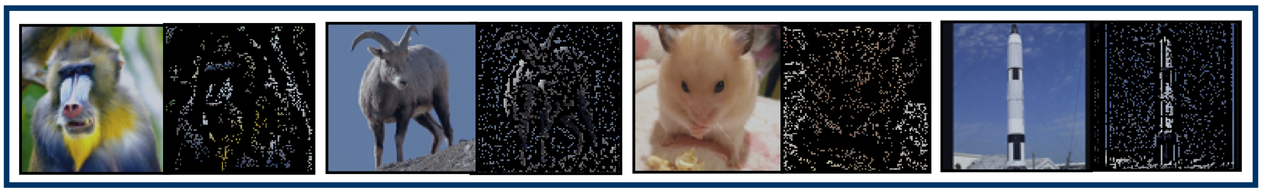
where $\bar{\mathbf{z}}_t^{ig} = g(\mathbf{z}_t^{ig}, \mathbf{a}_t)$ is a projection function that maps the state and action embedding to a space of the same size as $\mathbf{z}_t^{ig}$. Note that in the InfoNCE loss, the negatives now come from sampling different future observation representations from the batch.

We test the forward InfoGating model in the same distractor-based setup used for the inverse model. See Table 2 for results. We observe that the forward info-gates are indeed less performant than the inverse info-gates. In general, the forward objective can leave room for background information to leak into the representation. E.g., with structured or temporally correlated background noise the info gates may choose to look at the noise when predicting future states rather than at the agent pose. We see that the info-gates do not strictly recover the agent pose, thus hurting downstream performance. Nonetheless, InfoGating over the standard forward model does provide performance boosts.

Table 2: InfoGating with Forward Dyanmics Models. Experiment setup follows Table 1. "fwd" is the baseline forward dynamics model and "fwd + IG" adds InfoGating. For reference, training a forward dynamics model without distractors has a return of 173.6 ± 13.5.

case	easy	medium	hard	overall
fwd	6.0 ± 4.6	7.6 ± 5.9	4.2 ± 1.6	5.9
fwd + IG	**60.4** ± **27.9**	**62.1** ± **29.0**	**18.9** ± **5.2**	**47.1**

InfoGating for Generalizing to Test-Time Corruptions. Random masking has been used as an SSL objective to learn useful features for downstream object detection/classification. Can InfoGating be used as a general strategy to recover similarly useful features? We test this by applying InfoGating with self-supervised representation learning algorithms, in particular SimSiam (Chen and He, 2021). Specifically, given two augmented views of the input image $\mathbf{x}_1$ and $\mathbf{x}_2$, we info-gate both views to get $\mathbf{x}_1^{ig}$ and $\mathbf{x}_2^{ig}$, then process them through the regular encoder to compute the SimSiam loss $\mathcal{L}(\mathbf{z}_1^{ig}, \mathbf{z}_2^{ig})$. The SimSiam loss can be understood as being similar to the InfoNCE loss but without the need to sample negative pairs. It does away with this requirement by using simple heuristics such stop gradient and an additional predictor



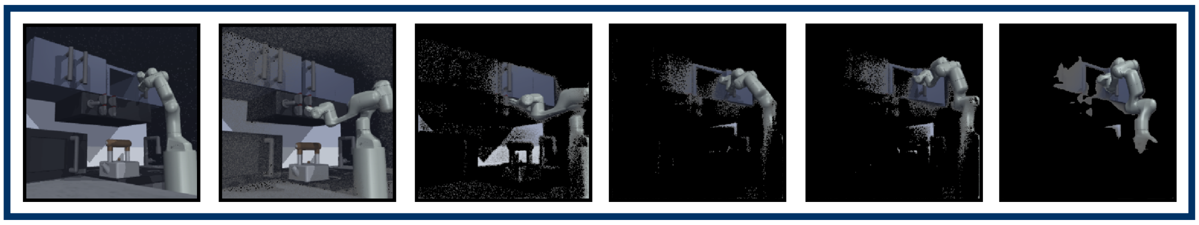


Figure 4: Visualizing Masks Learnt by InfoGating. **Top Row**: STL-10 images alongside the info-gated image. The masking network gradually learns to trace the edge boundaries for both foreground and background objects. **Bottom Row**: Kitchen-sdoor-open-v3 task and how the masks improve during training (left to right corresponds to increasing training steps). InfoGating is able to learn semantically meaningful masks even beyond the planar control domains such as the cheetah-run task discussed before.

