# DISENTANGLEMENT FROM NONLINEAR TASK SPARSITY

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

In this work, we study representation learning through *sparsity* of the task: each task depends on only small subset of the factors in the representation. Prior work has theoretically shown that such representations can be recovered through sparsity-aware multi-task learning. However, practical realizations have been limited to linear task heads, limiting their scalability. Extending task sparsity to nonlinear predictors is challenging because sparsity is inherently discrete and therefore difficult to optimize with gradient-based methods. To address this challenge, we propose **Disentanglement from Task Sparsity (DTS)**, a general sparsity-aware representation learning framework for recovering disentangled representations from nonlinear multi-task datasets. DTS formulates task sparsity as a task-conditioned binary masking problem, imposing sparsity only through the masking operation and allowing nonlinear task heads to learn from the masked features. Empirically, we evaluate DTS on a multi-task dataset generated following the CLEVR protocol (Johnson et al., 2017). We show that DTS recovers the underlying disentangled factors from nonlinear tasks and the learned representation enables strong OOD generalization through sparse downstream task learning. Finally, our analysis shows that both the masking architecture and the sparsity objective are crucial for learning disentangled representations from nonlinear tasks.

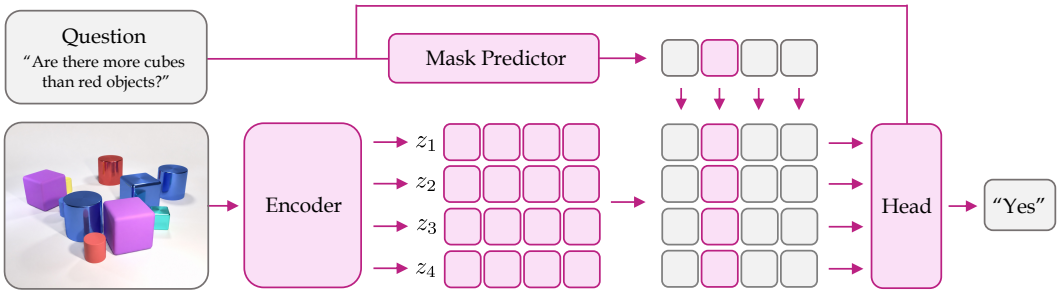


Figure 1: **Architecture of DTS**. DTS formulates the sparsity optimization problem as a task-conditioned binary masking problem. Sparsity is imposed by penalizing the masking probability, which can be optimized with gradient descent.

## 1 INTRODUCTION

Occam's razor suggests that, among hypotheses that explain the observed data equally well, we should prefer the simpler one (Domingos, 1998; Rissanen, 1978). Among several possible definitions of simplicity (Rissanen, 1978; Vapnik & Chervonenkis, 1971; Bartlett & Mendelson, 2002), we focus on the *sparsity* of a task in this work: the solution to the task should depend on the minimal subset of the underlying ground-truth factors (Jalali et al., 2010; Maurer et al., 2013). For example, a robotic policy grasping a mug should ignore distractors or lighting conditions, and predicting a pedestrian's path should not depend on its clothing or other minor visual attributes. If the representation disentangles such task-relevant factors from irrelevant ones, a *sparse* predictor that prefers to prune out irrelevant factors can support out-of-distribution (OOD) generalization (Fumero et al., 2023; Lachapelle et al., 2023; 2024).



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

How can we learn such disentangled representation from data, without access to the ground-truth factors? Prior works has theoretically shown that such representations can be recovered from multi-task learning with sparsity-aware predictors, which try to minimize the number of input factors used for each task (Lachapelle et al., 2023; Fumero et al., 2023; Lachapelle et al., 2024). However, practical realizations of those methodologies were limited to linear task heads, restricting those to tasks that are linearly decodable from the representation, lacking scalability.

The main challenge of optimizing task sparsity for nonlinear predictors stems from the discontinuity of sparsity. Ideally, we would like to minimize the number of latent factors each task depends on, but this objective is discontinuous and thus cannot be optimized with gradient-based methods. While this $\ell_0$ optimization problem can be relaxed with $\ell_1$ regularization for linear predictors (Tibshirani, 1996), this is no longer true for neural networks (Scardapane et al., 2017; Yamada et al., 2020). Therefore, extending task sparsity beyond linear predictors requires a fundamentally different approach compared to prior works.

To this end, we propose **Disentanglement from Task Sparsity (DTS)**, a general sparsity-aware representation learning framework that recovers disentangled representation from nonlinear multi-task datasets. DTS optimizes the task sparsity via gradient descent, by formulating the sparsity optimization problem as a task-conditioned binary masking problem. Specifically, given a task, DTS predicts a masking probability for each latent dimension and samples a binary mask from the corresponding Bernoulli distribution. These probabilities can be minimized via gradient descent for task sparsity, as the sum of the probabilities is exactly the expected $\ell_0$ norm of the sampled mask (Yamada et al., 2020). Therefore, by penalizing these probabilities, we can directly minimize the expected number of latent factors available to each task. For learning the task, we can attach any nonlinear task head on top of the masked representation.

Empirically, we evaluate DTS on a synthetic multi-task dataset consisting of 1M images and over 3000 tasks, generated following the CLEVR protocol (Johnson et al., 2017). We first show that DTS can recover the underlying disentangled representation from this nonlinear multi-task dataset, while prior disentanglement works cannot. Moreover, we show that this disentangled representation can enable OOD generalization via sparsity. Finally, we show that the DTS's masking architecture and training objectives are crucial for learning disentangled representation from nonlinear tasks. These empirical evaluations show that DTS learns disentangled representation from nonlinear multi-task datasets, and enables OOD generalization by utilizing *task sparsity* as an inductive bias.

## 2 Related Works

**Representation Learning.** Representation learning aims to learn a representations that capture useful structure from data and facilitate learning of downstream tasks. Early approaches learn representations through autoencoding and reconstruction objectives (Vincent et al., 2008; Hinton & Salakhutdinov, 2006; Vincent et al., 2008), while more recent methods leverage self-supervised and contrastive objectives to learn from large-scale unlabeled data (Oord et al., 2018; Chen et al., 2020; He et al., 2020; Grill et al., 2020; He et al., 2022). Multi-task learning provides another approach by jointly training a shared representation across multiple supervised tasks, encouraging the representation to capture factors useful across tasks (Caruana, 1997; Standley et al., 2020; Zamir et al., 2018; Xin et al., 2022). More recently, large-scale multimodal and vision language models (VLMs) have demonstrated that representations learned from diverse auxiliary supervision can transfer effectively to new tasks (Radford et al., 2021; 2019; Jia et al., 2021; Beyer et al., 2024). Our work shares the goal of learning transferable representations from multi-task learning, but emphasizes that the representation alone is insufficient for OOD generalization (Section 4.1), without the assumption in the structure of the task and a learning rule that exploits that structure (Section 4.2). We therefore learn a representation that each tasks depends on small subset of the representation, combined with sparsity-aware learning rule for downstream tasks.'

**Disentangled Representations.** Disentangled representation learning aims to learn representations that distinct latent factors correspond to distinct dimensions or components, separating the underlying sources of variation in the data (Higgins et al., 2018; Locatello et al., 2019). A large body of work has explored disentanglement through generative models (Higgins et al., 2017; Kim & Mnih, 2018; Chen et al., 2018; 2016; Kumar et al., 2017; Yang et al., 2022; 2024; Jun et al., 2025), weak or supervised signals (Shu et al., 2019; Locatello et al., 2020a; Khemakhem et al., 2020), and inductive

biases such as independence and group structure (Moran et al., 2021; Zheng et al., 2022; Quessard et al., 2020; Klindt et al., 2020). However, prior work primarily evaluates disentanglement according to properties of the representation itself, such as statistical independence or alignment with predefined semantic factors. In contrast, we study disentanglement from the perspective of *task sparsity*, where a representation can express diverse family of tasks only using its sparse subsets. This view allows the representation to be identified from multi-task learning and provides a direct connection between representation learning and OOD generalization of unseen tasks.

**Disentangled Representations from Task Sparsity.** Task sparsity has long been explored for learning disentangled and transferable representations. In particular, Lachapelle et al. (2023) shows that when tasks can be predicted by sparse linear map, sufficiently rich task distributions can identify disentangled representations and improve generalization. Fumero et al. (2023) similarly introduce sparse sufficiency and feature minimality as principles for learning disentangled representations with multi-task learning. Sparsity has also been exploited with causal mechanisms (Lachapelle et al., 2022; 2024), tree structures (Layne et al., 2024) to improve OOD generalization. However, existing practical realizations of prior works rely on linear task heads, limiting their scalability to realistic multi-task datasets where solving each task require expressive nonlinear computation. Our work extends task sparsity to nonlinear tasks by formulating sparse factor selection as a task-conditioned sparse mask prediction.

