TransMASK: Masked State Representation through Learned Transformation

Sagar Parekh, Preston Culbertson, and Dylan P. Losey

Abstract Humans train robots to complete tasks in one environment, and expect robots to perform those same tasks in new environments. As humans, we know which aspects of the environment (i.e., the state) are relevant to the task. But there are also things that do not matter; e.g., the color of the table or the presence of clutter in the background. Ideally, the robot's policy learns to ignore these irrelevant state components. Achieving this invariance improves generalization: the robot knows not to factor irrelevant variables into its control decisions, making the policy more robust to environment changes. In this paper we therefore propose a self-supervised method to learn a *mask* which, when multiplied by the observed state, *transforms* that state into a latent representation that is biased towards relevant elements. Our method — which we call *TransMASK* — can be combined with a variety of imitation learning frameworks (such as diffusion policies) without any additional labels or alterations to the loss function. To achieve this, we recognize that the learned policy updates to better match the human's true policy. This true policy only depends on the relevant parts of the state; hence, as the gradients pass back through the learned policy and our proposed mask, they increase the value for elements that cause the robot to better imitate the human. We can therefore train TransMASK at the same time as we learn the policy. By normalizing the magnitude of each row in TransMASK, we force the mask to align with the Jacobian of the expert policy: columns that correspond to relevant states have large magnitudes, while columns for irrelevant states approach zero magnitude. We compare our approach to other methods that extract relevant states for downstream imitation learning. Across experiments with visual and non-visual states, we see that TransMASK results in policies that are more robust to distribution shifts for irrelevant features. See our project website: https://collab.me.vt.edu/TransMASK/

Keywords Imitation Learning, State Representation, Representation Learning, Domain Generalization

S. Parekh
Mechanical Engineering Department, Virginia Tech
E-mail: sagarp@vt.edu

P. Culbertson
Computer Science Department, Cornell University
E-mail: pculbertson@cornell.edu

D. Losey
Mechanical Engineering Department, Virginia Tech
E-mail: losey@vt.edu

1 Introduction

Imitation learning enables robots to learn tasks from offline demonstrations provided by a human expert. Consider teaching a robot to pick up a block and place it at the center of the table (see Figure 1). The human teacher guides the robot through successive states such as reaching the block, grasping it, and placing it at the desired location. In the demonstration, the robot receives observations of the scene recorded through cameras and the actions applied by the expert. When demonstrating the task, the human expert focuses only features crucial for completing the task — the object, the goal, and the robot positions. However, the robot's observations record information about the entire scene, capturing lighting conditions, table texture, background objects, and other task-irrelevant features.

A policy trained using these observations inherits this issue: it can attend to all parts of the state, including the details a human demonstrator would ignore. This causes the policy to become increasingly brittle. Consider Figure 1: when deploying a policy trained with demonstrations over a wood table for an identical task

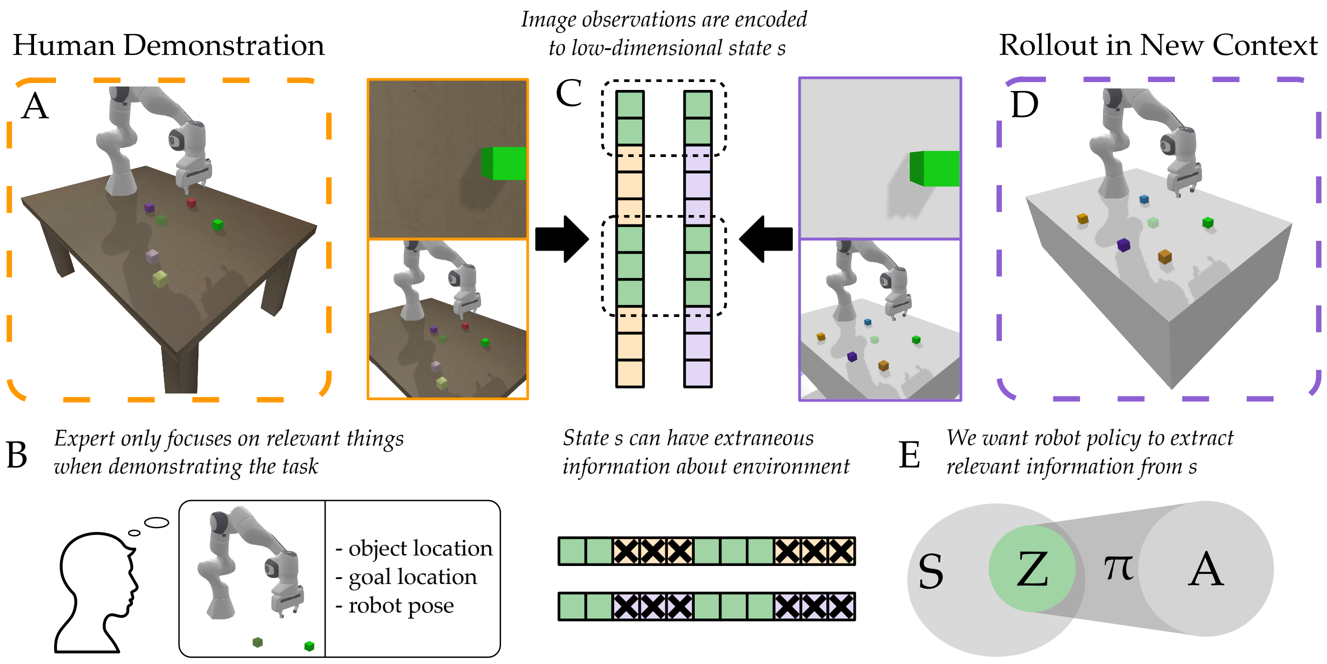
Human Demonstration
Image observations are encoded to low-dimensional state s
Rollout in New Context
A
C
D
B
Expert only focuses on relevant things when demonstrating the task
State s can have extraneous information about environment
E
We want robot policy to extract relevant information from s
- object location
- goal location
- robot pose
S
Z
π
A

Fig. 1 Robot learning to pick up a green block from a cluttered environment and place it at the center of the table. (A) The expert demonstrations are collected over a wooden table. (B) This expert's decisions depend only on features intrinsic to the task (e.g., green block, robot position, and target position). The robot's observations in these demonstrations, however, record information about the entire scene, capturing the table texture, background color, and task-irrelevant objects. (C) We assume a disentangled state s is extracted from these high-dimensional visual observations. In this state, some elements correspond to the task structure — positions of the green block, target, and robot, while others correspond to scene-specific factors. In the figure, relevant elements are shown in green and scene-specific elements are shown in orange and purple. (D) Standard imitation learning policy attends to the entire state. When this policy is deployed for the identical task in a different scene (e.g., over a marble table), it may fail due to spurious dependencies learned during training. (E) To learn a robust policy, we must encode the state to a representation z which masks-out task-irrelevant features. A policy that is conditioned on z is more robust to distribution shift caused by changes in the scene that do not alter the task structure, for instance, when the task is still to pick and place a green block but over a marble table instead of a wooden table.

on a marble table, the visual observations become out-of-distribution, causing the policy to fail.

Our key idea is that a robust policy is one that attends to the same features a human expert would. Thus, we aim to design policies with structures to mask out task-irrelevant information, enabling the policy to make decisions using features inherent to the task (e.g., object position, end-effector pose) instead of the scene(s) in its training data.

Existing works have developed a variety of methods for extracting this relevant state. A straightforward approach augments the training data with random transformations of visual observations (e.g., random cropping, adding noise to the image) [1, 2]. Nevertheless, introducing variability in the training data can degrade in-domain performance and does not guarantee robustness to large distribution shifts. Another line of work leverages vision-language model (VLM) backbones [3,4] that have robust state representations due to large-scale pretraining on diverse datasets. Despite these advantages, VLMs typically require task-specific finetuning, which can lead to catastrophic forgetting of pretrained features aversely effecting the robustness. Alternatively, a different class of literature focuses on learning minimal state representations that can extract only the most relevant information from the observations. This is achieved by introducing a trade-off between i) disconnecting the latent state from the actual state (compression) and ii) ensuring the latent state can map to correct actions (performance). Examples of this approach include using variational autoencoders (VAEs) for encoding the images to states [5, 6], training an encoder with the Information-Bottleneck (IB) principle [7], or adding structure to the representation space using contrastive learning [8,9]. However, these approaches — as we will show empirically and theoretically — rely on an ill-posed optimization problem that often collapses into inaccurate state representations that lack either compression or performance.

We therefore propose a new method for extracting the correct state from observations without modifying the optimization objective. We begin with the intuition that actions demonstrated by the expert are conditioned only on parts of the state that are relevant to the task. When optimizing a standard imitation learning objective, predicted actions will correlate strongly with these elements and weakly correlate with extraneous elements. Consequently, the magnitude of gradients associated with action-relevant elements will be larger. Building on this intuition, we hypothesize that:

Magnitude of policy Jacobian can be a proxy for causal relevance.

Put simply, we can exploit the gradients to identify and preserve the components of state that matter for control. This prevents the robot policy from learning spurious correlations between states and actions. In order to ensure that the state space is structured, we assume that the robot state is disentangled and can be decomposed into relevant and irrelevant elements. For example, the first k elements of the state could represent the location of the block and the robot[1]. Despite the additional structure imposed on the state space, the individual states can still contain environmental noise which can lead to causal confusion. In our running example, if the environment has other blocks in addition to the green one, the policy can incorrectly map the position of these “distractors” to the action. Consequently, a change in the position of any of these extra blocks can cause the state to go out-of-distribution even when the green block is not moved.

Applying this insight we derive TransMASK, a method that can extract relevant information from the state without additional supervision. TransMASK relies on the underlying optimization scheme of imitation learning and utilizes its gradients to learn a transformation matrix. This matrix transforms the robot's state into a representation which preserves elements important for action prediction while suppressing the rest. Our approach is a step towards robots that learn robust policies from human demonstrations, and can apply their policies to new testing environments.

Overall, we make the following contributions:

Identifying Why Existing Approaches Fail. We show how — in imitation learning settings — current approaches for state representations are ill-posed. i) These methods rely on a non-convex optimization that often gets stuck in local minima, ii) even when solved, the latent state can collapse to be just an action representation, iii) there are hyperparameters which are non-trivial to tune for achieving efficient trade-offs.

[1] We clarify that obtaining such disentangled features for our method does not require additional domain knowledge. As we will show later in our experiments, segmentation masks for all objects present in the scene can satisfy this assumption without the need for additional information from the designer, since the object labels can be simply obtained from a pretrained vision language model without needing finetuning.