network over the representation $\mathbf{z}$. More details about SimSiam can be found in the Appendix. Finally, we also add the original SimSiam loss, i.e. one over the unmasked inputs to the overall objective:

$$\mathcal{L} = \mathcal{L}_{\text{SimSiam}}(\mathbf{z}_1^{ig}, \mathbf{z}_2^{ig}) + \mathcal{L}_{\text{SimSiam}}(\mathbf{z}_1, \mathbf{z}_2) + \lambda \left(\|ig(\mathbf{x}_1)\|_1 + \|ig(\mathbf{x}_2)\|_1\right) \tag{9}$$

We test this version of InfoGating on CIFAR-10, while evaluating performance on Corrupted CIFAR-10 (Hendrycks and Dietterich, 2019). Figure 3 shows that InfoGating leads to improved evaluation scores in comparison to the base SimSiam method, thus showcasing better robustness to test-time induced corruptions.

InfoGating in the Feature Space. We compare the feature InfoGating version with the Variational Information Bottleneck (VIB), while using the same backbone encoders for both methods (see Table 3). Both VIB and feature InfoGating lead to some performance improvement over the base architecture, but do not come close to the score for the pixel InfoGating version. Interestingly, feature InfoGating almost matches the pixel InfoGating performance for the hard distractor case.

InfoGating for Stabilizing Q-Learning. In previous experiments, we have mainly dealt with pretraining representations and then learning regression-based policies or classification probes while keeping the representation fixed. In this section, we ask if the same conclusions hold if we train the representation end-to-end with a Q-Learning loss instead (see Table 4). Our InfoGating formulation for Q-Learning essentially amounts to the DrQ (Kostrikov et al., 2020) objective where the augmented observations are used to do Q-Learning but they are additionally masked by the info-gate masks. Note that the info-gate learning is driven by the Q-Learning loss directly as follows:

Table 3: InfoGating in the Feature Space. We compare the Variational Information Bottleneck with InfoGating in feature space. Experiment setup follows Table 1.

case	easy	medium	hard	overall
VIB	97.7 ± 32.2	86.2 ± 24.8	11.0 ± 5.2	64.9
feat. IG	71.4 ± 44.5	76.7 ± 21.3	**58.9** ± 28.3	**69.0**

$$\mathcal{L} = \left(Q(\mathbf{x}_t^{ig}, \mathbf{a}_t) - \mathbf{r}(\mathbf{x}_t, \mathbf{a}_t) - \gamma Q'(\mathbf{x}_{t+1}, \mathbf{a}_{t+1})\right)^2 + \lambda \|ig(\mathbf{x}_t)\|_1, \tag{10}$$

where Q and Q' denotes the current and target Q networks respectively, while $\mathbf{r}(\mathbf{x}_t, \mathbf{a}_t)$ is the obtained reward. We observe that the base DrQ algorithm is quite prone to failure for all three distractor settings, while with InfoGating we see significant improvements in performance.

Table 4: InfoGating for Q-Learning. We use DrQ as the base Q-Learning algorithm to test InfoGating. Experiment setup follows Table 1.

case	easy	medium	hard	overall
DrQ	5.7 ± 2.3	29.8 ± 20.5	3.4 ± 0.8	12.9
DrQ + IG	**63.4** ± **22.6**	**52.0** ± **6.2**	5.3 ± 1.9	**40.2**

Adversarial InfoGating. We now test the adversar-

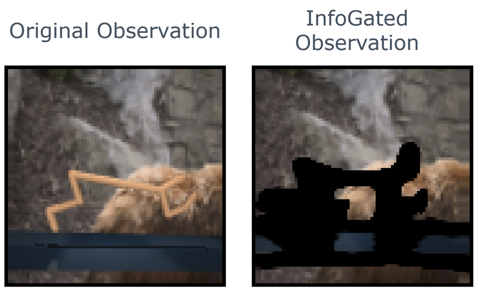


Figure 5: Left. Original distractor-based image. **Right**. Learnt adversarial mask over the original image.