## 3 Preliminaries

**Problem setting (OOD generalization).** Let $P_X$ denote a distribution over the input space $\mathcal{X}$, and let $P_U$ denote the conditional distribution restricted to a subspace $U \subset \mathcal{X}$, i.e., $\mathrm{supp}(P_U) \subsetneq \mathrm{supp}(P_X)$. We assume access to a large multi-task dataset sampled from $P_X$, where each sample is annotated with labels for a collection of auxiliary tasks. In addition, we are given a new downstream task whose supervision is available only for samples drawn from $P_U$. Our objective is to learn a representation $\varphi : \mathcal{X} \to \mathcal{Z}$ from the multi-task learning from $P_X$, such that a downstream predictor trained solely on labeled samples from $P_U$ generalizes to the full distribution $P_X$.

This setting differs from conventional learning setup, where the samples from the training set eventually cover the test distribution. Here, even infinitely many labeled samples from $P_U$ cannot reveal the behavior of the task outside $U$. Consequently, OOD generalization requires additional structural assumptions on the task, and requires the representation (and the downstream learning rule) to exploit those structures.

## 4 Motivation: Where OOD Generalization is Possible

In this section, we provide a theoretical motivation for *task sparsity* as an inductive bias for OOD generalization. We first show that OOD generalization is fundamentally impossible with representation learning alone, without structural assumptions on the task family and a learning rule that exploits that structure (Section 4.1). In contrast, if the downstream tasks have a sparse structure, then having both sparsity-aware learning rule and disentangled representation enables OOD generalization by uniquely identifying the downstream solution (Section 4.2). Finally, we establish that the disentangled representation can be recovered from sufficiently rich auxiliary tasks with sparsity-aware learning rule (Section 4.3). Throughout, we consider a binary latent space $\mathcal{Z} = \{0, 1\}^d$.

### 4.1 OOD generalization is impossible with representation alone

We begin with the unconstrained setting, where the learning rule makes no structural assumptions about the downstream task. Recall that the downstream task is observed only on a restricted subspace $U \subsetneq X$, while evaluation is performed over the full distribution $X$. Even with infinitely many labeled samples from $U$, there are generally many functions that agree on all observations in $U$ but behave differently outside it. Thus, without additional task structure, the target function is fundamentally unidentifiable.

**Theorem 1** (Impossibility without Task Structure)**.** *Let $\mathcal{F}$ denote the family of all measurable downstream tasks $f : \mathcal{X} \to \mathbb{R}$. For any representation $\varphi : \mathcal{X} \to \mathcal{Z}$, there exists a downstream task $f \in \mathcal{F}$*

*and a restricted subspace $U \subsetneq X$ such that labeled observations of $f$ restricted to $U$ do not uniquely determine $f$ on $X$.*

Theorem 1 shows that no representation can resolve this ambiguity: multiple predictors can perfectly match with all observations on $U$ while making different predictions outside it. In particular, a representation may achieve generalization by discarding factors that are irrelevant to one particular task, but those factors may be essential for other tasks. This motivates introducing structural assumptions on the downstream task family, along with a learning rule that exploits them. In the next section, we show that task sparsity provides such a structure, and can enable OOD generalization beyond the observed subspace, when sparsity is incorporated into the learning rule.

### 4.2 OOD GENERALIZATION IS POSSIBLE WHEN TASKS ARE SPARSE

We introduce two assumptions that characterize when OOD generalization is possible. First, we assume that there exists a latent representation in which downstream tasks depend on a small subset of the underlying factors. This assumption reflects the observation that many real-world tasks are sparse; for example, a VQA question typically depends only on a few attributes, and a robotic manipulation task depends only on the objects and physical factors relevant to the task.

**Assumption 1** (Sparse Disentangled Representation)**.** *There exists a latent representation $z^\star = \varphi^\star(x) \in \mathcal{Z}$ such that every task is a sparse function of the latent variables. Specifically, for every task $t$, there exists a predictor $f_t$ satisfying*

$$y_t = f_t(z^\star),$$

*where $f_t$ depends on only a sparse subset of the latent dimensions of $z^\star$. Furthermore, the latent is fully utilized:*

$$p(z^\star = z) > 0, \forall z \in \mathcal{Z}.$$

Assumption 1 formalizes the desired structure of the latent representation: each task depends on only a small subset of latent factors. Task sparsity limits the information that a downstream predictor must rely on, while full utilization ensures that each dimension of the latent is essential for the downstream tasks. Together, these properties allow the representation to support OOD generalization, when the labeled subspace contains sufficient information to identify the downstream task via sparsity.

**Assumption 2** (Sparse Task Minimality (*Occam's Razor*))**.** *Let $z^\star = \varphi^\star(x)$ satisfy Assumption 1, and let the downstream task be generated as*

$$y = f(z^\star),$$

*where $f$ is $k$-sparse. The ground-truth task $f$ is the unique sparsest predictor consistent with the supervision on the labeled subspace $U$. Specifically, if a predictor $g$ satisfies*

$$g(z^\star) = f(z^\star), \qquad \forall x \in U,$$

*then $g$ depends on at least as many latent variables as $f$, with equality only if $g \equiv f$.*

Assumption 2 assumes the sparsity version of *Occam's razor*: among hypotheses that explain the observed data, the ground-truth task is the sparsest one. With this assumption, now we show that sparsity-aware learning rule can enable OOD generalization.

**Theorem 2** (OOD generalization under Sparse Representations)**.** *Let $z^\star = \varphi^\star(x)$ denote the latent representation satisfying Assumption 1. Suppose the downstream task is generated as*

$$y = f(z^\star),$$

*where $f$ is $k$-sparse and $x$ is sampled from subspace $U$ following Assumption 2.*

*Then the learning rule that maximizes the sparsity of the predictor from labeled samples in $U$ uniquely recovers $f$ and therefore generalizes to the full dataset $X$.*

Theorem 2 demonstrates that if the learning rule chooses most simplest solution among all predictors that explains the dataset, then the solution is unique and equals the ground-truth task, therefore generalizes beyond the support of the labeled data. Notably, the learner requires no knowledge of the sparsity level $k$: the sparsity of the task is recovered by the learning rule, not assumed by it (Please refer to Section C.2 for a formal proof). The remaining question is therefore how to recover such a representation.

### 4.3 Sparse Representations are Uniquely Identified by Sparse Tasks

Theorem 2 assumes access to a latent representation satisfying Assumption 1. We now show that such a representation can be identified from a sufficiently rich collection of sparse auxiliary tasks.

**Theorem 3** (Identifiability from Task Sparsity (Informal))**.** *Let $z^\star = \varphi^\star(x) \in \mathcal{Z}$ denote the ground-truth representation satisfying Assumption 1, and suppose the auxiliary tasks are sampled from a sufficiently rich sparse-support task distribution $\mathcal{D}_\mathcal{T}$. Then, every global optimizer $\hat{z}^\star$ of a sparsity-aware multi-task learning rule satisfies*

$$\hat{z}^\star = P\left(h_1(z_1^\star(x)), \ldots, h_d(z_d^\star(x))\right),$$

*where $P$ is a permutation and each $h_i$ is invertible.*

Theorem 3 establishes that the representation can be recovered from multi-task learning with sparsity-aware learning rule, up to permutation and element-wise transformations. This theorem provides the theoretical motivation for our learning objective in Section 5, which optimizes the expected task sparsity.

**Scope of Theorem 3.** We emphasize that the assumptions used to establish Theorem 3 are idealized sufficient conditions and are not intended to exactly characterize the task distribution used in our experiments. Rather, the theorem motivates task sparsity as an identifiability principle, and our experiments test whether the same inductive bias remains useful beyond these sufficient conditions. Please refer to Section C.3 for the formal theorem and its proof.

## 5 DTS: Disentanglement from Task Sparsity

Motivated by our theoretical results, we introduce DTS (Disentanglement from Task Sparsity), a framework for learning disentangled representations in which downstream tasks depend on only a small subset of latent factors, while still allowing expressive *nonlinear* task heads over the selected factors. Our key design choice is to formulate sparsity optimization as a task-conditioned mask prediction problem, replacing the non-differentiable sparsity objective with a differentiable expected sparsity of the sampled mask. We describe the sparsity objective in Section 5.1, the mutual information objective for discouraging redundant latent dimensions in Section 5.2, and the joint multi-task training of the encoder and task heads in Section 5.3.