Deriving TransMASK. We specifically focus on settings where the state is disentangled into relevant and irrelevant elements. Within this setting we develop a state representation method that naturally learns a representation z of the relevant state while purely optimizing for performance. Our approach is linked to attention networks, and we harness the properties of the policy Jacobian to reach our proposed approach.

Comparing to Baselines. We compare our approach to state-of-the-art representation baselines across a variety of simulated and real-world environments with visual as well as privileged state information. TransMASK achieves up to 15% higher performance than the next best baseline when tested in settings without distribution shift and approximately 9% higher success under environmental perturbations.

2 Related Works

Robust Imitation Learning. Imitation learning has advanced rapidly in recent years, driven by the development of expressive architectures such as diffusion policies [10] and transformer-based policies [11]. Despite these successes, deploying imitation learning policies in scenarios that differ from the training distribution (e.g., different object locations, clutter, or visual features) remains an open challenge [12, 13]. Several works have attempted to address this challenge and make policies robust to distribution shift, out-of-distribution observations, or noisy observations. Broadly, these works focus on two distinct approaches to achieve robustness: one set of methods focus on the *training data* while another focuses on the *training procedure.*

Data curation and diversification: The first class of methods focus on structuring the demonstration dataset so that the learned policy attends to the most relevant aspects of the state space. A straightforward strategy is to collect datasets while ensuring data diversity to improve robustness [14, 15]. However, in many practical settings, we may have limited control over data collection, since it is common to use readily available datasets such as [16, 17]. Therefore, several approaches aim to re-balance the given dataset to learn diverse behaviors reliably [18, 19]. Other works improve generalization through domain randomization [1, 2], exposing the policy to variations in the observations during training. Alternatively, we can augment the available dataset with auxiliary data to improve robustness of the trained policy. For example, [20, 21] leverage auxiliary datasets alongside optimal data to improve performance, [22, 23] learn a dynamics model from expert data and use it to generate additional synthetic demonstrations, and [24] uses NERF [25] to generate high-fidelity observations from noisy states to augment the expert dataset.

Adversarial BC: The second class of methods modifies the training pipeline to enhance robustness. This is typically achieved through adversarial training which improves model robustness by minimizing the prediction loss under worst-case perturbations of the input. In the context of imitation learning this is formulated as a generative adversarial game in which the policy is trained jointly with a discriminator that distinguishes between expert trajectories and those generated by the policy. The most prominent works in this domain include GAIL [26] and f-GAIL [27], while other works have extended this idea to achieve better sample efficiency and performance [28–31]. Adversarial training has also been extended to diffusion policies to further improve generalization [32].

While these approaches in imitation learning have been shown to improve performance and robustness, they do not address the more fundamental problem: we should condition the policy only on the relevant parts of the state. Data curation approaches can have computational overheads related to generating additional data or finding the right balance for the demonstration's prior to training the policy. Some of these methods rely on a human expert for labeling the dataset which can be time consuming. Adversarial training can often introduce instability in optimization, requiring precisely tuning the discriminator, and is often unsuitable for high-dimensional inputs such as visual observations. Additionally, variants of GAIL are sample-inefficient as they handle distributional shift by interacting with the environment.

State Representations. There are various works that explicitly focus on extracting relevant information about the task to achieve robust performance by encoding the observations to a structured representation space. Recent advances in object-centric representations, such as VIMA [33] and Transporter Nets [34], decompose the scene into object tokens using segmentation. While this has shown to improve spatial reasoning, they attend to all discovered entities equally. A more general approach requires compressing the input by retaining only critical information from the state and discarding redundant or unnecessary features. Contrastive learning, which optimizes the latent representation space by minimizing the distance between semantically similar samples while maximizing the distance between dissimilar samples, has emerged as a viable solution to this problem. In imitation learning, contrastive loss is leveraged to learn representations that are robust to variations in visual observations [8, 9, 35]. While [35] uses auxiliary static observations of the robot alongside demonstrations and identify positive and negative pairs using temporal proximity of the observations, [8] uses action sequence similarity to identify positive and negative pairs.

However, a contrastive approach typically requires comparing each representation with many negative examples to prevent representation collapse. An alternative is to use the Bootstrap Your Own Latent (BYOL) objective [36] to learn invariant state representation from image observations as in [37]. A simpler approach is to leverage self-supervised training using variational autoencoder (VAE) for extracting representations. For example, [38, 39] use VAE to extract motion primitives from robot trajectories; and [5, 6, 40] use VAE to extract better state representations from image observations.

Learning state representation has also been viewed from an information theoretic perspective, where mutual information can eliminate redundancy in the learned representations. Specifically, research has examined the use of the Information Bottleneck (IB) principle [7] for deep representation learning, demonstrating that IB can make latent representations invariant to irrelevant parts of the inputs [41, 42]. This principle has also been leveraged for learning task-relevant state representations for robust imitation learning [43].

Despite their success, there are limitations to these approaches. First, contrastive learning requires identifying positive and negative samples in the demonstrations which requires additional domain knowledge. Training can also be unstable in contrastive learning resulting in representation collapse. Further, approaches like VAEs can encode high-dimensional visual observations to a structured representation space, but they do not explicitly aim to retain only task-relevant information. Finally, while the IB seeks to compress the state, in the context of imitation learning, it often lacks an explicit supervision signal to distinguish between nuisance factors and features important for task completion. Later in the text, we will provide a detailed discussion on why the IB objective is ill-posed for imitation learning applications. In contrast, TransMASK extracts task-relevant features as a byproduct of the imitation learning gradient flow, alleviating the need for additional data or modifying the training pipeline and the objective function for optimization.

3 Problem Statement

We consider offline imitation learning settings. Within these settings, the robot is given a fixed dataset of demonstrations. In each demonstration a human expert teleoperates the robot through instances of the desired task. Based on these examples the robot learns a control policy that can robustly perform the demonstrated task across a variety of initial states. In particular, the learned control policy should be robust to changes in the scene that do not alter the task (e.g., if the color of a table changes, or if clutter is added to the background). We specifically focus on settings where the state is *disentangled.* There are some elements of the state that are relevant to the expert's control decisions, while other state elements are irrelevant noise that do not impact how the expert behaves. To separate the relevant and irrelevant state elements, we introduce a latent state representation $z \in \mathbb{R}^d$.

Dataset. Each demonstration consists of a sequence of tuples $\xi = \{(x^1, I^1, a^1), \cdots, (x^t, I^t, a^t)\}$ where $x \in \mathbb{R}^m$ is the proprioceptive state (e.g., joint angles), $a \in \mathbb{R}^m$ is the applied action (e.g., joint velocities), and I is the robot's observation (e.g., images captured from onboard and/or static camera(s)). Overall, the dataset includes a few demonstrations for the desired task. In our settings, we consider state $s \in \mathbb{R}^n$ which includes the proprioceptive states x as well as features ϕ encoded from image observations I, i.e., $s = [x, \Phi(I)]$ using visual encoder Φ. For every state in the dataset, the human expert labels what action the robot should take in order to complete their desired task.

Human. The expert's policy π^*, which they use to provide demonstrations, only considers task-relevant features and is not influenced by scene-specific components. For example, when demonstrating how to pick up a block, the human only focuses on the block's location and the robot's location relative to the block. Their decisions are not influenced by texture of the table, lighting conditions, or the presence of clutter[2]. However, the state s that the robot records when the human is demonstrating the task includes all the information from the scene.

Disentangled State. In our problem setting, we assume that the state vector s is disentangled and can be separated into two components:

$$s = [s_1, \ldots, s_n]^T, \quad s_i \in \mu \text{ or } s_i \in \eta \tag{1}$$

Here $\mu \in \mathbb{R}^k$ are the *relevant* elements of the state that the human expert uses to make their control decisions. Returning to our running example, μ could contain the location of the block. The remaining state elements $\eta \in \mathbb{R}^{n-k}$ are *irrelevant* to the expert's control decisions, and can be treated as extraneous information. For example, perhaps η includes the color of the table or the location of background objects.

We emphasize that the *disentanglement* of state s is fundamental to our proposed approach. We argue that this sort of separation is often inherent to the learning context (e.g., humans only focus on a few relevant features when making decisions [44]). However, we recognize that the state s that the robot extracts from the scene may not align with these disentangled representations, and we will test the limits of this assumption in our experiments.

Latent State. Since the state s can contain irrelevant information, we do not want our robot learner to make decisions based on every aspect of this state. To make the policy robust, it should focus only on task-specific information μ. However, we do not know which elements of the state constitute μ and which elements constitute η. Therefore, we seek to learn a state representation $z \in \mathbb{R}^n$ that only encodes the elements of s that strongly correlate with the expert's demonstrations. We instantiate this encoding as a *state mapping* $z = f(s)$. Ideally, we want to optimize the following objective:

$$f_\theta \in \arg\min_\theta \; \mathbb{E}_{s \in \mathcal{D}} \Big[H(\mu \mid f_\theta(s)) - H(\eta \mid f_\theta(s)) \Big] \tag{2}$$

Here $H(a \mid b)$ is the conditional entropy (i.e., the uncertainty) over variable a given variable b [45], and f is parameterized by weights θ. Intuitively, if μ and z are correlated, then $H(\mu \mid z) \to 0$ because we can determine exactly what μ is after $z = f(s)$ is measured. Conversely, if η and z are uncorrelated, then $H(\eta \mid z) \to H(\eta)$ because z does not provide any information about η. Overall, Equation (2) formalizes the type of state mapping f_θ we want to achieve — the state mapping should extract μ from s while disregarding η. However, we cannot directly implement Equation (2) because the robot does not know which state elements are relevant or irrelevant *a priori.* Our approach will aim to optimize this objective implicitly using the supervision available in the imitation learning loss.

Policy. Using the offline dataset $\mathcal{D}$ and the state mapping f_θ, the robot learns a control policy that automates the demonstrated task. Let $a = \pi^*(\mu)$ be the expert policy that outputs the correct action a for each state s. We seek to imitate this expert policy through a compositional approach: first, the state mapping $z = f_\theta(s)$ encodes the latent state, and second, our learned policy $a = \pi(z)$ maps that latent state into robot actions. We write our overall policy as $a = \pi_\psi(f_\theta(s))$ with model weights ψ and θ. Our objective is for this compositional policy to achieve two things. To learn per-

[2] provided that such clutter does not interfere with grasp feasibility. In our setting, clutter is treated as a potential source of spurious correlation or causal confusion, rather than an obstruction to manipulation.

formant behaviors, the actions $a = \pi(z)$ should match the expert's demonstrated actions across dataset $\mathcal{D}$. To increase robustness to environmental variations, the latent $z = f_\theta(s)$ should retain task-relevant information μ and discard extraneous elements η.