ial InfoGating algorithm using a multi-step inverse model objective, just as we did for cooperative InfoGating. Figure 5 shows how the adversarial info-gates tend to hide the agent body. Some additional image content is also erased, since a precise silhouette of the agent's pose would be highly predictive of the agent's pose. To test how useful this kind of masking is, we simultaneously train a separate encoder over the reverse of the adversarial masks. If the adversarial process hides the robot pose successfully, then the reverse mask should contain maximal information about the agent and minimal information about the background. We indeed see that the adversarial InfoGating model performs similarly to the cooperative version, thus verifying that it is an equally viable approach for learning parsimonious representations (see Table 5).

Table 5: Adversarial InfoGating. We compare differences in the representations learnt with Cooperative vs Adversarial InfoGating. Experiment setup follows Table 1.

case	easy	medium	hard	overall
cop.	176.2 ± 9.1	97.0 ± 5.7	**44.8 ± 18.4**	**105.9**
adv.	**202.3 ± 2.64**	84.8 ± 41.2	4.6 ± 1.7	96.8

Remark. Note that even though some InfoGating results on the hard distractor setting seem to be completely failing in the results discussed thus far, interestingly the masks learnt for these domains are still fairly accurate in terms of removing most information except the agent. The performance drop in this case happens because of how the regular encoder processes the masked images. Particularly in the hard distractor case, there exist variations such as change in body color, camera zoom, etc. Such variations remain even in the info-gated images, thus hurting the performance. InfoGating at the feature-level can lead to robustness to such kind of noise, as is evident in Table 3. This motivates the idea of info-gating multiple layers at the same time, something which we leave for future work.

Finetuning General Representations. A lot of recent work has investigated learning general representations from large datasets involving diverse object interactions and different varieties of tasks. A natural question arises when using such pretrained representations for downstream tasks – how should the representation be fine-tuned so that only the relevant object and task features for the given task are used for learning the task's policy? InfoGating can be seen as a natural way to only extract information pertaining to the downstream task. Having tested InfoGating extensively on noisy environments, we now ask whether there are similar benefits when there is no explicit noise present in the environments, but there are multiple objects/pixel components which could act as potential distractors. We choose five different tasks from the Kitchen (Gupta et al., 2019) environment to test this hypothesis.

Table 6: Kitchen Results. We compare fine-tuning performance of BC w/ and w/o InfoGating. Representations are pre-trained either on ImageNet labels or through CLIP training. In both cases, InfoGating learns to remove irrelevant objects in the environment surroundings and leads to consistently higher scores.

	ImageNet		CLIP	
env	BC w/ IG	BC	BC w/ IG	BC
knob1-on	**17.6 ± 3.1**	7.3 ± 4.2	13.3 ± 1.49	11.3 ± 0.94
ldoor-open	9 ± 5.3	5 ± 1.9	11.0 ± 4.8	4.6 ± 1.49
light-on	19.6 ± 7.7	11 ± 3.6	24.0 ± 12.8	11.0 ± 3.0
sdoor-open	**61.6 ± 5.0**	36.6 ± 16.6	**61.6 ± 8.7**	42.0 ± 5.9
micro-open	13.0 ± 7.8	4.3 ± 2.9	12.0 ± 5.6	5.5 ± 2.1
overall	**24.1**	12.8	**24.3**	14.8

Specifically, given pretrained features, we test whether fine-tuning them using InfoGating is more powerful than fine-tuning with only a behavior cloning (BC) loss. We test two pretraining variations here, one corresponding to ImageNet features and the other corresponding to CLIP features (both are trained on ImageNet data, alongside language clippings for CLIP). Both are fine-tuned using a behavior cloning (BC) policy with a 2-layer MLP attached on top of the pre-trained encoding. Table 6 shows normalized success rates (each out of a maximum score of 100) for both BC and BC w/ InfoGating. We consistently see that InfoGating is able to mask out most pixel-level information except the robot gripper and the objects being manipulated (see Figure 4 for visualizations of the learnt masks), thus leading to strictly better success rates than the baseline BC policy.