### 5.1 Disentangled Representation Learning with Task Sparsity

**DETR encoder.** Given an input image $x$, we encode the image to a set of tokens $\{z_1, \ldots, z_K\}$ using DETR (Carion et al., 2020) encoder $F_\theta$. Specifically, the image is encoded to $K$ tokens, each represented by a $d$-dimensional binary vector:

$$z = \{z_1, \ldots, z_K\}, \qquad z_k \in \{0,1\}^d. \tag{1}$$

Each dimension of token is intended to represent an individual latent property of the corresponding object, such as "is red," "is small," or "is a sphere," if we learn a disentangled representation.

The binary representation is learned jointly with the downstream task objectives. During training, we use a straight-through gradient estimator (Bengio et al., 2013) to enable gradient-based optimization through the discrete latent variables. Specifically, given encoder probablities $\{p_k\} = \sigma(F_\theta(x)) \in [0,1]^d$, we compute

$$\hat{z}_k \sim \text{Bernoulli}(p_k),\ z_k = \text{sg}(\hat{z}_k - p_k) + p_k, \tag{2}$$

where $\text{sg}(\cdot)$ denotes the stop-gradient operator and $\sigma(\cdot)$ denotes the sigmoid function.

**Task sparsity via masking.** Different tasks depend on different subsets of the latent factors. We therefore introduce a task-conditioned mask predictor that selects the dimensions relevant to each task. Given a task description represented by a question $q = [w_1, \ldots, w_L]$, we predict a binary mask $m_q \in \{0,1\}^d$ that is shared across the tokens. Specifically, a transformer $G_\varphi$ predicts gate probability $\pi_q = G_\varphi(q) \in [0,1]^d$, and we use a straight-through estimator to obtain a binary task mask from the probability:

$$\hat{m}_q \sim \text{Bernoulli}(\pi_q),\ m_q = \text{sg}\left(\hat{m}_q - \pi_q\right) + \pi_q. \tag{3}$$

We mask out the dimensions of the tokens with the predicted mask and obtain the task-conditioned tokens $\tilde{z}_{\cdot|q} = \{\tilde{z}_{1|q}, \tilde{z}_{2|q}, \cdots, \tilde{z}_{K|q}\}$ as

$$z_{k|q} = z_k \odot m_q, \tag{4}$$

where $\odot$ denotes element-wise multiplication. Here, the non-zero elements of $m_q$ controls the sparsity of this task. We can directly optimize the expected sparsity $\mathbb{E}_{q\sim D, m_q\sim \text{Bernouii}(\pi_q)}[\|m_q\|_0]$ by minimizing the probability $\mathbb{E}_{q\sim D}[\pi_q]$ (Yamada et al., 2020):

$$\mathcal{L}_{\text{sp}} = \mathbb{E}_{q\sim D}[\pi_q]. \tag{5}$$

### 5.2 Removing the redundancy of Latent Factors

One problem that rises in the practical scenario is that, we do not know the dimension of the ground-truth representation as we did in the theoretical derivation. With slack dimensions for the representation, task sparsity alone does not prevent the representation from containing redundant or derived factors. For example, the model could represent an additional factor as

$$z_i = z_j \wedge z_k, \tag{6}$$

which may allow some tasks to be represented using fewer dimensions while making the representation redundant. While this is beneficial in terms of the sparsity, this is not desirable for OOD generalization, as it makes the latent unidentifiable from the subspace (e.g., training from $z_j = 1$ subspace makes $z_i$ and $z_k$ unidentifiable, it might use $z_i$ instead of the true variable $z_k$, and fail to generalize on the full dataset). We therefore additionally encourage each latent dimension to contain information that cannot be predicted from the other dimensions.

For each slot $z_k$, we measure the dependence between a latent dimension $z_{k,i}$ and another latent dimension $z_{k,j}$ using their mutual information:

$$I(z_{k,i}; z_{k,j}) = \sum_{a,b\in\{0,1\}} p_{ij}(a,b) \log \frac{p_{ij}(a,b)}{p_i(a)p_j(b)}. \tag{7}$$

Since the latent variables are binary, the joint distribution contains only four outcomes and the mutual information can therefore be computed in closed form. Thus, rather than introducing an additional neural network to estimate these quantities, we compute the probabilities directly from the minibatch. Given a batch of $B$ examples, the empirical distribution can be calculated as

$$p_i(a) = \frac{1}{B}\sum_{n=1}^{B} p_{k,i}^{(n)} \cdot a + (1 - p_{k,i}^{(n)})(1-a), \tag{8}$$

where $p_{k,i} = \sigma(l_{k,i})$. We denote the mutual information calculated by the minibatch statistics as $\hat{I}(z_{k,i}; z_{k,j})$. We minimize the aggregated pairwise mutual information,

$$\mathcal{L}_{\text{MI}} = \sum_{k=1}^{K}\sum_{i<j} \hat{I}(z_{k,i}; z_{k,j}). \tag{9}$$

This objective encourages latent dimensions within each slot to be statistically independent, discouraging redundant representations in which one property can be inferred from another.

### 5.3 Task objectives

DTS utilizes a general transformer head, so DTS can learn a disentangled representation from any nonlinear tasks. Akin to the VLM architectures, the inputs to the VQA prediction head are concatenation of visual tokens and language tokens

$$[\tilde{z}, q] = \tilde{z}_1, \ldots, \tilde{z}_K, w_1, \cdots, w_k, \tag{10}$$

and it learns from a task objective from the multi-task dataset $\mathcal{D}$:

$$\mathcal{L}_{\text{VQA}} = \mathbb{E}_{(x,q,y)\sim\mathcal{D}}\left[-\log p_\psi(y \mid \tilde{z}, q)\right]. \tag{11}$$

Table 1: **DTS shows better disentanglement.** We report the average of TPR (true positive rate) and TNR (true negative rate), 50% being the floor accuracy can be achieved by chance. The matching accuracy is averaged over the validation set. DTS successfully recovers a disentangled representation (98% in average), while prior approaches cannot (best baseline having 78% in average). Best scores are highlighted in magenta.

| Algorithm | Color | Shape | Size | Material | PosX | PosY | Presence | Avg. |
|---|---|---|---|---|---|---|---|---|
| Vanilla | 73 $\pm 2$ | 84 $\pm 4$ | 81 $\pm 3$ | 82 $\pm 3$ | 79 $\pm 3$ | 77 $\pm 1$ | 82 $\pm 1$ | 78 $\pm 0$ |
| $\beta$-VAE | 75 $\pm 2$ | 86 $\pm 3$ | 85 $\pm 3$ | 84 $\pm 2$ | 76 $\pm 2$ | 75 $\pm 2$ | 82 $\pm 2$ | 78 $\pm 1$ |
| TC-VAE | 80 $\pm 5$ | 80 $\pm 11$ | 88 $\pm 7$ | 83 $\pm 4$ | 62 $\pm 6$ | 64 $\pm 8$ | 76 $\pm 3$ | 72 $\pm 5$ |
| PACS | 58 $\pm 4$ | 67 $\pm 5$ | 81 $\pm 2$ | 81 $\pm 2$ | 56 $\pm 2$ | 57 $\pm 3$ | 69 $\pm 6$ | 61 $\pm 1$ |
| DTS | 100 $\pm 0$ | 96 $\pm 6$ | 100 $\pm 0$ | 100 $\pm 0$ | 97 $\pm 1$ | 97 $\pm 1$ | 94 $\pm 2$ | 98 $\pm 1$ |

We do not apply positional encoding on the visual latents and apply only at the language token embedding, as the tokens have a set structure.

We additionally include an image reconstruction objective to encourage the learned tokens to preserve sufficient information about the input. Since reconstruction is not expected to depend on only a sparse subset of latent factors, we do not apply the task-conditioned mask to this objective. Instead, we pass the feature vectors from the layer immediately preceding token sampling ( denoted as $h(x)$) to the decoder $p_\omega$ to reconstruct the input image:

$$\mathcal{L}_{\text{rec}} = \mathbb{E}_{x\sim\mathcal{D}}\left[\|p_\omega(h(x)) - x\|_2^2\right]. \tag{12}$$

The representation encoder, task-conditioned gate, and VQA prediction head are trained jointly. Given an input image $x$ and question $q$, The overall training objective is

$$\mathcal{L} = \mathcal{L}_{\text{VQA}} + \lambda_{\text{rec}}\mathcal{L}_{\text{rec}} + \lambda_{\text{sp}}\mathcal{L}_{\text{sp}} + \lambda_{\text{MI}}\mathcal{L}_{\text{MI}}. \tag{13}$$

The VQA objective ensures that the learned representation retains sufficient information to solve the auxiliary tasks, while the sparsity and independence objectives impose the structural bias required for disentanglement and OOD generalization.