4 TransMASK

We want the robot to learn tasks from human demonstration and generalize to unseen contexts. As previously discussed, only the task relevant features are factored into the expert's decision making, and a robust robot policy should only be conditioned on these aspects of the state μ. To extract μ, we propose a method to learn state representations $z = f(s)$ that fulfill two desired characteristics — it must retain information from s that enables the policy to match the actions demonstrated by the human, while removing information η that does not relate to the task structure. In this section, we start by demonstrating how standard approaches for learning latent representations — such as autoencoders — are poorly designed for this objective (Section 4.1). Next, we analyze the Jacobian of the expert policy to determine the characteristics of an effective state representation (Section 4.2). Finally, we derive our proposed framework for learning z, and prove that this approach addresses the issues with standard representation techniques (Section 4.3).

4.1 State Representation and Collapse

In Equation (2) we present an objective that can be optimized to learn a state representation z which only extracts task-critical information from the state while excluding unimportant information. As mentioned earlier, we cannot directly optimize this objective because the relevant and irrelevant parts of the state are not known a priori. However, we know that while demonstrating the task, the human expert's actions are not influenced by some parts of the state. Taking this into consideration, we can reformulate the objective from an information theoretic perspective and use mutual information to quantify the relevance of the elements in the state. Specifically, we can rewrite the objective from Equation (2) in terms of state s, action a, and state representation z where we want to maximize $\mathcal{I}(f(s), a)$ and minimize $\mathcal{I}(s, f(s))$. Here, $\mathcal{I}(A, B)$ is the mutual information between A and B, measuring the amount of information gained about one random variable by knowing the other. The commonly used framework in previous literature for such an objective is the Information Bottleneck (IB) principle [7]:

$$\begin{aligned} L &= \lambda \mathcal{I}(s, z) - \mathcal{I}(z, a) \\ &= H(a \mid z) - \lambda H(s \mid z) \end{aligned} \tag{3}$$

We derive the second step by using the definition of mutual information $\mathcal{I}(A, B) = H(A) - H(A \mid B)$, and $z = f(s)$. Equivalently, the loss can be re-written as:

$$\max H(s \mid z) \quad \text{subject to} \quad D_{KL}(\pi^*, \pi) \leq \lambda \tag{4}$$

This objective can be broken down into two loss functions — a performance loss function that encourages the model output to match the expert dataset and a regularization loss that serves to disconnect z from s. Together these terms can ensure that the state representation only extracts the information from s which is necessary for accurately predicting the expert action. Different approaches utilize different regularization: for example, [43] estimates the mutual information $\mathcal{I}(z, s)$ to learn reliable robot policies, while others such as [46] use a VAE to constrain the representation space to be close to a unit Normal distribution. *However, this framework is ill-posed due to four fundamental issues.*

First, this type of objective with both a minimizing and a maximizing term is tricky to optimize due to the presence of multiple optima as well as the instability introduced by the conflicting terms. Second, these methods require the designer to choose λ "by hand" to balance the loss terms, and this heuristic tuning can be challenging. If we decrease λ too much, then we are overly focused on performance, and may learn a latent state z that matches the original state s, which is counter to our goal of extracting only relevant information from s. By contrast, if we increase λ too much, then we are overly focused on compression, and may learn a latent z that is completely separated from s — harming our ability to actually learn and perform the task. Third, estimating the mutual information between s and z can be intractable for most applications. Finally, when applied to imitation learning contexts, these structures allow z to collapse into an action representation rather than a state representation.

To demonstrate this collapse, consider a simple network architecture with an encoder that maps s to z and a policy which maps z to a. The architecture can be trained by optimizing the reconstruction loss on action a. This simple architecture is a straightforward, albeit naive, approach to implementing the optimization scheme presented above. In this simple case, z could simply be a sufficient encoding for reconstructing the action accurately, but it may not necessarily extract relevant information from s. Put simply, z learns a latent space that is an action representation and not a state representation. This issue arises because the optimization scheme of VAEs does not explicitly incentivize it to encode state information, rather it forces the decoder to predict the action given a latent representation. As a result, the latent space can collapse into a minimal sufficient statistic of s for predicting a. This may exclude semantic information from the visual observations, such as task object, goal location, and even the distractors.

One way to force z to be a representation of the state is to add a decoder that maps z back to the state s, i.e., a typical autoencoder. This structure does prevent the representation collapse described above. However, the latent space learned by this autoencoder embeds all the elements of s — relevant μ as well as irrelevant η. Indeed, if the irrelevant features fluctuate more across the dataset than the relevant features, then z is likely to emphasize these irrelevant features in order to accurately reconstruct s, again leading to failure.

4.2 Jacobian of the Expert Policy

Instead of using the ill-posed optimization of the IB principle for imitation learning, we intend to optimize for performance alone and utilize the underlying gradient structure to extract a compact representation of the state. We start by taking a closer look at the expert policy $\pi^*(s)$. Remember, in our problem settings we assume that the state $s = (\mu, \eta)$ is composed of relevant and irrelevant elements, and these elements are disentangled, i.e., μ and η are mutually exclusive and exhaustive ($\mu \cap \eta = \emptyset$, $\mu \cup \eta = s$). With this in mind, consider the Jacobian of the expert policy:

$$\nabla_s \pi^*(s) \tag{5}$$

This Jacobian is a $m \times n$ matrix, where m is the dimension of the action a and n is the dimension of the state s. The columns of this matrix indicate the impact of the corresponding element of the state on the robot's actions, or in other words, the sensitivity of the action to the corresponding element of state. For example, the first column indicates how the first element of s affects each dimension of a. Since the expert's decisions are not influenced by the irrelevant elements of s, the column i of the Jacobian corresponding to $s_i \in \eta$ will be all zeros. In contrast, the columns j of the Jacobian corresponding to $s_j \in \mu$ will have non-zero values since these elements affect the expert's actions. Overall, $\nabla_s \pi^*(s)$ will be a sparse matrix such that the states that contain purely irrelevant information are in the *null space* of the Jacobian. In the next subsection, we discuss how we can leverage this property of the Jacobian in learning a state representation that extracts only the relevant information from the state.

4.3 Deriving TransMASK

Following our discussion about the pitfalls of the IB objective and the Jacobian of the expert actions, we now discuss how we can learn a correct state representation. The fundamental problem with the IB objective arises due to conflicting objectives — performance and regularization. We aim to eliminate the regularization loss term from the objective. As the Jacobian of the action emerges naturally when optimizing the imitation learning objective (performance term), we can leverage the Jacobian to learn a state representation without the need for regularization. To this end, we define state representation z as an affine transformation of state s:

$$z = Ms \tag{6}$$

where M is a $n \times n$ square matrix. We will refer to M as a *mask*, since it can mask-out or remove elements of s from the resulting z. Indeed, each element of z here is a linear combination of the state elements, i.e., $z_i = M_{i1}s_1 + \ldots + M_{in}s_n$. Selecting $z = Ms$, where M is a constant matrix, ensures that the state representation maintains a correlation with the state and does not collapse into an action representation. This also eliminates the need for estimating the mutual information between s and z, which is often intractable. Further, as we will show later, the regularization loss emerges naturally as part of the training and the structure of the model. Now, we only need a performance loss to encourage the model to predict accurate actions given a state. The natural choice is the MSE loss which is commonly used in behavior cloning:

$$\mathcal{L}(\psi, M) = \sum_{(s,a)\in\mathcal{D}} \frac{1}{2}\|\pi_\psi(Ms) - a\|^2 \tag{7}$$

Since the matrix M is constant, we parameterize it with a learnable parameter θ. The state s, which can be privileged information (i.e., exact location of the objects) or disentangled features obtained from image observations, is transformed into a representation z according to Equation (6). As a result, our approach is versatile because the mask can be easily appended to any existing IL policy without augmenting the training process or the loss function. We show later in our experiments that this enables the network to learn a mask that only focuses on parts of the state that relate to the task structure μ while removing the unimportant elements η. Our method is summarized in Figure 2.

To demonstrate how the Jacobian can naturally learn such a mask, we take the gradient of our loss in Equa-

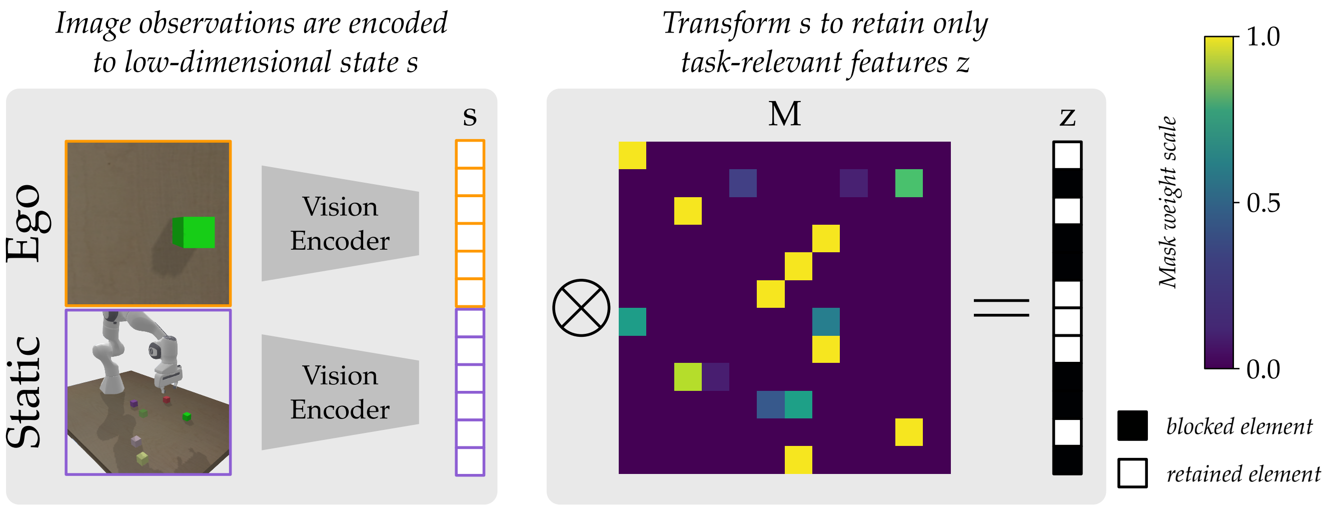
Image observations are encoded to low-dimensional state s
Transform s to retain only task-relevant features z
Ego
Static
Vision Encoder
Vision Encoder
s
M
z
Mask weight scale
1.0
0.5
0.0
blocked element
retained element

Fig. 2 Schematic diagram of TransMASK. The visual observations from the static and robot-mounted (ego) cameras are encoded to a disentangled state vector s. We introduce a mask encoder that outputs a constant mask M of shape $n \times n$, where n is the dimension of s. This mask is a sparse matrix in which the columns that correspond to task-irrelevant elements of s are close to zero. Therefore, when we compute the representation z by transforming the state with M, it only retains the elements critical for accurately predicting actions. For example, if element i of s is not important for action prediction, the i^{th} column of M contains all zeros. Consequently, the matrix product Ms eliminates s_i from z. The robot policy is conditioned on z rather than the entire state to achieve robustness.