Visualizing Masks Produced by InfoGating. We test the qualitative utility of masks learnt through InfoGating, particularly when moving beyond planar control domains like the MuJoCo-based cheetah, walker, etc. In particular, we test on the STL-10 vision dataset and on the Kitchen sliding door task (Figure 4). For STL-10, we learn masks using the same formulation as described earlier for CIFAR-10, i.e. adding InfoGating to the SimSiam objective. For the Kitchen door sliding task (sdoor-open-v3), we train masks in an end-to-end manner, using a behavior cloning loss, exactly as described in the previous section. Figure 4 shows different samples learnt through these two formulations. We clearly see that the masks are semantically meaningful in both cases, tracing object contours for STL-10, while focusing on the object of interest and the gripper in the Kitchen task.

6. Ablations

InfoGating Without Distractors. Learning infogates directly from the downstream loss allows for adapting the mask based on the task at hand. On the other hand, a fixed random masking scheme offers limited benefits, largely pertaining to enhanced data augmentation. For instance, using an InfoNCE loss derived from augmented views of the input with no distractors leads to learning accurate masks that capture the agent pose almost perfectly (see Figure 6). However, when distractor noise is added, the same InfoNCE loss leads to masks that are spread across the input, failing to remove background noise. Switching the loss to a multi-step inverse dynamics loss leads to masks that focus on the agent pose and successfully remove the background noise. Similarly, using the forward dynamics loss as the downstream objective, masks do not clearly follow the agent pose as in the inverse model case. These different variations highlight the importance of choosing the right downstream loss as well as *learning* InfoGating masks in conjunction with the given loss.

Effect of Mask Sparsity. We test what range of λ values leads to improved performance (Figure 7). As is expected, when λ is too high, the entire input is masked and no useful information is captured. Similarly, when λ is too low, none of the input is masked and we recover the performance of the base model, one that directly minimizes the downstream loss.

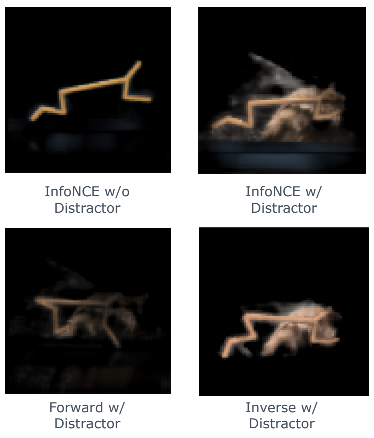


Figure 6: Effect of Losses on InfoGating Masks. InfoNCE w/o Distractor (mean score: **208.0**) learns almost perfect masks, while InfoNCE w/ Distractor (mean score: **10.4**) learns masks that are spread across the input, thus failing to learn a robust enough representation. Forward dynamics w/ Distractor (mean score: **60.4**) and Inverse dynamics w/ Distractor (mean score: **176.2**) lead to much more accurate masks and thus lead to better performing representations as well.

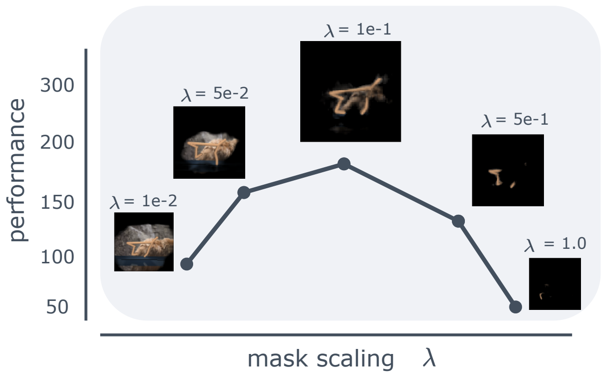


Figure 7: Scaling λ leads to different masks which reveal different amounts of information. For small λ, the masked observation still contains distracting/non-salient information, thus hurting performance. Similarly, when masking is too aggressive, too much information is lost and performance goes down.