## 6 Experiments

We now empirically show that DTS can learn disentangled representations from nonlinear multi-task datasets, enabling OOD generalization. First, we show that DTS is able to recover disentangled representations from nonlinear multi-task datasets, while prior disentanglement works fail to learn the representation (Section 6.1). Second, we show that this disentangled representation enables OOD generalization with sparsity-aware learning rule (Section 6.2). Finally, we analyze each components of DTS and show that its training objectives and masking architecture is crucial for achieving OOD generalization (Section 6.3). Results are aggregated over 4 seeds, and we report its mean and standard deviation.

**Baselines.** We compare DTS against disentangled representation learning methods, including $\beta$-VAE (Higgins et al., 2017), TC-VAE (Chen et al., 2018). We also compare with PACS (Fumero et al., 2023), a prior task-sparsity approach that utilized linear heads. We also compare with conventional transformers (denoted as vanilla), without masking algorithm nor binary latents.

### 6.1 Disentanglement from Nonlinear Multi-task Datasets

We first examine whether DTS is able to recover disentangled representations from the nonlinear multi-task datasets.

**Tasks and datasets.** We expand the CLEVR dataset (Johnson et al., 2017) from 70K to 1M images using the official CLEVR dataset's generation and rendering code. Each image has a resolution of $192 \times 128$ and contains between 3 and 10 objects. Each object is described by the following properties (with the number of possible values in parentheses): shape (3), color (8), posX (8), posY (8), size (2), material (2), and presence (1), together forming a 32-dimensional binary representation. From these properties, we generate over 3000 VQA tasks using the question generation templates from CLEVR (see Table 7 for detailed descriptions of the task families). We note that these VQA templates naturally exhibit task sparsity (e.g., the question "Are there an equal number of large things and spheres?" depends only on object's size and shape, but not on its color, position, material).

Table 2: **DTS shows better OOD generalization.** We report the train, validation, test accuracy, and the number of latent dimensions used for solving the tasks of each methods. Training and validation set is sampled from the subspace $U$, while the test set is sampled from the full dataset $X$. The accuracy is averaged over all 72 tasks. Best scores in test set and sparsity $|S|$ for each task are highlighted in magenta.

| Algorithm | Train | Val | Test | $\lvert S\rvert$ |
|---|---|---|---|---|
| Vanilla | 100 $\pm 0$ | 98 $\pm 0$ | 89 $\pm 1$ | 736 $\pm 109$ |
| $\beta$-VAE | 100 $\pm 0$ | 97 $\pm 1$ | 86 $\pm 4$ | 1276 $\pm 584$ |
| TC-VAE | 100 $\pm 0$ | 95 $\pm 1$ | 76 $\pm 7$ | 1627 $\pm 669$ |
| PACS | 89 $\pm 5$ | 80 $\pm 1$ | 76 $\pm 1$ | 159 $\pm 117$ |
| DTS | 100 $\pm 0$ | 97 $\pm 0$ | 93 $\pm 2$ | 7 $\pm 1$ |

**Results.** Table 1 reports the matching accuracy between the ground-truth factors and the learned latent dimension that matches that factor the best. Because the binary labels are imbalanced, we define matching accuracy as the average of the true positive rate (TPR) and true negative rate (TNR). The results show that DTS successfully recovers a disentangled representation, achieving $98\%$ matching accuracy across all property categories. In contrast, prior approaches fail to recover the underlying factors, with even the strongest baselines achieving only $78\%$ matching accuracy.

### 6.2 Generalization for downstream tasks

We now show that the learned disentangled representation transfers to OOD generalization, combined with sparsity-aware learning rule. Specifically, we train the task head on correlated datasets (e.g., all cubes are red, the object in the right is always a cube) and show its generalization to a full distribution.

**Tasks and datasets.** To evaluate OOD generalization, we generate $15$ biased downstream datasets in which different properties are correlated. Each dataset consists of $10$k images and have $4$ to $5$ downstream tasks per dataset, having $72$ tasks in total. Please refer to Table 2 for detailed descriptions of the downstream tasks. For these experiments, we do not provide the language instruction to the network. Instead, we treat each task instance as a separate downstream task and train a separate task head for each instance.

However, we found that optimizing the sparse downstream gate in the single-task setting is sensitive to local optima. To address this issue, we run the optimization with $128$ different initializations and select the solution based on its validation accuracy and sparsity. Except for PACS, which uses an optimization procedure specifically designed for linear heads, we apply this same downstream optimization procedure to all methods, including the ablations. Hence, these comparisons isolate the quality of the learned representations rather than differences in downstream optimization. Please refer to Section A.4 for details.

**Results.** Table 2 reports the training, validation, and test accuracies, with the number of latent dimensions used by each task head. The training and validation sets are sampled from the biased subspace $U$, whereas the test set is sampled from the full data distribution $X$. As shown in Table 2, DTS is the only method that achieves strong OOD generalization, with $4\%$ gap between validation and test accuracy, while the other methods exhibit larger gaps (except PACS, which severely underfits due to the limited capacity of its linear task head). Importantly, these failures are not due to overfitting (validation accuracies remain close to the training accuracy); the performance gap arises from the distributional shift from the correlated dataset to the full dataset.

### 6.3 Analysis

In this section, we analyze the components in DTS on enabling disentanglement from nonlinear multi-task dataset. First, we show that DTS's objective is crucial for OOD generalization. Moreover, we show that DTS is able to achieve OOD generalization in nonlinear downstream tasks, while prior works with linear task head (e.g., PACS) is not. We reuse the training scheme of previous experiments for representation learning and downstream task evaluation.

**Q: How do each component of DTS's objective matter for OOD generalization?**

**A:** Table 3 reports the training, validation, and test accuracies, together with the number of latent dimensions used by each task head. Removing either the MI objective or the sparsity objective from DTS significantly hinders the test accuracy. These results show that both objectives are important for learning a disentangled representation and achieving OOD generalization.

Table 3: **Ablation of the training objectives in DTS.**

| Algorithm | Train | Val | Test | $\lvert S\rvert$ |
|---|---|---|---|---|
| w/o MI | $100_{\pm 0}$ | $97_{\pm 1}$ | $91_{\pm 3}$ | $44_{\pm 24}$ |
| w/o sparsity | $100_{\pm 0}$ | $98_{\pm 0}$ | $88_{\pm 1}$ | $944_{\pm 226}$ |
| w/o MI & sparsity | $100_{\pm 0}$ | $96_{\pm 3}$ | $83_{\pm 6}$ | $2028_{\pm 559}$ |
| DTS | $100_{\pm 0}$ | $97_{\pm 0}$ | $93_{\pm 2}$ | $7_{\pm 1}$ |

**Q: Does DTS improve nonlinear task generalization over prior sparsity-aware methods?**

**A:** To answer this question, we compare our method to PACS, which learns a linear task head, in downstream OOD generalization experiments. Table 4 reports test accuracy for each task category. PACS perform better on linear tasks, with achieving higher mean accuracies. On nonlinear tasks, however, DTS substantially outperforms PACS, achieving 93% vs. 63% on average. These improvements shows that nonlinear task head of DTS is crucial for achieving OOD generalization in nonlinear tasks.

Table 4: **DTS strikes in nonlinear tasks.**

| Task | PACS | DTS |
|---|---|---|
| Linear tasks | | |
| Antecedent count | $96_{\pm 2}$ | $88_{\pm 6}$ |
| Consequent count | $99_{\pm 0}$ | $97_{\pm 2}$ |
| Nonlinear tasks | | |
| Conjunction | $65_{\pm 2}$ | $95_{\pm 4}$ |
| Negated conjunction | $71_{\pm 2}$ | $97_{\pm 2}$ |
| Spatial relation | $54_{\pm 1}$ | $87_{\pm 3}$ |

**Q: Do we need sparsity-aware learning rule in downstream tasks for OOD generalization?**

**A:** Section 4.1 shows that learning the representation alone is impossible to achieve OOD generalization, and Section 4.2 shows that we need a sparsity-aware learning rule for OOD generalization. We empirically confirm this result by evaluating DTS and baselines with and without sparsity penalties during downstream training. Table 3 reports the test accuracy under both learning rules. comparing naive learning rule and sparsity-aware learning rule. The performance of DTS deteriorates without sparsity penalties, whereas it outperforms the baselines when paired with sparsity-aware downstream learning. hese results show that learning a disentangled representation alone is insufficient to ensure OOD generalization in our setting, highlighting the importance of sparsity-aware downstream learning rule for OOD generalization from disentangled representation.