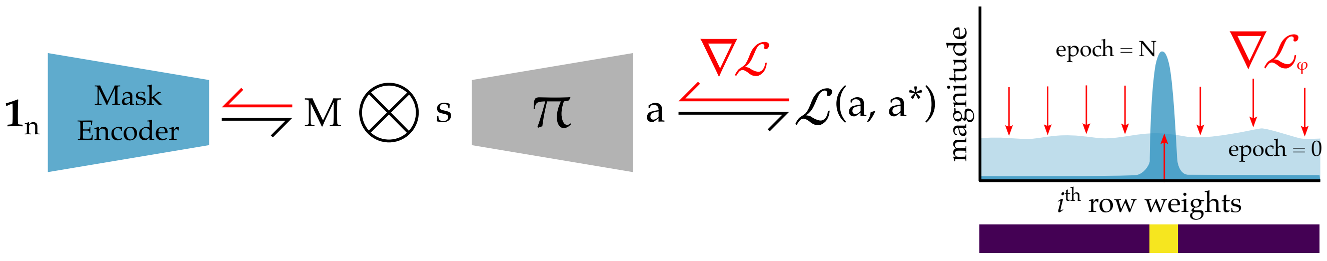

Fig. 3 Overview of how the mask is learned. (Left) The mask encoder is a small fully-connected neural network that inputs a 1-vector of size k ($\mathbf{1}_k = [1, 1, \cdots, 1] \in \mathbb{R}^k$) and outputs a matrix $M \in \mathbb{R}^{n\times n}$. This matrix is used to transform the state $s \in \mathbb{R}^n$ into a representation $z = Ms$. Since the expert's actions when providing demonstrations are only influenced by the task-relevant features in the environment, the columns of the Jacobian of the expert policy corresponding to the irrelevant elements in the state will be near-zero magnitude as discussed in Section 4.2. (Right) When training the robot policy, as the loss converges the Jacobian of the robot's learned policy changes the rows of M until the values start weighting the task-relevant elements more than the irrelevant ones.

tion (7) with respect to the mask weights θ:

$$\begin{aligned}\frac{\delta\mathcal{L}}{\delta\theta} &= \frac{(\pi(Ms)-a)^2}{\delta\theta} \\ &= \frac{\delta\mathcal{L}}{\delta\pi(Ms)}\cdot\frac{\delta\pi(Ms)}{\delta z}\cdot\frac{\delta z}{\delta M}\cdot\frac{\delta M}{\delta\theta} \\ &= (\pi(Ms)-a)^T\cdot J_\pi z\cdot s\cdot J_\theta M \end{aligned} \tag{8}$$

Here, we drop the subscripts denoting parameterization for simplicity. The dimension of the Jacobian depends on the architecture of the mask. If M is a learnable parameter matrix $\theta \in \mathbb{R}^{n\times n}$, the Jacobian simplifies to an $n \times n$ matrix. On the other hand, if M is the output of a neural network with parameters θ, the Jacobian becomes a matrix of shape $n \times n \times p$, where p is the input dimension of the network. In both cases, the Jacobian's structure dictates which elements of the state s influence the action. Since the Jacobian of the expert policy is a sparse matrix, we aim to learn π with the same sparsity. This ensures that the model remains invariant to environmental nuisances (η).

Intuitively, when the policy and mask are jointly trained to match the expert policy, the optimization scheme naturally separates the state elements. Task-relevant elements ($s_i \in \mu$) yield higher magnitude gradients, as they are critical for minimizing the error between predicted actions and expert actions. In contrast, unimportant elements ($s_j \in \eta$) yield lower magnitude gradients, as the expert actions are not influenced by these values. Consequently, the parameter θ updates to produce M that emphasizes the correct elements in s while suppressing the rest. To prevent the mask elements from becoming unbounded and disproportionately scaling s, we utilize a normalization layer such as softmax or sparsemax [47]. This normalization also encourages the mask to converge to a hard selection of features, by explicitly pushing values of some elements to be zero. The activation is applied to each row of M to ensure that weights for each state element are constrained and interpretable. We acknowledge that while this justification for mask convergence is heuristically sound and supported by empirical results discussed in the next section, a formal proof of convergence remains an open question.

4.4 Comparing with Attention Mechanism

Here we provide an alternative perspective on our proposed TransMASK framework. Our mask can be viewed as an attention mechanism [48] where we learn the attention weights for the elements of the input state; the state representation is the matrix product of the state with these learned attention weights. In attention mechanisms, the attention weights for the token pairs is defined by the formula: softmax $\left(\frac{QK^T}{\sqrt{d_k}}\right)V$. During training, there is no direct supervision for learning the correct attention weights. Rather, the back propagation of gradients from the loss push query Q and key K to produce a larger dot product for the important pairs and lower dot product for unimportant pairs. As a result, the networks that learn the key, query, and value are updated such that the model is able to discover the correct attention patterns that minimize the loss. Similar to this, in our method there is no direct supervision for training the mask. Rather, it is learned as a result of the gradients that encourage the mask elements to focus more on the parts of s which influence the output while ignoring the irrelevant parts.

Nevertheless, our method fundamentally differs from conventional attention mechanism in the way it weights the elements of the state. Attention mechanisms are designed to model the relative importance of different components of the input, whether these correspond to different elements in a sequence or spatial patches of images. In contrast, our task requires transforming a single disentangled state vector into a compressed representation that retains only the task-relevant elements. At first glance, attention might appear to be a natural solution. However, a standard attention mechanism computes dynamic attention weights that are conditioned on the input, i.e., the key, query, and value are computed via neural networks that take the state vector as input. While this allows the model to focus adaptively on different elements of the state, it also introduces the risk of losing the original disentanglement as task-irrelevant elements may be inadvertently weighted and propagated through the representation. In contrast, our method uses the policy Jacobian as supervision for feature selection. As a result, we learn a static mask that is input-invariant as it is independent of the state. This static mask reflects our underlying assumption about the state decomposition into μ and η. This design is particularly advantageous because, even when deploying a policy in a new environment, task relevance should be determined by task structure, rather than the variations in the input values. For instance, in a pick-and-place task, the end-effector and object poses are always relevant, while visual textures, or background color are generally not relevant. Thus, making our attention mask input-dependent, as in the conventional attention mechanism, would simply provide another path to allow task-irrelevant information to enter the policy.

To summarize, our method shares some similarity with the attention mechanism in the learning dynamics — the way attention mechanism is able to learn the attention weights is closely related to how our mask learns which elements of the state to attend to. However, it has a principal distinction from the attention mechanism. TransMASK learns a static, state independent mask leveraging the disentangled structure of the state, transforming it into a meaningful state representation for downstream policies.

5 Experiments

TransMASK learns a mask that attends only to the elements of the state that are relevant for performing the given task. In this section, our goal is to validate the proposed approach and test whether it can achieve robustness against data which is outside the training distribution. We conduct extensive experiments in simulation as well as real-world tasks. Additionally, we incorporate our method with two policy classes: MLP, which uses fully-connected architecture as the policy head; and Diffusion Policy (DP), which is a policy head trained with the diffusion objective [10].

Baselines. Across our experiments, we compare TransMASK with state-of-the-art state representation baselines. These baselines learn well-structured representations for the state either through pretrained visual encoders, contrastive learning, or by imposing a prior on the representation space. Below we discuss these baselines in more detail.

– **Behavior Cloning (BC).** [49] A standard imitation learning approach. BC learns to map the state to robot actions as demonstrated by the expert. This method does not explicitly encode the state to extract task-relevant information; rather, it attends to the entire state when predicting robot actions.

– **Variational Autoencoder (VAE).** [50] A self-supervised approach that learns to encode the state into a structured representation space. The decoder is modified to act as the policy head and maps the encoded feature to the expert action. This architecture introduces a bottleneck that tries to force the robot to retain only critical information from s.
– **CLASS.** [8] A supervised contrastive learning approach. It aligns observations in the representation space based on the similarity of the future action sequences. This allows the robot to learn shared structure across demonstrations.
– **VINN.** [37] A self-supervised approach that separates representation learning and policy learning. The visual encoder is pretrained using the Bootstrap Your Own Latent (BYOL) [36] objective. For new observations, it finds the nearest neighbors in the representation space and chooses the weighted average of the actions corresponding to the neighbors.

5.1 Simulations

We test our method in two simulated scenarios. In the first scenario we have access to privileged state information about the environment. This includes the robot proprioceptive states as well as the position and orientation of all the objects in the environment. In the second scenario we have image observations along with the robot's proprioceptive states.

Simulation Environment. We train and evaluate all methods on three tasks in the Panda-Gym environment [51]. The environment includes a 7 degrees of freedom (DOF) Franka Emika robot arm mounted on a table. Different colored blocks are randomly initialized in the robot's workspace during data collection and evaluation. During data collection, the blocks are randomly initialized within a circle of radius 0.1 meter. During evaluation, we test the methods on in-distribution data, where the blocks are still initialized within the circle of radius 0.1 meter, and out-of-distribution data, where the blocks are initialized within a circle of radius 0.2 meter. Additionally, we change the table on which the robot is mounted during out-of-distribution evaluation. The left columns in Figure 4 display the two scenes. The figure on top shows the robot setup used for data collection and in-distribution evaluation, while the figure on the bottom shows the setup used for out-of-distribution evaluation. Demonstrations are collected using a simulated human teacher and we record the ground truth pose of all objects from the simulation environment.