Sharing Parameters for Masking and Regular Encoder. Instead of using two different encoders for the masking network and the encoding network, we ask if the same kind of masks can be recovered when sharing parameters between the two encoders (see Table 7). In such a case, the network first outputs the mask by processing the original image through its encoder, then masks the image, and then passes the masked image through its encoder again to generate an embedding vector. Such a shared parameter setup by default ensures that the encoder has a chance to see both masked and unmasked images, which could help avoid overfitting to the distribution of masks output by the mask encoder.

Table 7: Sharing Masking and Regular Encoder Parameters. Experiment setup follows Table 1. Default InfoGating version is marked in gray .

case	easy	medium	hard	overall
unshared	**176.2 ± 9.1**	**97.0 ± 5.7**	44.8 ± 18.4	**106.0**
shared	104.3 ± 22.6	44.2 ± 44.0	22.6 ± 6.3	57.0

We generally observe that the performance of InfoGating suffers when sharing parameters. We suspect this result may depend strongly on hyperparameters (e.g. model capacity, noise scale, etc.).

Mixing Original and Masked Inputs to Regular Encoder. Since the masking encoder and the regular encoder process different kinds of input (unmasked and masked input respectively), it is worth asking if the regular encoder can be regularized to a certain degree by enforcing it to be capable of processing both masked and unmasked images. We can also encourage the regular encoder to be robust to changes in the input/mask mapping learned by the masking encoder by occasionally swapping masks between images in a batch while training the regular encoder. We test both of these separately (Table 8) and observe that training the regular encoder on masked and unmasked inputs provides strong improvements over the base InfoGating model. This also has the added benefit of not using the masking network at evaluation time, and only using the regular encoder, since the regular encoder is implicitly trained to be invariant to the masked and unmasked inputs.

Table 8: Mixing Masking and Unmasked Inputs. Experiment setup follows Table 1. Default InfoGating version is marked in gray .

case	easy	medium	hard	overall
w/ mix input	176.2 ± 9.1	**97.0 ± 5.7**	**44.8 ± 18.4**	**106**
shuffle mask	170.0 ± 31.6	54.6 ± 22.3	9.8 ± 10.2	78.1
w/o mix input	119.4 ± 15.6	18.4 ± 17.5	10.5 ± 7.4	49.4

7. Limitations and Future Work

Although InfoGating helps learn robust representations, it does not recover the same performance for all distractor levels as in the case when no distractors are present. Ideally, we should be able to learn masks that fully remove the background information and thus recover the distractor-free performance. This might be down to two reasons - 1) The masks still leave room for noise information to escape around the edges of the object/robot they mask and 2) The regular encoder that processes the masked images should be robust to slight variations in the masking patterns between training and evaluation samples. In practice, we were able to recover better fine-grained masks using a UNet architecture which helped significantly with 1). We make some progress on 2) by training the regular encoder on a mix of masked and unmasked inputs. Nonetheless, there remains a clear gap in performance with the hard distractor setting even when the masks are accurate. This motivates the idea of efficiently InfoGating at multiple layers, i.e. without having separate masking networks for each info-gated layer.

Furthermore, our initial exploration of InfoGating can be viewed as a step towards learning object-centric representations without any explicit notion of objects forced into the network architecture or learning objective. E.g., in settings where an agent's actions may affect multiple objects, and we want to predict future observations conditioned on the agent's actions, the masks produced by InfoGating will need to reveal some information about (i.e. "look at") each object. The pursuit of informational parsimony should discourage the masks from revealing more of each object than necessary. Thus, the masking encoder will need to learn that objects exist and the general structure of the sorts of objects encountered by the agent so that the masks can efficiently reveal sufficient information about each object without revealing redundant information.

8. Conclusion

We present InfoGating, a general method to learn parsimonious representations that are robust to irrelevant features and noise. We describe two different versions: cooperative and adversarial, while demonstrating that the info-gates can be learnt both at the input pixel level or any intermediate feature level. We show how our approach can be combined with multiple downstream objectives including multi-step inverse dynamics models, Q-Learning, standard self-supervised tasks, and behavior cloning objectives. InfoGating leads to semantically meaningful masks that improve interpretability, and leads to consistently better performing representations, in terms of both out-of-distribution and in-distribution generalization.