Table 5: **DTS requires sparsity-aware learning rule.**

| Algorithm | No sparsity rule | Sparsity rule |
|---|---|---|
| Vanilla | $90_{\pm 2}$ | $89_{\pm 1}$ |
| $\beta$-VAE | $86_{\pm 4}$ | $86_{\pm 4}$ |
| TC-VAE | $75_{\pm 5}$ | $76_{\pm 7}$ |
| PACS | $76_{\pm 1}$ | $76_{\pm 1}$ |
| DTS | $83_{\pm 2}$ | $93_{\pm 2}$ |

## 7 Closing Remarks

In this work, we introduced DTS, a sparsity-aware representation learning framework that discovers disentangled representations from nonlinear multi-task datasets. We first theoretically show that task sparsity can unlock OOD generalization, and we can learn a corresponding disentangled representation with multi-task learning. We realize this theoretical intuition in DTS, which formulates task sparsity as task-conditioned binary masking problem and trains a transformer task head that can be used for any nonlinear tasks. We empirically demonstrate that DTS is able to learn those representation from nonlinear multi-task dataset, and utilize the representation to generalize from correlated datasets. Our analysis shows that the architecture and the objective of DTS is crucial for learning disentangled representation from nonlinear multi-task dataset.

Several open questions remain for DTS, suggesting directions for future work. First, the evaluation of DTS is limited to synthetic VQA tasks with small transformers. Scaling up DTS to to more diverse, real-world multi-task datasets and larger models could be a promising direction. Second, DTS utilizes pairwise mutual information to encourage independence among latent coordinates, but those pairwise independence does not imply joint independence. Incorporating objectives that capture higher-order dependencies, such as total correlation regularization in TC-VAE (Chen et al., 2018), could be a natural extension. Finally, sparse downstream adaptation currently requires a multi-start optimization ( Section A.4) procedure because single-task gate optimization is sensitive to local optima, which is more expensive than a single downstream training run. Developing a more robust and efficient sparse downstream optimizer is therefore an important direction for future work.

### AI USE STATEMENT

In this work, we used generative AI tools for polishing the writing, formulating mathematical claims, and implementing the methods. We have not used generative AI tools for designing or providing feedback on research methodology or experiments, proposing or refining hypotheses, assisting with translation. Supporting qualitative and thematic data analysis and interpreting the results, and generating synthetic data sets, helping develop theoretical models or conceptual frameworks, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, cleaning and reformatting dataset, interpreting results, are not applicable to this work. The authors have manually reviewed the paper to prevent plagiarism, and read the LLM-generated code and verified and tested its correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

### REPRODUCIBILITY STATEMENT

We include implementation details needed for reproduction, such as hyperparameters (including the baselines), datasets, and evaluation protocols in Section A. We provide the full results of each tasks in Section B for better future reproducibility. Clear explanation and the proof of the theoretical results are provided in Section C.

## REFERENCES

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of machine learning research*, 3(Nov):463–482, 2002.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pp. 213–229. Springer, 2020.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, 31, 2018.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in neural information processing systems*, 29, 2016.

Pedro Domingos. Occam's two razors: The sharp and the blunt. In *KDD*, pp. 37–43, 1998.

Marco Fumero, Florian Wenzel, Luca Zancato, Alessandro Achille, Emanuele Rodolà, Stefano Soatto, Bernhard Schölkopf, and Francesco Locatello. Leveraging sparse and shared feature activations for disentangled representation learning. *Advances in Neural Information Processing Systems*, 36:27682–27698, 2023.

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.

Philip Hall. On representatives of subsets. In *Classic papers in Combinatorics*, pp. 58–62. Springer, 1987.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 9726–9735. IEEE, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 15979–15988. IEEE, 2022.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*, 2017.

Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230*, 2018.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

Paul W Holland and Roy E Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics-theory and Methods*, 6(9):813–827, 1977.

Ali Jalali, Sujay Sanghavi, Chao Ruan, and Pradeep Ravikumar. A dirty model for multi-task learning. *Advances in neural information processing systems*, 23, 2010.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pp. 4904–4916. PMLR, 2021.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2901–2910, 2017.

Youngjun Jun, Jiwoo Park, Kyobin Choo, Tae Eun Choi, and Seong Jae Hwang. Disentangling disentangled representations: Towards improved latent units via diffusion models. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 3559–3569. IEEE, 2025.

Ilyes Khemakhem, Diederik Kingma, Ricardo Monti, and Aapo Hyvarinen. Variational autoencoders and nonlinear ica: A unifying framework. In *International conference on artificial intelligence and statistics*, pp. 2207–2217. PMLR, 2020.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International conference on machine learning*, pp. 2649–2658. PMLR, 2018.

David Klindt, Lukas Schott, Yash Sharma, Ivan Ustyuzhaninov, Wieland Brendel, Matthias Bethge, and Dylan Paiton. Towards nonlinear disentanglement in natural data with temporal sparse coding. *arXiv preprint arXiv:2007.10930*, 2020.

Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. *arXiv preprint arXiv:1711.00848*, 2017.

Sébastien Lachapelle, Pau Rodriguez, Yash Sharma, Katie E Everett, Rémi Le Priol, Alexandre Lacoste, and Simon Lacoste-Julien. Disentanglement via mechanism sparsity regularization: A new principle for nonlinear ica. In *Conference on Causal Learning and Reasoning*, pp. 428–484. PMLR, 2022.

Sébastien Lachapelle, Tristan Deleu, Divyat Mahajan, Ioannis Mitliagkas, Yoshua Bengio, Simon Lacoste-Julien, and Quentin Bertrand. Synergies between disentanglement and sparsity: Generalization and identifiability in multi-task learning. In *International Conference on Machine Learning*, pp. 18171–18206. PMLR, 2023.

Sébastien Lachapelle, Pau Rodríguez López, Yash Sharma, Katie Everett, Rémi Le Priol, Alexandre Lacoste, and Simon Lacoste-Julien. Nonparametric partial disentanglement via mechanism sparsity: Sparse actions, interventions and sparse temporal dependencies. *arXiv preprint arXiv:2401.04890*, 2024.

Elliot Layne, Jason Hartford, Sébastien Lachapelle, Mathieu Blanchette, and Dhanya Sridhar. Sparsity regularization via tree-structured environments for disentangled representations. *arXiv preprint arXiv:2405.20482*, 2024.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pp. 4114–4124. PMLR, 2019.

Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises. In *International conference on machine learning*, pp. 6348–6359. PMLR, 2020a.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *Advances in neural information processing systems*, 33:11525–11538, 2020b.

Andreas Maurer, Massi Pontil, and Bernardino Romera-Paredes. Sparse coding for multitask and transfer learning. In *International conference on machine learning*, pp. 343–351. PMLR, 2013.

Gemma E Moran, Dhanya Sridhar, Yixin Wang, and David M Blei. Identifiable deep generative models via sparse decoding. *arXiv preprint arXiv:2110.10804*, 2021.

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

Robin Quessard, Thomas Barrett, and William Clements. Learning disentangled representations and group structure of dynamical environments. *Advances in Neural Information Processing Systems*, 33:19727–19737, 2020.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

Craig Saunders, Alexander Gammerman, and Volodya Vovk. Ridge regression learning algorithm in dual variables. 1998.

Simone Scardapane, Danilo Comminiello, Amir Hussain, and Aurelio Uncini. Group sparse regularization for deep neural networks. *Neurocomputing*, 241:81–89, 2017.

Rui Shu, Yining Chen, Abhishek Kumar, Stefano Ermon, and Ben Poole. Weakly supervised disentanglement with guarantees. *arXiv preprint arXiv:1910.09772*, 2019.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *International conference on machine learning*, pp. 9120–9132. PMLR, 2020.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

VN Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103, 2008.

Derrick Xin, Behrooz Ghorbani, Justin Gilmer, Ankush Garg, and Orhan Firat. Do current multi-task optimization methods in deep learning even help? *Advances in neural information processing systems*, 35:13597–13609, 2022.

Yutaro Yamada, Ofir Lindenbaum, Sahand Negahban, and Yuval Kluger. Feature selection using stochastic gates. In *International conference on machine learning*, pp. 10648–10659. PMLR, 2020.

Tao Yang, Yuwang Wang, Yan Lu, and Nanning Zheng. Visual concepts tokenization. *Advances in Neural Information Processing Systems*, 35:31571–31582, 2022.

Tao Yang, Cuiling Lan, Yan Lu, and Nanning Zheng. Diffusion model with cross attention as an inductive bias for disentanglement. *Advances in Neural Information Processing Systems*, 37: 82465–82492, 2024.

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3712–3722, 2018.