We design three simulated tasks: picking up and placing a block at a target location (*Pick*), pushing a block to a target location (*Push*), and rotating a Rubik's cube to a desired orientation (*Rotate*). In *Pick* and *Push* tasks, the environment is initialized with a green block that the robot needs to move and a target location. Additionally, 4 different colored blocks are also initialized to act as distractors — the robot should learn that only the green block and the target location are the relevant features in the state, and therefore the robot can ignore the irrelevant information, i.e., the state of the distractor blocks. In *Rotate*, the environment is initialized with 5 Rubik's cube 4 of which have randomly initialized orientation while the desired cube always starts with the same orientation. The robot must pick and rotate the cube by 90° about the longitudinal and vertical axes. The robot should learn to pick up the target cube with the same orientation while ignoring the other cubes and table color.

Demonstrations. At each timestep of the demonstration we collect an RGB image I_{env} of size 256×256 from a static camera that observes the entire simulated environment, an ego-centric image I_{ego} captured from a camera mounted on the robot's end-effector of size 256×256, the robot's state x, and the robot's action a applied by the expert human. In *Pick* and *Push*, x is an 8-dimensional vector including end-effector pose and a boolean gripper state, while a is a 7-dimensional vector including linear and angular velocity of the end-effector and the gripper actuation. In *Rotate*, x is an 8-dimensional vector including the joint angles of the robot and a boolean gripper state, while a is also an 8-dimensional vector including the joint velocities and gripper actuation. This target pose is the Euclidean position where the block should be moved in *Pick* and *Push*; while in *Rotate* it is the desired orientation that the cube must obtain. Additionally, we extract segmentation masks of all the blocks, the robot, the gripper, and the table in the form of bounding boxes $\{\mathcal{B}\}$. The coordinates of the bounding boxes are concatenated into a segmentation feature ϕ_{seg} of dimension 32. This segmentation feature satisfies our assumption that the state is disentangled — only the bounding boxes of the robot and object are relevant for the task.

Procedure. We test our method with two different types of states. In the first case, we use privileged state information which has direct access to the pose of all the blocks in the environment. For this, we record the ground truth pose $x_i, i \in [0, 4]$ of all the blocks in the

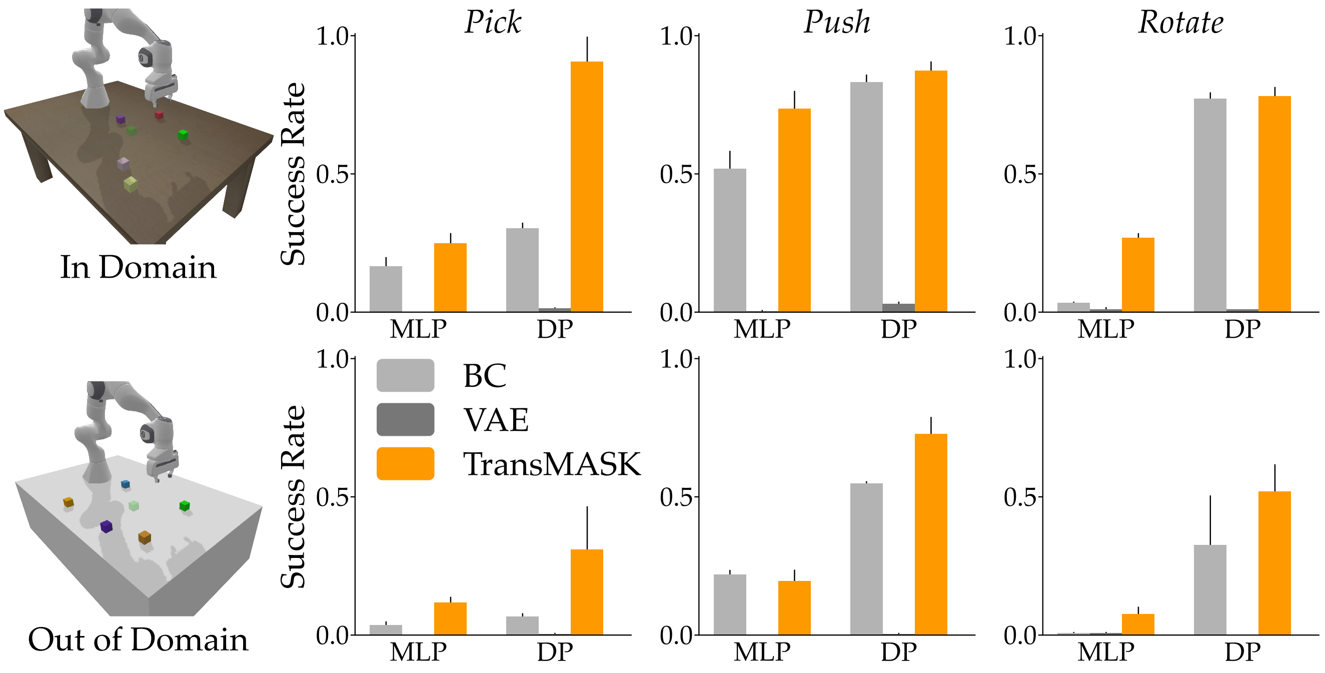

Fig. 4 Results from our simulated experiments with privileged states. We perform the simulated experiments in three tasks — *Pick*, where the robot picks up a green block and places it at a desired location, *Push*, where the robot pushes a green block to the desired location, and *Rotate*, where the robot picks up a Rubik's cube and rotates it by 90° about the longitudinal and vertical axes. We train all methods on demonstrations collected over a wooden table. The methods are trained with two different policy heads: a fully connected policy (MLP) and a diffusion policy (DP). We evaluate the methods in two scenes, denoted as In-Distribution or ID (over wooden table), Out-of-Distribution or OOD (over marble table). The top row shows the results for ID evaluation and the bottom row shows the results for OOD evaluation. The bars indicate success rate over 100 evaluation rollouts averaged across three random seeds.

environment, 0 being the index of the desired block, and the target pose g when collecting demonstrations. Therefore, the state is $s = \left[x^T, x_0^T, x_1^T, x_2^T, x_3^T, x_4^T, g^T\right]$. Our method uses this state and transforms it into a state representation z using the mask M. Intuitively, the policy only needs the state of the robot, the desired block, and the target pose to successfully complete the task. Therefore, z should only include x, x_0, g. Since the CLASS and VINN baselines are designed for image observation data, for this privileged information case we only compare our method with VAE and BC.

Next, we test our method with high-dimensional image observations. In this case, we first use a ResNet-18 vision encoder to map the high-dimensional images I_{env}, I_{ego} to a low-dimensional space ϕ_{env}, ϕ_{ego}. The state includes these image features, the robot state, and the segmentation feature $s = \left[\phi_{env}^T, \phi_{ego}^T, \phi_{seg}^T, x^T\right]$. As in the previous case, our method transforms this state into a state representation z retaining only the relevant information needed to complete the task. Ideally, z should focus on the robot state and segmentation features corresponding to the target object, i.e., $z = \left[x, \phi'_{seg} \subset \phi_{seg}\right]$. For a fair comparison, all the baselines receive the same state s as inputs.

In both these cases, we train the policies on 100, 100, and 150 demonstrations for the *Pick*, *Push*, and *Rotate* tasks, respectively. We highlight that TransMASK, BC, and VAE are trained only on demonstrations collected in the first robot scene shown in the top left corner of Figure 4. Since VINN and CLASS are designed to reduce the discrepancy between the image features of different observations with similar behaviors, we train these two baselines on demonstrations that are collected from both the robot scenes. Concretely, VINN and CLASS are trained on datasets with half of the demonstrations collected from the first robot scene and half from the second scene. Our procedure provides an advantage to VINN and CLASS, since they see examples with both types of tables.

Results. Figure 4 and Figure 5 summarize the results of our simulated experiments. All methods are evaluated for a 100 rollouts each in two robot scenes. In the first scene: In-Distribution (ID), which is shown in the top left corner of Figure 4, the observations are from the same distrbiution as the training set. In the second scene: Out-of-Distribution (OOD), which is shown in the bottom left corner of Figure 4, there is significant distributional shift as the robot is mounted on a different table and the objects are sampled from a larger

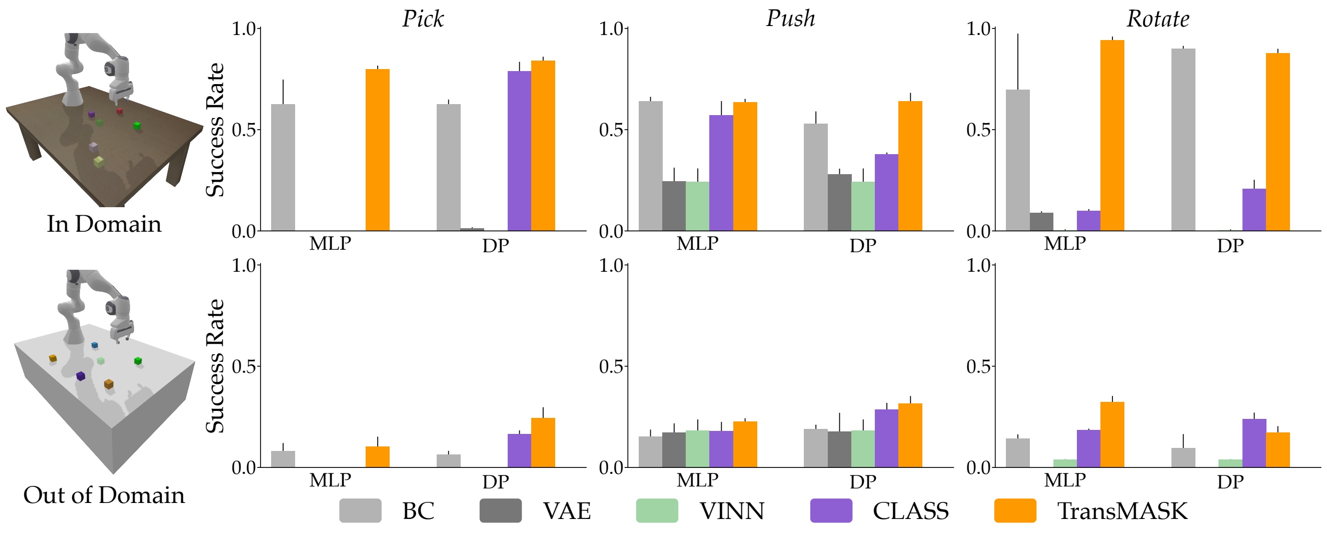

Fig. 5 Results from our simulated experiments with image observations. We perform the simulated experiments in three tasks — *Pick* where the robot picks up a green block and places it at a goal, *Push* where the robot pushes a green block to the desired location, and *Rotate* where the robot picks up a Rubik's cube and rotates it by 90° about the longitudinal and vertical axes. All the methods are trained on demonstrations collected over a wooden table, except VINN and CLASS which are trained on a mixed dataset with equal number of demonstrations collected over a wooden table and a marble table. The methods are trained with two different policy heads: a fully connected policy (MLP) and a diffusion policy (DP). We evaluate the methods in two scenes, denoted as In-Distribution or ID (over wooden table, top row), and Out-of-Distribution or OOD (over marble table, bottom row). While the robot observations in the ID scene are within the training distribution, those in the OOD scene are outside the training distribution. As a clarification, for VINN and CLASS the OOD scene does not cause a distribution shift from the training set. The bars indicate success rate over 100 evaluation rollouts averaged across three random seeds.