9. Acknowledgments

The authors would like to thank Charlie Snell, Colin Li, and Dibya Ghosh for providing feedback on an earlier draft, and Katie Kang for suggesting the paper title.

References

Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2897–2905, 2018.

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016.

Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Florian Bordes, Pascal Vincent, Armand Joulin, Mike Rabbat, and Nicolas Ballas. Masked siamese networks for label-efficient learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXI*, pages 456–473. Springer, 2022.

Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *Advances in neural information processing systems*, 32, 2019.

Chenjia Bai, Lingxiao Wang, Lei Han, Animesh Garg, Jianye Hao, Peng Liu, and Zhaoran Wang. Dynamic bottleneck for robust self-supervised exploration. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 17007–17020, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15750–15758, 2021.

Thomas G. Dietterich, George Trimponias, and Zhitang Chen. Discovering and removing exogenous state variables and rewards for reinforcement learning. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1261–1269. PMLR, 2018. URL http://proceedings.mlr.press/v80/dietterich18a.html.

Yonathan Efroni, Dipendra Misra, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Provable rl with exogenous distractors via multistep inverse dynamics. *arXiv preprint arXiv:2110.08847*, 2021.

Norm Ferns, Prakash Panangaden, and Doina Precup. Bisimulation metrics for continuous markov decision processes. *SIAM Journal on Computing*, 40 (6):1662–1714, 2011.

Chelsea Finn, Ian J. Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 64–72, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/d9d4f495e875a2e075a1a4a6e1b9770f-Abstract.html.

Xiang Fu, Ge Yang, Pulkit Agrawal, and Tommi Jaakkola. Learning task informed abstractions. In *International Conference on Machine Learning*, pages 3480–3491. PMLR, 2021.

Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Riashat Islam, Manan Tomar, Alex Lamb, Yonathan Efroni, Hongyu Zang, Aniket Didolkar, Dipendra Misra, Xin Li, Harm van Seijen, Remi Tachet des Combes, et al. Agent-controller representations: Principled offline rl with rich exogenous information. *arXiv preprint arXiv:2211.00164*, 2022a.

Riashat Islam, Hongyu Zang, Manan Tomar, Aniket Didolkar, Md Mofijul Islam, Samin Yeasar Arnob, Tariq Iqbal, Xin Li, Anirudh Goyal, Nicolas Heess, and Alex Lamb. Representation learning in deep RL via discrete information bottleneck. *CoRR*, abs/2212.13835, 2022b. doi: 10.48550/arXiv.2212.13835. URL https://doi.org/10.48550/arXiv.2212.13835.

Hyoungseok Kim, Jaekyeom Kim, Yeonwoo Jeong, Sergey Levine, and Hyun Oh Song. Emi: Exploration with mutual information. In *International Conference on Machine Learning (ICML)*, 2019.

Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. *Advances in neural information processing systems*, 28, 2015.

Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

Alex Lamb, Riashat Islam, Yonathan Efroni, Aniket Didolkar, Dipendra Misra, Dylan Foster, Lekan Molu, Rajan Chari, Akshay Krishnamurthy, and John Langford. Guaranteed discovery of controllable latent states with multi-step inverse models. *arXiv preprint arXiv:2207.08229*, 2022.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.*, 17:39:1–39:40, 2016. URL http://jmlr.org/papers/v17/15-522.html.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: differentiable architecture search. In *International Conference on Learning Representations (ICLR)*, 2019.

Cong Lu, Philip J Ball, Tim GJ Rudner, Jack Parker-Holder, Michael A Osborne, and Yee Whye Teh. Challenges and opportunities in offline reinforcement learning from visual observations. *arXiv preprint arXiv:2206.04779*, 2022.