Yujia Zheng, Ignavier Ng, and Kun Zhang. On the identifiability of nonlinear ica: Sparsity and beyond. *Advances in neural information processing systems*, 35:16411–16422, 2022.

# A EXPERIMENTAL DETAILS

## A.1 IMPLEMENTATION DETAILS

**Implementation details of DTS.** We resize the image input to $128 \times 192$. The image is first processed by a ResNet-18 trunk without max-pooling (stride 16, $8 \times 12$ feature map, 512 channels). The feature vector is fed to a DETR-style decoder to produce $S = 11$ learned tokens with 6 transformer decoder layers ($d = 256$, 8 heads). Each token is mapped by a two-layer MLP (256 hidden units) to $F = 4096$-dimensional tokens, that we utilize for downstream task learning. The task loss is averaged over 4 independent Bernoulli draws of the bits and masks. For reconstruction, we use the spatial-broadcast decoder from Locatello et al. (2020b). All algorithms are trained for 60k steps with AdamW (weight decay 0.01) with learning rate of $3 \times 10^{-4}$ after a linear warmup of 6k steps. The sparsity penalty is switched on at step 10k and ramped linearly to full weight over 6k steps, the MI penalty is ramped over the first 6k steps.

**Baseline implementations.** Table 6 lists the implementation of the baselines. DTS and its three ablations uses binary latent, which first calculates the field probabilities $p = \sigma(z/\tau)$ with $\tau = 0.3$ (fixed throughout training), and sampling a binary latent $b \sim \text{Bernoulli}(p)$ with the straight-through gradient of $p$. For evaluation, we use the deterministic latent $\mathbb{1}(z > 0.5)$. For $\beta$-VAE and TC-VAE, we sample the latent from reparameterized Gaussian distribution $q(z \mid x) = \mathcal{N}(\mu, \sigma^2)$ , as in the original paper. Each method disentangles the representation via KL regularization $\text{KL}(q \,\|\, \mathcal{N}(0, I))$ and total correlation $\text{TC}(z)$, where total correlation is estimated within the minibatch. For PACS, we use the continuous latent without sampling, and use a per-task *linear* heads. However, with the paper's inner loop (30 unrolled steps), we found that algorithm becomes highly unstable with our encoder due to the correlated latents. We instead apply kernel ridge regression (Saunders et al., 1998) with ridge coefficient of 0.01, then 3 IRLS (Holland & Welsch, 1977) iterations with the regularizer $\alpha\|\varphi\|_1 + \beta \sum_m \|\varphi_{\cdot m}\|_2$ with $\alpha = \beta = 0.01$.

Table 6: **Implementation of baselines.** All algorithms share the same encoder, reconstruction loss and the optimization schedule. ST binary denotes straight-through Bernoulli sampling.

| Algorithm | Latent (per slot) | Masking | Disentanglement objectives | Batch |
|---|---|---|---|---|
| DTS (ours) | ST binary | Stochastic masking | gate $L_0$ $2\times10^{-3}$, MI $10^{-6}$ | 64 |
| w/o MI | ST binary | Stochastic masking | gate $L_0$ $2\times10^{-3}$ | 64 |
| w/o sparsity | ST binary | Stochastic masking | MI $10^{-6}$ | 64 |
| w/o MI & sparsity | ST binary | Stochastic masking | – | 128 |
| Vanilla | Continuous | none | – | 128 |
| $\beta$-VAE | Gaussian | none | KL, $\beta = 10^{-5}$ | 128 |
| TC-VAE | Gaussian | none | KL $10^{-4}$, TC $10^{-3}$ | 128 |
| PACS | Continuous | Sparse linear heads | $\alpha = \beta = 10^{-2}$ (IRLS) | 64 |

## A.2 DATASET FOR REPRESENTATION LEARNING

**Scene generation.** All Stage-1 training uses synthetic CLEVR-style scenes rendered with the CLEVR Blender pipeline and down-sampled to $128 \times 192$ RGB. A scene contains between 3 and 10 objects (mean 6.5); each object is described by a state vector

$$(\text{presence},\ \text{shape},\ \text{color},\ \text{size},\ \text{material},\ x,\ y),$$

with shape $\in \{\text{cube}, \text{sphere}, \text{cylinder}\}$, color $\in$ 8 CLEVR colors, size $\in \{\text{large}, \text{small}\}$, material $\in \{\text{rubber}, \text{metal}\}$, and image-plane position $(x, y) \in [0.2, 0.8]^2$ (continuous). Attribute values are sampled uniformly and independently per object. The dataset consists of 1M scenes and we use 5k scene for calculating the matching accuracy (Table 1). For the task programs, positions are quantized to 8 grids, so these 7 attributes can be described $1 + 8 + 3 + 2 + 2 + 8 + 8 = 32$ values.

**Task generation.** Task is generated from a bank of 3072 functional programs instantiated from the CLEVR question DSL (Table **??**). Programs are built from *conjunctive filters* $f = \bigwedge_{a \in A_f} [\text{attr}_a = v_a]$ over the six attributes, each filter holding one or two literals. A task carries a primary filter $f_1$

(1–2 literals) and, depending on the family, a second filter $f_2$ (0 or 1 literals, applied to `hop` and `same` families) or a base filter $f_B$ (1 or 2 literals, the second operand of `or` and `cmp` families), a spatial relation, a queried attribute, or a comparison operator. We restrict each task to be depend at most three distinct attributes, thus depending on at most 24 values (among 32 values) from the disentangled representation.

Table 7: **Twelve CLEVR-program task families in the task bank.** Adapted from the CLEVR question generation templates, each task is a functional program over object attributes (*color*, *shape*, *size*, *material*) and grid positions (*posX*, *posY*). Filters constrain up to `max_attrs`$= 3$ attributes. A bank of 3072 instances is sampled (uniformly across families).

| Family | Description | Answer | # tasks | Example question |
|---|---|---|---|---|
| **Zero-hop**: filter → count / exist / query | | | | |
| `zh_count` | Count objects matching a filter | int 0–10 | 263 | *How many small spheres are there?* |
| `zh_exist` | Whether any object matches the filter | yes/no | 262 | *Are there any large spheres?* |
| `zh_query` | Attribute of the unique matching object | attr. val | 250 | *What size is the purple object?* |
| **One-hop**: anchor → relate (direction) → filter | | | | |
| `hop_count` | Count filtered objects in a direction from a unique anchor | int 0–10 | 241 | *How many objects are below the green object?* |
| `hop_exist` | Whether such a related object exists | yes/no | 277 | *Is there a small object below the purple object?* |
| `hop_query` | Attribute of the unique related object | attr. val | 271 | *What shape is the object below the cylinder?* |
| **Same-attribute**: anchor → same-attr set → filter | | | | |
| `same_count` | Among objects sharing an attribute with the anchor, count matches | int 0–10 | 255 | *How many cylinders have the same color as the gray cube?* |
| `same_exist` | Whether such a same-attribute object exists | yes/no | 233 | *Is there a brown object with the same shape as the gray cube?* |
| **Disjunction**: union of two filters | | | | |
| `or_count` | Count objects matching filter A or B | int 0–10 | 248 | *How many objects are brown spheres or gray spheres?* |
| `or_exist` | Whether any object matches A or B | yes/no | 255 | *Are there any gray cubes or spheres?* |
| **Comparison**: compare two subprograms | | | | |
| `cmp_int` | Compare two object counts (more / fewer / equal) | yes/no | 275 | *Are there more cylinders than small objects?* |
| `cmp_attr` | Whether two unique anchors share an attribute value | yes/no | 242 | *Do the yellow cylinder and the yellow object have the same color?* |
| **Total** | | | **3072** | |

### A.3 Dataset for Downstream Task Learning

**Correlation datasets.** We generate 15 datasets of 10k scenes each, to satisfy a spurious correlation (Table 8). Ten datasets cover the attribute pairs of the ontology in both directions (shape → material, shape → size, color → shape, color → size, size → material, material → color, size → shape, color → material) and the two continuous attributes (shape → position, color → position); `multi2` and `multi3` impose two and three simultaneous rules; `cube_region` is a two-way rule (cube ⇔ a

fixed position region); and `cube_gray` and `cube_red` are the shape $\rightarrow$ color rules found adversarial for sparse codes in preliminary experiments. Each dataset is split into 8k training and 2k validation scenes, with a test set of 2k scenes from the unconstrained distribution, where the rule no longer holds.

**Tasks.** For every dataset we instantiate five tasks from a fixed template (Table 8): (i) *antecedent count*, (ii) *consequent count*, (iii) antecedent with an unrelated literal, (iv) *negated conjunction*, and (v) a *spatial relation*. Tasks (i), (ii) are linear in the per-object literals, while (iii), (iv) and (v) are not. Together, we utilize 72 (dataset, task) pairs, 25 linear tasks and 47 nonlinear tasks.