	Privileged State				**Image Observations**			
	MLP		DP		MLP		DP	
	ID	OOD	ID	OOD	ID	OOD	ID	OOD
BC	0.23 ± 0.22	0.09 ± 0.1	0.64 ± 0.24	0.33 ± 0.21	0.66 ± 0.18	0.13 ± 0.05	0.69 ± 0.16	0.12 ± 0.068
VAE	0.01 ± 0.003	0.0	0.02 ± 0.02	0.006 ± 0.004	0.08 ± 0.12	0.09 ± 0.08	0.10 ± 0.13	0.06 ± 0.09
VINN	-	-	-	-	0.08 ± 0.12	0.07 ± 0.08	0.08 ± 0.12	0.07 ± 0.08
CLASS	-	-	-	-	0.22 ± 0.25	0.12 ± 0.09	0.46 ± 0.24	0.23 ± 0.06
TransMASK	$\mathbf{0.33 \pm 0.3}$	$\mathbf{0.11 \pm 0.08}$	$\mathbf{0.86 \pm 0.08}$	$\mathbf{0.53 \pm 0.19}$	$\mathbf{0.79 \pm 0.13}$	$\mathbf{0.22 \pm 0.09}$	$\mathbf{0.78 \pm 0.11}$	$\mathbf{0.25 \pm 0.07}$

Table 1 The table shows the overall success rate of TransMASK and the baselines averaged across all the tasks. Each method is evaluated over 100 rollouts in both ID and OOD scenes. The results on the left correspond to sims where the policy has access to privileged states and those on the right correspond to sims with visual observations. We report the mean ± standard deviation of success rate calculated over three random seeds.

region. Importantly, the observations in OOD are truly out-of-distribution only for TransMASK, BC, and VAE, because VINN and CLASS are trained on demonstrations from both these scenes. We report the success rate of each method averaged over three seeds.

5.1.1 Can TransMASK achieve robustness with privileged states?

The top row of Figure 4 shows the success rate in ID and the bottom row shows the success rate in OOD. In each plot, we compare the methods with an MLP action head on the left and a DP action head on the right. TransMASK consistently outperforms the baselines in both test scenes. Larger gains in performance are observed with a DP policy, especially in OOD. While a vanilla BC policy is able to achieve good performance, at least in ID, a naive approach like VAE for learning state representations achieves very poor performance. This is possibly because VAE's reconstruction loss is dominated by the high-variance distractor poses, causing the latent bottleneck to discard the relatively lower-variance target object features. Our method, on the other hand, is able to identify which parts of the privileged states are important for the task as suggested by a lower drop in performance when moving from ID to OOD.

5.1.2 Can TransMASK achieve robustness with image observations?

Next, we compare the methods trained using image observations instead of privileged states. We compare our method against all the baselines; see Figure 5 for a summary of the results. In the ID scene, TransMASK consistently outperforms the baselines across the three tasks. This trend holds for both action heads, MLP and DP. One plausible explanation for the effectiveness of masking even in the ID scene is the presence of distractor objects. Standard imitation learning policies can form spurious correlations with the clutter in the scene, leading to causal confusion. Since the target object and distractors are randomly sampled in our setup, the joint configuration of distractors may vary significantly and become out-of-distribution if the policy fails to ignore them. This issue is less pronounced in *Pick* and *Push* tasks. In these tasks, distractor block colors are sampled randomly, while the target block remains consistently green. This provides a stable visual cue that allows the policy to resolve this confusion without masking. However, in the *Rotate* task, distractors are also Rubik's cubes, making them visually similar to the targe object. As a result, the policy is more likely to form spurious correlations, and the advantages of masking are more prominent. In OOD scenes, all policies experience a performance drop of approximately 75%. This reduction is more severe compared to previous experiments with privileged state. This is likely because the inclusion of ResNet features violates the assumption of state disentanglement. While TransMASK suppresses the majority of the ResNet features, some elements retain small, non-zero magnitudes. Since ResNet features are likely to shift drastically with OOD image observations, these residual values negatively affect performance. Despite this, our method is able to outperform the baselines across all the tasks except the DP policy for *Rotate* which underperforms compared to CLASS. A likely explanation is that while our method is trained on demonstrations collected exclusively in the ID scene, CLASS is trained on demonstrations collected in both ID and OOD scenes. Therefore, CLASS experiences a smaller distribution shift at test time, leading to better performance in the OOD case.

Additionally, in Table 1 we report the overall results of our simulated experiments averaged across all the tasks. TransMASK achieves higher success rate than all baselines for both privileged states and image features. In particular, TransMASK outperforms the next best baseline by approximately 15% in the ID scene and by about 9% in the OOD scene. Interestingly, TransMASK achieves a higher success rate than CLASS in the OOD scene, even though CLASS is trained on demonstrations collected directly in that setting. A repeated measures ANOVA test revealed that TransMASK significantly outperforms the baselines when using privileged state information (ID: $F(2, 16) = 9.516, p < 0.005$; OOD: $F(2, 16) = 8.368, p < 0.005$). Our method also performs significantly better than all the baselines except CLASS with image observations (ID: $F(4, 32) = 27.019, p < 0.005$; OOD: $F(4, 32) = 37.25, p < 0.005$). Overall, our simulated results suggest that our masking approach effectively extracts task-relevant information from the state, leading to improved policy robustness to environmental variations.

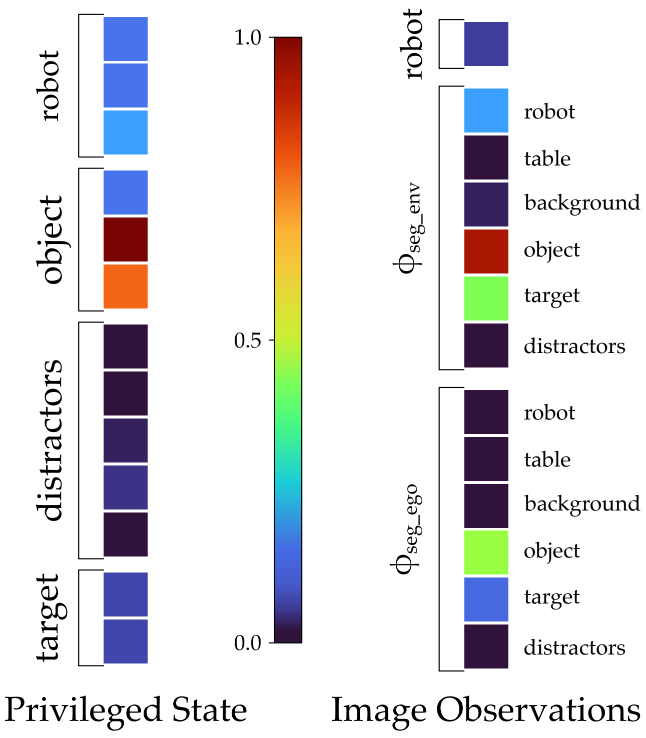

Fig. 6 Example learned masks in our simulated experiments. Our method learns a mask $M \in \mathbb{R}^{n \times n}$, and computes the latent state as $z = Ms$. Therefore, the magnitude of the i-th column of M corresponds to the i^{th} element of s. As the robot policy converges, the values in the columns of M which correspond to task-irrelevant elements of s should ideally approach zero. To visualize the relative importance assigned to each element of s, for each column of M, we take the sum of its entries and normalize the resulting vector such that its values lie in the range $[0, 1]$. Here, we show two representative examples from our simulated experiments with privileged state and image observations. We label which components of the column relate to which information in the scene. The components with a dark blue shade have values closer to 0, indicating that the corresponding feature of the scene is removed from z. In contrast, the components with a red shade have values closer to 1, indicating that the associated feature is retained in z.

5.1.3 Does TransMASK learn correct masks?

After examining the results, we visualize the masks learned by TransMASK to examine how it weights the elements of the states. The mask M output by our approach is an $n \times n$ matrix where n is the dimension of the state. To visualize how the mask assigns importance to different elements of the state, we normalize each column of M. Specifically, for each column, we compute the sum of all its elements and rescale the resulting column such that its values lie in the range $[0, 1]$. These normalized values of the columns indicate which elements of the state are allowed to pass to the state representation z. The normalized values of the columns are close to zero for elements of s that are identified as irrelevant and are not included in z.

We group the normalized mask columns corresponding to the different objects in the scene and display a representative example in Figure 6. We find that for privileged states, the mask assigns high weights for the elements corresponding to the object and non-zero values for the elements corresponding to the robot's proprioceptive state and the target location. In contrast, the elements of the state that correspond to the distractors are down-weighted to zero. Similarly, in the case where s includes robot proprioceptive state and segmentation features the elements corresponding to the bounding boxes of the object, the robot, and the target are non-zero while those corresponding to the bounding boxes of the table, the robot, and the background are zero. Interestingly, this is found to be true for segmentation features obtained from both the static camera as well as the robot mounted camera.

5.2 Real-World Experiments

Next, we move to a real-world setting where a robot arm performs table-top manipulation tasks. For this setting we only test the methods with image observations. Unlike our simulation environment, the images in a real-world setting present unexpected challenges: the objects have textures and shadows, the lighting conditions are not constant, and the robot may not segment the objects consistently and accurately. Our objective is to test if our proposed method can still be effective in this imperfect scenario.