Bogdan Mazoure, Remi Tachet des Combes, Thang Long Doan, Philip Bachman, and R Devon Hjelm. Deep reinforcement and infomax learning. *Advances in Neural Information Processing Systems*, 33:3686–3698, 2020.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. 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=HyztsoC5Y7.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

Yingjun Pei and Xinwen Hou. Learning representations in reinforcement learning: An information bottleneck approach. *CoRR*, abs/1911.05695, 2019.

Yuge Shi, N Siddharth, Philip Torr, and Adam R Kosiorek. Adversarial masking for self-supervised learning. In *International Conference on Machine Learning*, pages 20026–20040. PMLR, 2022.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Alex Tamkin, Mike Wu, and Noah Goodman. Viewmaker networks: Learning views for unsupervised representation learning. *arXiv preprint arXiv:2010.07432*, 2020.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pages 1–5. IEEE, 2015.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Manan Tomar, Utkarsh A Mishra, Amy Zhang, and Matthew E Taylor. Learning representations for pixel-based control: What matters and why? *arXiv preprint arXiv:2111.07775*, 2021.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

Tongzhou Wang, Simon S Du, Antonio Torralba, Phillip Isola, Amy Zhang, and Yuandong Tian. Denoised mdps: Learning world models better than the world itself. *arXiv preprint arXiv:2206.15477*, 2022.

Pei Yingjun, Hou Xinwen, Li Jian, and Lei Wang. Optimizing information bottleneck in reinforcement learning: A stein variational approach. 2021. URL https://openreview.net/forum?id=IKqCy8i1XL3.

A. InfoGating with SimSiam

The SimSiam (Chen and He, 2021) objective uses a cosine similarity loss between the representation of one view $\mathbf{z}_1$ and a predicted output of the representation of the second view $\mathbf{p} = p(\mathbf{z}_2)$, where the function p is called the predictor. This is in place of using the InfoNCE loss to define contrastive pairs for $\mathbf{z}_1$ and $\mathbf{z}_2$. Both methods lead to similar performing representations with the main difference being that SimSiam does not require negative examples. The overall SimSiam loss can be written as:

$$\mathcal{L} = \mathcal{D}(\mathbf{p}_1, \mathbf{z}_2) \,/\, 2 + \mathcal{D}(\mathbf{p}_2, \mathbf{z}_1) \,/\, 2,$$

where $\mathcal{D}$ denotes the cosine similarity function:

$$\mathcal{D}(\mathbf{p}, \mathbf{z}) = -\frac{\mathbf{p}}{\|\mathbf{p}\|_2} \cdot \frac{\mathbf{z}}{\|\mathbf{z}\|_2},$$

Note that $\mathcal{D}$ is not a symmetric quantity as it uses a stop gradient operation on its second argument $\mathbf{z}$.

Having described how SimSiam works, we can now go into how InfoGating is applied alongside it. Besides the unmasked input views $\mathbf{z}_1$ and $\mathbf{z}_2$ we consider info-gated versions of the pixel input as well, namely $\mathbf{x}_1^{ig}$ and $\mathbf{x}_2^{ig}$. We then compute the respective info-gated representations $\mathbf{z}_1^{ig}$ and $\mathbf{z}_2^{ig}$ and use the SimSiam cossine similarity loss to predict the unmasked representations from the masked representations. In particular, the complete objective looks as follows:

$$\begin{aligned} \mathcal{L} = \; & \mathcal{D}(\mathbf{p}_{ig1}, \mathbf{z}_1) \; + \; \mathcal{D}(\mathbf{p}_{ig1}, \mathbf{z}_2) && \text{(InfoMask Invariance)} \\ + \; & \mathcal{D}(\mathbf{p}_{ig2}, \mathbf{z}_2) \; + \; \mathcal{D}(\mathbf{p}_{ig2}, \mathbf{z}_1) && \text{(InfoMask Invariance)} \\ + \; & \mathcal{D}(\mathbf{p}_1, \mathbf{z}_2) \; + \; \mathcal{D}(\mathbf{p}_2, \mathbf{z}_1), && \text{(Data-Aug Invariance)} \end{aligned} \tag{11}$$