Table 8: **Correlated datasets and their downstream tasks**. Every task is to count the number of objects that satsifies the condition stated in the table.

| Dataset | Rules | Antecedent | Consequent | Conjunction | Negated | Relation |
|---|---|---|---|---|---|---|
| `sphere_metal` | sphere $\rightarrow$ metal | sphere | metal | small sphere | sphere $\wedge\neg$red | left of sphere |
| `cyl_small` | cylinder $\rightarrow$ small | cylinder | small | metal cylinder | cylinder $\wedge\neg$blue | left of cylinder |
| `red_cube` | red $\rightarrow$ cube | red | cube | small red | red $\wedge\neg x{=}0$ | left of red |
| `blue_large` | blue $\rightarrow$ large | blue | large | metal blue | blue $\wedge\neg$cube | left of blue |
| `small_rubber` | small $\rightarrow$ rubber | small | rubber | small cube | small $\wedge\neg$green | left of small |
| `metal_purple` | metal $\rightarrow$ purple | metal | purple | metal sphere | metal $\wedge\neg$sphere | left of metal |
| `cube_left` | cube $\rightarrow$ left half | cube | – | small cube | cube $\wedge\neg$red | left of cube |
| `green_front` | green $\rightarrow$ front half | green | – | small green | green $\wedge\neg$cylinder | left of green |
| `large_cyl` | large $\rightarrow$ cylinder | large | cylinder | red large | large $\wedge\neg$yellow | left of large |
| `yellow_metal` | yellow $\rightarrow$ metal | yellow | metal | small yellow | yellow $\wedge\neg$sphere | left of yellow |
| `multi2` | sphere $\rightarrow$ blue; large $\rightarrow$ metal | sphere; large | – | large sphere | sphere $\wedge\neg x{=}0$ | left of sphere |
| `multi3` | cube $\rightarrow$ gray; small $\rightarrow$ metal; red $\rightarrow$ left | cube; small; red | – | small cube | cube $\wedge\neg$small | left of cube |
| `cube_gray` | cube $\rightarrow$ gray | cube | gray | small metal cube | cube $\wedge\neg$small | left of cube |
| `cube_region` | cube $\Leftrightarrow$ ($x{=}7$ or $y{=}0$) | cube | – | small cube | cube $\wedge\neg$red | left of cube |
| `cube_red` | cube $\rightarrow$ red | cube | red | small cube | cube $\wedge\neg$small | left of cube |

### A.4 Algorithm for Downstream Task Learning

To tackle the local optima problem of our method in single-task setting, we search over various initialization and choose the best one with its sparsity and validation accuracy (we do not use the test set). Specifically, We search with a population of 128 readouts trained jointly, each with a learnable stochastic binary gate over the fields (straight-through Bernoulli, gate probabilities initialized at 0.5, gate learning rate 0.03) and a sparsity penalty $\lambda \in \{2, 4, 8, 16, 32, 64, 128, 256\} \times 10^{-4}$, 16 random initializations per $\lambda$. For the gated codes, we prune out the dimensions that never had non-zero masking during representation learning.

For vanilla, $\beta$-VAE, TC-VAE, DTS, and all DTS ablations, we freeze the learned representation and use exactly the same downstream architecture and optimization procedure. Thus, differences among these methods isolate the learned representation. PACS is the sole exception because its method specifically requires a sparse linear predictor. We use an adapted solver for PACS because its original unrolled inner-loop optimization is unstable in our CLEVR setting; our replacement provides a stronger PACS baseline.

## B Complete Numerical Results

For completeness, we provide the full per-task results for Tables 2, 3 and 5 in Tables 9 to 11. The results are averaged over 4 seeds and we report the test accuracy and the size of the dependent factors for each tasks. We highlight the number with the best test accuracy.

Table 9: **Complete results on downstream tasks.**

| Dataset | Task | Vanilla | $\beta$-VAE | TC-VAE | PACS | DTS |
|---|---|---|---|---|---|---|
| sphere_metal | sphere | 97(12) | 83(1034) | 78(1032) | 100(1) | 98(5) |
| | metal obj. | 98(16) | 98(2059) | 96(1034) | 100(5) | 98(6) |
| | small sphere | 96(14) | 91(1034) | 95(7) | 65(1026) | 99(6) |
| | sphere ¬red | 97(16) | 78(20) | 66(2053) | 80(5) | 99(6) |
| | left of sphere | 94(3079) | 88(2060) | 81(4096) | 51(1025) | 90(9) |
| cyl_small | cyl | 96(1034) | 85(1038) | 63(2054) | 99(1) | 98(5) |
| | small obj. | 99(18) | 99(2057) | 96(2054) | 100(6) | 98(6) |
| | metal cyl | 96(15) | 96(15) | 76(12) | 61(2) | 99(6) |
| | cyl ¬blue | 98(18) | 92(2057) | 63(3074) | 81(4) | 98(7) |
| | left of cyl | 95(3079) | 87(2060) | 77(4096) | 49(1) | 89(10) |
| red_cube | red obj. | 97(9) | 98(12) | 92(1030) | 100(1) | 99(4) |
| | cube | 98(14) | 97(2056) | 95(1034) | 99(1) | 97(4) |
| | small red obj. | 95(10) | 97(12) | 94(1029) | 78(2) | 99(5) |
| | red obj. ¬x-bin 0 | 97(11) | 94(16) | 92(2053) | 93(1) | 98(5) |
| | left of red | 95(4096) | 90(4096) | 84(4096) | 60(4) | 91(32) |
| blue_large | blue obj. | 99(8) | 97(10) | 93(1029) | 100(1) | 99(3) |
| | large obj. | 99(14) | 99(2056) | 97(10) | 100(1) | 99(6) |
| | blue metal obj. | 98(9) | 98(10) | 95(1028) | 79(2) | 99(5) |
| | blue obj. ¬cube | 97(8) | 95(11) | 86(1029) | 79(2) | 99(5) |
| | left of blue | 95(3079) | 89(3079) | 87(4096) | 60(4) | 91(8) |
| small_rubber | small obj. | 90(18) | 88(2057) | 62(11) | 99(3) | 98(6) |
| | rubber obj. | 95(20) | 91(3078) | 87(1036) | 98(4) | 97(5) |
| | small cube | 96(18) | 96(1037) | 86(8) | 62(2) | 97(6) |
| | small obj. ¬green | 95(1038) | 86(1042) | 68(2055) | 75(5) | 98(7) |
| | left of small | 95(3078) | 90(2058) | 89(4096) | 63(2048) | 93(9) |
| metal_purple | metal obj. | 54(1033) | 61(2057) | 48(1030) | 48(5) | 98(5) |
| | purple obj. | 36(14) | 49(22) | 58(1030) | 100(1) | 99(6) |
| | metal sphere | 91(11) | 88(11) | 79(2052) | 48(4) | 99(7) |
| | metal obj. ¬sphere | 86(14) | 74(3076) | 57(2053) | 24(4) | 78(7) |
| | left of metal | 93(2060) | 89(3076) | 82(4096) | 59(1025) | 90(8) |
| cube_left | cube | 86(14) | 84(1036) | 61(10) | 99(1) | 97(5) |
| | small cube | 92(16) | 87(1035) | 74(9) | 62(2) | 97(6) |
| | cube ¬red | 86(1034) | 82(1038) | 69(1031) | 77(3) | 97(6) |
| | left of cube | 77(2058) | 76(1038) | 70(3076) | 51(1) | 75(29) |
| green_front | green obj. | 96(8) | 97(11) | 96(1028) | 100(1) | 99(4) |
| | small green obj. | 97(9) | 94(11) | 96(1028) | 78(2) | 99(5) |
| | green obj. ¬cyl | 96(8) | 95(10) | 95(1030) | 78(1025) | 99(6) |
| | left of green | 95(3081) | 88(3080) | 87(4096) | 59(6) | 92(9) |
| large_cyl | large obj. | 89(12) | 82(1034) | 71(8) | 100(1) | 99(4) |
| | cyl | 86(18) | 73(1038) | 78(1034) | 99(1) | 84(6) |
| | large red obj. | 96(9) | 90(9) | 95(1028) | 74(1) | 100(5) |
| | large obj. ¬yellow | 97(15) | 80(1038) | 93(2053) | 70(4) | 99(7) |
| | left of large | 95(4096) | 89(3076) | 88(4096) | 68(1025) | 90(8) |
| yellow_metal | yellow obj. | 98(9) | 98(10) | 97(5) | 100(1) | 99(3) |
| | metal obj. | 98(19) | 99(1039) | 96(1034) | 100(5) | 98(5) |
| | small yellow obj. | 97(9) | 98(11) | 98(1028) | 78(2) | 99(5) |
| | yellow obj. ¬sphere | 98(10) | 95(12) | 93(8) | 78(2) | 99(5) |
| | left of yellow | 96(4096) | 91(3078) | 89(4096) | 59(8) | 92(9) |
| multi2 | sphere | 80(11) | 77(1035) | 57(1031) | 100(1) | 78(5) |
| | large obj. | 95(13) | 81(1033) | 78(8) | 100(1) | 99(4) |
| | large sphere | 78(10) | 80(9) | 74(6) | 64(4) | 100(6) |
| | sphere ¬x-bin 0 | 85(1034) | 67(2057) | 42(3074) | 71(3) | 95(8) |
| | left of sphere | 93(3078) | 86(4096) | 64(4096) | 50(1025) | 75(9) |
| multi3 | cube | 71(13) | 78(2056) | 48(10) | 98(1) | 62(5) |
| | small obj. | 90(1036) | 93(3076) | 79(1032) | 99(2) | 95(5) |
| | small cube | 77(13) | 85(2053) | 54(9) | 56(3) | 71(8) |
| | cube ¬small | 86(9) | 92(11) | 81(1030) | 61(4) | 99(6) |
| | left of cube | 91(2060) | 85(3078) | 64(4096) | 47(2) | 89(8) |
| cube_gray10k | cube | 79(12) | 77(2054) | 45(1033) | 98(1) | 55(6) |
| | gray obj. | 64(14) | 78(1037) | 96(9) | 100(1) | 99(6) |
| | small metal cube | 81(12) | 89(13) | 72(2052) | 59(2) | 99(7) |
| | cube ¬small | 90(9) | 93(12) | 84(1030) | 60(4) | 99(5) |
| | left of cube | 91(2060) | 85(3078) | 60(4096) | 51(1) | 90(9) |
| cube_region | cube | 66(7) | 61(10) | 51(1030) | 99(1) | 81(7) |
| | small cube | 69(12) | 73(10) | 61(11) | 60(6) | 91(7) |
| | cube ¬red | 71(12) | 63(15) | 53(1033) | 79(4) | 96(7) |
| | left of cube | 70(4096) | 53(3078) | 46(4096) | 32(3076) | 71(10) |
| cube_red | cube | 77(13) | 77(2055) | 31(2053) | 99(1) | 60(6) |
| | red obj. | 80(14) | 78(18) | 96(10) | 100(1) | 100(5) |
| | small cube | 78(14) | 83(1035) | 53(1030) | 55(3) | 72(7) |
| | cube ¬small | 88(9) | 89(11) | 78(2052) | 56(4) | 100(6) |
| | left of cube | 92(3079) | 84(1041) | 59(4096) | 50(3) | 89(9) |