Experiment Setup. We train and evaluate all methods on a 6 DOF UR10 (Universal Robots) robot mounted on a table. We use two Logitech C920 cameras to capture image observations. One camera is mounted on the robot's end-effector to record the ego-centric images and the other camera is mounted on the table such that it observes the entire robot environment. At the start of each task we place the desired object in the environment which the robot needs to manipulate. Additionally, we place other objects in the environment as distractors that can confuse the robot. We want to test whether the robot learns to ignore the clutter in the environment and focus only on the object it needs to manipulate. Similar to the simulated tasks, during evaluation we consider two conditions: in-distribution data, where the scene is the same as during data collection; and out-of-distribution data where we cover the table with a sheet to change the background.

We test our method in three tasks: picking up and placing a block at a target location (*Pick*), stacking two LEGO blocks (*Stack*), and scooping from a bowl placed on the table (*Scoop*). In all tasks, the target object is initialized within a predefined region on the left side of the robot workspace, with 2-3 additional objects randomly placed on the table as distractors.

We compare TransMASK against the same baselines used in the simulated experiments — BC, VAE, VINN, and CLASS. We train each method on 40, 25, and 20 demonstrations for the *Pick*, *Stack*, and *Scoop* tasks, respectively. Preliminary experiments showed that the MLP policy struggled to learn the tasks reliably; therefore, all real-world experiments are conducted using a diffusion policy (DP).

Demonstrations. We collect robot demonstrations using a joystick to teleoperate the robot. At each timestep, we collect two images of size 256×256 from the robot and table mounted cameras (I_{ego}, I_{env} respectively), the robot's state x, and the robot action a applied by the human demonstrator. Robot state x is a 7-dimensional vector including the end-effector pose and a boolean gripper state, the robot action is also a 7-dimensional vector consisting of linear and angular end-effector velocity along with the gripper actuation. To detect the segmentation masks of the objects we use the open-source langSAM model [52] built on SegmentAnything [53] and GroundingDINO [54]. langSAM allows us to extract bounding boxes $\{\mathcal{B}\}$ for all the objects in the environment which are concatenated together to form the segmentation feature ϕ_{seg}. Notably, the while langSAM provides bounding boxes for everything present in the environment, the robot still doesn't explicitly know which bounding box is important for task completion.

Results. Figure 7 summarizes the results of our real-world experiments. As in simulated experiments, we consider two robot scenes: first, an In-Distribution (ID) scene where the observations are from the same distribution as the training set, and second, an Out-of-Distribution (OOD) scene in which a distributional shift is introduced by changing the color of the table. We placed a white sheet on the surface — note that this also removed reflections and changed the intensity of the shadows. These ID and OOD scenes are shown in the left column of Figure 7. Each method is evaluated

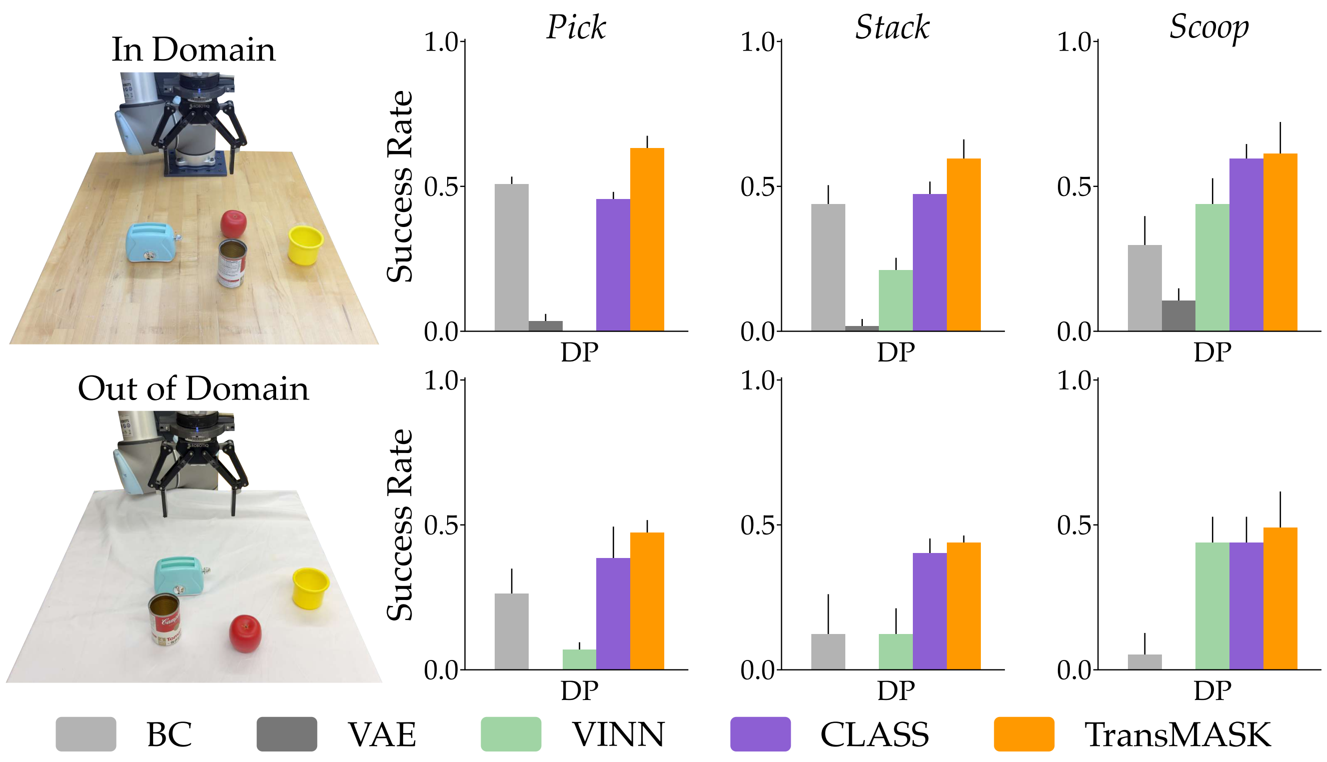

Fig. 7 Results of our real-world experiments. We test the methods in three tasks — *Pick*, where the robot picks up a yellow cup and places it at a desired location, *Stack*, where the robot stacks a LEGO block over another, and *Scoop*, where the robot scoops from a bowl. We evaluate the methods in two scenes, denoted as In-Distribution or ID (over wooden table), Out-of-Distribution or OOD (the table is covered with a white sheet). All the methods are trained on demonstrations collected over the wooden table, except VINN and CLASS — which are trained on a mixed dataset with equal number of demonstrations collected from ID and OOD scenes. We train all the methods with a diffusion policy (DP) head. While the robot observations in the ID scene are within the training distribution, those in the OOD scene are outside the training distribution. As a clarification, for VINN and CLASS the OOD scene does not cause a distribution shift from the training set. We compare the methods using success rate as the metric. The top row shows the results for ID evaluation and the bottom row shows the results for OOD evaluation. The bars indicate success rate over 20 evaluation rollouts averaged across three random seeds.

over 20 rollouts per scene, and we report the success rates averaged over three random seeds.

Across all three tasks, TransMASK consistently outperforms all the baselines in the ID scene. Our method significantly outperforms BC and VAE in the OOD scene as well. It also substantially outperforms VINN in both scenes on *Pick* and *Stack*. However, in *Scoop*, TransMASK achieves a substantial gain in performance in ID and only a modest improvement in OOD. Similarly, compared to CLASS our method achieves slight improvement in the success rate in OOD. Notably, our method is trained exclusively on demonstrations collected in the ID scene, *whereas VINN and CLASS are trained on a mixed dataset containing equal numbers of ID and OOD demonstrations.* Although performance in the ID scene is generally higher than in the OOD scene, the degradation under distribution shift is less pronounced than in the simulated experiments. This is primarily explained by the fact that the target object positions vary less drastically in the real-world setup — it is sampled from a smaller region, with variations within 5 cm. A repeated measures ANOVA across all three tasks revealed that TransMASK significantly outperforms VAE and VINN in the ID scene ($\mathrm{F}(4, 32) = 28.934, p < 0.001$), and it achieves significant improvement over all baselines except CLASS in the OOD scene ($\mathrm{F}(4, 32) = 21.375, p < 0.001$). Overall, these results suggest that TransMASK effectively conditions the policy on task-relevant state representations, enabling improved robustness to visual distribution shifts.

6 Conclusion

In this manuscript we explored the problem of robust imitation learning under environmental variations. One way to make learned policies more robust is by pruning the states they are conditioned on; instead of a policy which depends on every observed feature, we want the robot to only reason over features that are relevant to the current task. To this end, we introduced TransMASK, a modular mask which can be added to a variety of standard imitation learning frameworks. TransMASK does not require modifying the imitation learning objective or the training procedure; instead, it relies on the intrinsic gradients of the optimization scheme. Specifically, TransMASK learns a sparse mask matrix that transforms the input state into a compressed representation by eliminating irrelevant components (e.g., background color, table texture, clutter). The resulting representation preserves features that significantly influence the policy's loss, thereby aligning the learned representations with task structure. By conditioning the policy on this compressed state representation, TransMASK produces policies that generalize better to distributional shifts.

Additionally, we analyzed the commonly used representation strategies in imitation learning, including the information bottleneck objective and contrastive learning. We showed how these approaches can inadvertently introduce instability or eliminate task-critical information due to the misalignment of the auxiliary objective with policy optimization. By contrast, TransMASK directly exploits gradients from the primary imitation objective, ensuring that the feature selection is tightly coupled to task performance. The structure of M also prevents TransMASK from learning a state embedding which collapses into an action representation.

We tested TransMASK through experiments with state-of-the-art baselines in both simulated environments and real-world robotic manipulation tasks. Across all settings, particularly under environmental perturbation, TransMASK consistently achieved significantly higher success rates, demonstrating improved robustness.

Limitations and Future Work. Our core limitation lies in our assumption that the input space admits a clear separation between task-relevant and task-irrelevant features, i.e., the state s is sufficiently disentangled into μ and η. In practice, this disentanglement is not guaranteed. Object-level segmentation offers a promising mechanism to approximate this structure by explicitly decomposing the scene into semantically meaningful features. Although segmentation models introduce their own failure modes, our real-world experimental results indicate that these do not constitute a significant bottleneck for our proposed method. A second limitation arises from the implicit learning of the mask. The mask is not trained with direct supervision; instead, it is optimized through policy gradients during training. Consequently, its quality depends on the stability of the optimization process and the quality of the training data. In cases where the loss landscape is noisy or training data is limited, the policy may converge to a suboptimal solution, leading the mask to encode incorrect features. This issue is particularly likely in small datasets, where irrelevant features may exhibit accidental correlations with actions, causing the mask to include irrelevant features in the state representation. Therefore, our method should be applied with careful consideration of data quality, dataset size, and optimization stability.