B. Contrastive Dynamics Models

Multi-step Inverse Dynamics. Throughout the paper, we use a contrastive variant of the multi-step inverse dynamics loss. This is implemented by first encoding the current and future/goal observations (which are info-gated) x_t^{ig} and x_{t+k}^{ig} into corresponding encodings z_t^{ig} and z_{t+k}^{ig}. We then concatenate the two encodings along with the action to form a single encoding triplet. This is then passed through a 2-layer MLP to output logits for the energy value corresponding to the given $(z_t^{ig}, z_{t+k}^{ig}, a_t)$ triplet. The actions actually taken by the agent make up for a 'positive' triplet and the model is trained to output a low energy value. Similarly, a_t is replaced by random actions $\bar{a}_t$ to form a 'negative' triplet, for which the model is trained to output a high energy value.

Multi-step Forward Dynamics. For the forward contrastive loss, we simply encode the current action and observation pair $(\mathbf{x}_t^{ig}, \mathbf{a}_t)$ to $\bar{\mathbf{z}}_t^{ig}$) while encoding the future/goal observation $\mathbf{x}_t^{ig}$ to a similar size vector $\mathbf{z}_{t+k}^{ig}$. We then train these by applying standard InfoNCE over the two encodings.

C. Hyperparameter Details

We use a UNet architecture all throughout this paper for implementing InfoGating on the pixel-level. We use a warm-up consisting of 5k gradient steps where the InfoGating network is not trained, while the downstream encoder is. This is done so that the learnt masks are not affected by initial gradient errors in the downstream loss. Once the warm-up period is over, the loss in Equation 3 is deployed as usual. The details for the UNet architecture and the regular encoder are described in Table 9 and Table 10 respectively. Note that these network architectures correspond to the MuJoCo locomotion tasks. We add additional layers based on differences in input image sizes (84 x 84) for the Kitchen (256 x 256) and CIFAR/STL-10 experiments (32 x 32 / 96 x 96).

Table 9: UNet Architecture.

Down Sampling (InfoGating Encoder)
3 x 3 conv2d 32, stride 1, pad 1, GroupNorm, ReLU
3 x 3 conv2d 32, stride 1, pad 1, GroupNorm, ReLU
3 x 3 conv2d 64, stride 1, pad 1, GroupNorm, ReLU
3 x 3 conv2d 64, stride 1, pad 1, GroupNorm, ReLU
3 x 3 conv2d 64, stride 1, pad 1, GroupNorm, ReLU
MLP
FC 128, ReLU
FC 128, ReLU
FC 1600, ReLU
Up Sampling (InfoGating Decoder)
3 x 3 conv2d 64, stride 1, pad 1, GroupNorm, ReLU
3 x 3 conv2d 64, stride 1, pad 1, GroupNorm, ReLU
3 x 3 conv2d 32, stride 1, pad 1, GroupNorm, ReLU
3 x 3 conv2d 32, stride 1, pad 1, GroupNorm, ReLU
3 x 3 conv2d 32, stride 1, pad 1, GroupNorm, ReLU

Table 10: Regular Encoder Architecture.

6 x 6 conv2d 128, stride 6, pad 0, ReLU
1 x 1 conv2d 128, stride 1, pad 0, ReLU
3 x 3 conv2d 128, stride 1, pad 0, ReLU
4 x 4 conv2d 256, stride 2, pad 0, ReLU
FC 256, LayerNorm, ReLU

Table 11: MuJoCo Locomotion Training Details.

batch size	128
λ	0.1
IG warm-up	5k steps
cropping padding	4
frame_stack	3
action_repeat	2
buffer_size	100000
learning_rate	1e-4
eval_episodes	10

(a) Kitchen Training Details.

batch size	32
λ	schedule(0.1, 3.0, 2k steps)
IG warm-up	1k steps
cropping padding	none
frame_stack	1
num_demos	5 trajectories
learning_rate	1e-3
IG learning_rate	1e-4
action_repeat	1
eval_trajectories	50

(b) CIFAR/STL-10 Training Details.

batch size	32
momentum	0.9
weight_decay	1e-6
λ	0.06
IG warm-up	100 steps
cropping scale	(0.2, 1.0)
learning_rate	0.3
IG learning_rate	0.3
num_epochs	60