While our experiments focus on imitation learning, the underlying principle is not restricted to this setting. The approach could potentially be extended to reinforcement learning, particularly in scenarios involving sim-to-real transfer, where robustness to distribution shift and the extraction of task-relevant features are critical. Future work will focus on exploring the application of TransMASK to this application. Another potential avenue for future research is to provide theoretical guarantee for the convergence of the mask.

7 Declarations

Funding. This research was supported in part by the USDA National Institute of Food and Agriculture, Grant 2022-67021-37868.

Conflict of Interest. The authors declare that they have no conflicts of interest.

Author Contribution. S.P.: Conceptualization, Investigation, Software, Methodology, Formal analysis, Writing - original draft. P.C.: Conceptualization, Supervision, Writing - review and editing. D.L.: Conceptualization, Supervision, Funding Acquisition, Writing - review and editing.

References

1. Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30, 2017.
2. Oleg Kaidanov, Firas Al-Hafez, Yusuf Suvari, Boris Belousov, and Jan Peters. The role of domain randomization in training diffusion policies for whole-body humanoid control. *arXiv preprint arXiv:2411.01349*, 2024.
3. Mohit Sharma, Claudio Fantacci, Yuxiang Zhou, Skanda Koppula, Nicolas Heess, Jon Scholz, and Yusuf Aytar. Lossless adaptation of pretrained vision models for robotic manipulation. *arXiv preprint arXiv:2304.06600*, 2023.
4. Jingyun Yang, Max Sobol Mark, Brandon Vu, Archit Sharma, Jeannette Bohg, and Chelsea Finn. Robot fine-tuning made easy: Pre-training rewards and policies for autonomous real-world reinforcement learning. In *International Conference on Robotics and Automation (ICRA)*, pages 4804–4811, 2024.

5. Rouhollah Rahmatizadeh, Pooya Abolghasemi, Ladislau Bölöni, and Sergey Levine. Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. In *IEEE international conference on robotics and automation (ICRA)*, pages 3758–3765, 2018.
6. Jinghuan Shang and Michael S Ryoo. Self-supervised disentangled representation learning for third-person imitation learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 214–221, 2021.
7. Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.
8. Sung-Wook Lee, Xuhui Kang, Brandon Y Yang, and Yen-Ling Kuo. Class: Contrastive learning via action sequence supervision for robot manipulation. In *Conference on Robot Learning (CoRL)*, pages 4743–4766, 2025.
9. Vittorio Giammarino, James Queeney, and Ioannis Ch Paschalidis. Visually robust adversarial imitation learning from videos with contrastive learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 15642–15648, 2025.
10. Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44:1684–1704, 2025.
11. Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, and Krzysztof Choromanski. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *arXiv preprint arXiv:2307.15818*, 2023.
12. Pranay Gupta, Henny Admoni, and Andrea Bajcsy. Adapting by analogy: Ood generalization of visuomotor policies via functional correspondence. *arXiv preprint arXiv:2506.12678*, 2025.
13. Jilei Mao, Jiarui Guan, Yingjuan Tang, Qirui Hu, Zhihang Li, Junjie Yu, Yongjie Mao, Yunzhe Sun, Shuang Liu, and Xiaozhu Ju. Omnid: Generalizable robot manipulation policy via image-based bev representation. *arXiv preprint arXiv:2508.11898*, 2025.
14. Vaibhav Saxena, Matthew Bronars, Nadun Ranawaka Arachchige, Kuancheng Wang, Woo Chul Shin, Soroush Nasiriany, Ajay Mandlekar, and Danfei Xu. What matters in learning from large-scale datasets for robot manipulation. *arXiv preprint arXiv:2506.13536*, 2025.
15. Modi Shi, Li Chen, Jin Chen, Yuxiang Lu, Chiming Liu, Guanghui Ren, Ping Luo, Di Huang, Maoqing Yao, and Hongyang Li. Is diversity all you need for scalable robotic manipulation? *arXiv preprint arXiv:2507.06219*, 2025.
16. Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, and Abhishek Gupta. Open x-embodiment: Robotic learning datasets and rt-x models : Open x-embodiment collaboration0. In *International Conference on Robotics and Automation (ICRA)*, pages 6892–6903, 2024.
17. Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning (CoRL)*, 2021.
18. Sagar Parekh, Heramb Nemlekar, and Dylan P Losey. Towards balanced behavior cloning from imbalanced datasets. *Autonomous Robots*, 50:9, 2026.
19. Joey Hejna, Chethan Bhateja, Yichen Jiang, Karl Pertsch, and Dorsa Sadigh. Re-mix: Optimizing data mixtures for large scale imitation learning. *arXiv preprint arXiv:2408.14037*, 2024.
20. Haoran Xu, Xianyuan Zhan, Honglei Yin, and Huiling Qin. Discriminator-weighted offline imitation learning from suboptimal demonstrations. In *International Conference on Machine Learning (ICML)*, pages 24725–24742, 2022.
21. Geon-Hyeong Kim, Seokin Seo, Jongmin Lee, Wonseok Jeon, HyeongJoo Hwang, Hongseok Yang, and Kee-Eung Kim. Demodice: Offline imitation learning with supplementary imperfect demonstrations. In *International Conference on Learning Representations (ICLR)*, 2021.
22. Liyiming Ke, Yunchu Zhang, Abhay Deshpande, Siddhartha Srinivasa, and Abhishek Gupta. Ccil: Continuity-based data augmentation for corrective imitation learning. *arXiv preprint arXiv:2310.12972*, 2023.
23. Jung Yeon Park and Lawson Wong. Robust imitation of a few demonstrations with a backwards model. *Advances in Neural Information Processing Systems*, 35:19759–19772, 2022.
24. Allan Zhou, Moo Jin Kim, Lirui Wang, Pete Florence, and Chelsea Finn. Nerf in the palm of your hand: Corrective augmentation for robotics via novel-view synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17907–17917, 2023.
25. Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65:99–106, 2021.
26. Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.
27. Xin Zhang, Yanhua Li, Ziming Zhang, and Zhi-Li Zhang. f-gail: Learning f-divergence for generative adversarial imitation learning. *Advances in neural information processing systems*, 33:12805–12815, 2020.
28. Jianren Wang, Ziwen Zhuang, Yuyang Wang, and Hang Zhao. Adversarially robust imitation learning. In *Conference on Robot Learning (CoRL)*, pages 320–331, 2022.
29. Dahuin Jung, Hyungyu Lee, and Sungroh Yoon. Sample-efficient adversarial imitation learning. *Journal of Machine Learning Research*, 25:1–32, 2024.
30. Ziyu Wang, Josh S Merel, Scott E Reed, Nando de Freitas, Gregory Wayne, and Nicolas Heess. Robust imitation of diverse behaviors. *Advances in Neural Information Processing Systems*, 30, 2017.
31. Kaifeng Zhang, Rui Zhao, Ziming Zhang, and Yang Gao. Auto-encoding adversarial imitation learning. *arXiv preprint arXiv:2206.11004*, 2022.
32. Chun-Mao Lai, Hsiang-Chun Wang, Ping-Chun Hsieh, Frank Wang, Min-Hung Chen, and Shao-Hua Sun. Diffusion-reward adversarial imitation learning. *Advances in Neural Information Processing Systems*, 37:95456–95487, 2024.
33. Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: Robot manipulation with multimodal prompts. 2023.
34. Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747, 2021.
35. Jonathan Yang, Chelsea Finn, and Dorsa Sadigh. Invariance co-training for robot visual generalization. *arXiv preprint arXiv:2512.05230*, 2025.

36. Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.
37. Jyothish Pari, Nur Muhammad Shafiullah, Sridhar Pandian Arunachalam, and Lerrel Pinto. The surprising effectiveness of representation learning for visual imitation. *arXiv preprint arXiv:2112.01511*, 2021.
38. Bian Xihan, Oscar Mendez, and Simon Hadfield. Skill-il: Disentangling skill and knowledge in multitask imitation learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7060–7065, 2022.
39. Michael Noseworthy, Rohan Paul, Subhro Roy, Daehyung Park, and Nicholas Roy. Task-conditioned variational autoencoders for learning movement primitives. In *Conference on robot learning (CoRL)*, pages 933–944, 2020.
40. Jongjin Park, Younggyo Seo, Chang Liu, Li Zhao, Tao Qin, Jinwoo Shin, and Tie-Yan Liu. Object-aware regularization for addressing causal confusion in imitation learning. *Advances in Neural Information Processing Systems*, 34:3029–3042, 2021.
41. Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *Journal of Machine Learning Research*, 19:1–34, 2018.
42. Alessandro Achille and Stefano Soatto. A separation principle for control in the age of deep learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:287–307, 2018.
43. Shuanghao Bai, Wanqi Zhou, Pengxiang Ding, Wei Zhao, Donglin Wang, and Badong Chen. Rethinking latent redundancy in behavior cloning: An information bottleneck approach for robot manipulation. *arXiv preprint arXiv:2502.02853*, 2025.
44. Ritwik Bera, Vinicius G Goecks, Gregory M Gremillion, Vernon J Lawhern, John Valasek, and Nicholas R Waytowich. Gaze-informed multi-objective imitation learning from human demonstrations. *arXiv preprint arXiv:2102.13008*, 2021.
45. Robert M Gray. *Entropy and information theory*. Springer Science & Business Media, 2011.
46. Ashvin Nair, Shikhar Bahl, Alexander Khazatsky, Vitchyr Pong, Glen Berseth, and Sergey Levine. Contextual imagined goals for self-supervised robotic learning. In *Conference on robot learning (CoRL)*, pages 530–539, 2020.
47. Andre Martins and Ramon Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International conference on machine learning (ICML)*, pages 1614–1623, 2016.
48. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
49. Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.
50. Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
51. Quentin Gallouédec, Nicolas Cazin, Emmanuel Dellandréa, and Liming Chen. panda-gym: Open-source goal-conditioned environments for robotic learning. *arXiv preprint arXiv:2106.13687*, 2021.
52. Luca Medeiros. Lang segment anything: Segment anything model with text prompts. https://github.com/luca-medeiros/lang-segment-anything.git, 2025. GitHub repository, accessed January 14, 2025.
53. Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.
54. Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.