Table 10: **Complete results on downstream tasks for ablations.**

| Dataset | Task | DTS | w/o MI | w/o sparsity | w/o MI, sparsity |
|---|---|---|---|---|---|
| sphere_metal | sphere | 98 (5) | 98 (32) | 97 (917) | 88 (1548) |
| | metal obj. | 98 (6) | 97 (88) | 99 (1780) | 94 (3094) |
| | small sphere | 99 (6) | 98 (33) | 98 (513) | 96 (1527) |
| | sphere ¬red | 99 (6) | 98 (33) | 96 (918) | 88 (2331) |
| | left of sphere | 90 (9) | 90 (65) | 92 (423) | 89 (2294) |
| cyl_small | cyl | 98 (5) | 96 (34) | 96 (920) | 90 (2330) |
| | small obj. | 98 (6) | 97 (88) | 99 (969) | 95 (3094) |
| | metal cyl | 99 (6) | 98 (62) | 99 (1372) | 96 (1528) |
| | cyl ¬blue | 98 (7) | 97 (63) | 97 (515) | 90 (3094) |
| | left of cyl | 89 (10) | 90 (89) | 91 (18) | 88 (3094) |
| red_cube | red obj. | 99 (4) | 99 (3) | 99 (8) | 97 (779) |
| | cube | 97 (4) | 96 (88) | 98 (918) | 94 (2293) |
| | small red obj. | 99 (5) | 99 (6) | 99 (9) | 97 (778) |
| | red obj. ¬x-bin 0 | 98 (5) | 97 (88) | 96 (1780) | 94 (2292) |
| | left of red | 91 (32) | 90 (89) | 93 (1328) | 90 (3094) |
| blue_large | blue obj. | 99 (3) | 99 (4) | 99 (6) | 97 (779) |
| | large obj. | 99 (6) | 98 (32) | 99 (1279) | 96 (1528) |
| | blue metal obj. | 99 (5) | 99 (6) | 98 (1780) | 98 (777) |
| | blue obj. ¬cube | 99 (5) | 99 (6) | 99 (9) | 98 (779) |
| | left of blue | 91 (8) | 91 (66) | 93 (1327) | 91 (3094) |
| small_rubber | small obj. | 98 (6) | 96 (88) | 90 (1780) | 90 (3094) |
| | rubber obj. | 97 (5) | 93 (88) | 97 (1372) | 77 (3094) |
| | small cube | 97 (6) | 97 (63) | 97 (875) | 95 (2330) |
| | small obj. ¬green | 98 (7) | 94 (36) | 95 (1377) | 89 (3094) |
| | left of small | 93 (9) | 92 (41) | 93 (421) | 91 (1552) |
| metal_purple | metal obj. | 98 (5) | 77 (88) | 40 (1376) | 40 (2292) |
| | purple obj. | 99 (6) | 94 (88) | 99 (922) | 85 (3094) |
| | metal sphere | 99 (7) | 97 (33) | 88 (967) | 89 (1546) |
| | metal obj. ¬sphere | 78 (7) | 96 (36) | 84 (514) | 44 (3094) |
| | left of metal | 90 (8) | 92 (41) | 93 (514) | 89 (3094) |
| cube_left | cube | 97 (5) | 81 (60) | 90 (921) | 63 (3094) |
| | small cube | 97 (6) | 87 (88) | 91 (824) | 81 (3094) |
| | cube ¬red | 97 (6) | 75 (88) | 85 (1376) | 66 (3094) |
| | left of cube | 75 (29) | 77 (12) | 77 (14) | 75 (1530) |
| green_front | green obj. | 99 (4) | 97 (33) | 99 (6) | 95 (778) |
| | small green obj. | 99 (5) | 98 (35) | 99 (417) | 98 (7) |
| | green obj. ¬cyl | 99 (6) | 99 (7) | 98 (416) | 98 (7) |
| | left of green | 92 (9) | 88 (66) | 93 (1326) | 89 (3094) |
| large_cyl | large obj. | 99 (4) | 98 (4) | 90 (414) | 87 (784) |
| | cyl | 84 (6) | 81 (34) | 98 (1780) | 72 (3094) |
| | large red obj. | 100(5) | 99 (5) | 98 (1780) | 98 (778) |
| | large obj. ¬yellow | 99 (7) | 97 (34) | 89 (417) | 89 (786) |
| | left of large | 90 (8) | 90 (40) | 93 (1377) | 89 (1532) |
| yellow_metal | yellow obj. | 99 (3) | 99 (4) | 99 (964) | 98 (780) |
| | metal obj. | 98 (5) | 97 (60) | 99 (922) | 96 (3094) |
| | small yellow obj. | 99 (5) | 99 (6) | 99 (919) | 98 (779) |
| | yellow obj. ¬sphere | 99 (5) | 99 (6) | 99 (7) | 98 (779) |
| | left of yellow | 92 (9) | 91 (42) | 94 (1328) | 91 (2297) |
| multi2 | sphere | 78 (5) | 92 (34) | 71 (915) | 72 (1547) |
| | large obj. | 99 (4) | 98 (6) | 90 (1324) | 79 (2292) |
| | large sphere | 100(6) | 99 (6) | 82 (915) | 91 (780) |
| | sphere ¬x-bin 0 | 95 (8) | 80 (88) | 57 (1780) | 59 (2292) |
| | left of sphere | 75 (9) | 90 (66) | 92 (1780) | 86 (3094) |
| multi3 | cube | 62 (5) | 68 (62) | 56 (1371) | 45 (2292) |
| | small obj. | 95 (5) | 96 (88) | 90 (1325) | 86 (3094) |
| | small cube | 71 (8) | 71 (36) | 54 (1780) | 62 (3094) |
| | cube ¬small | 99 (6) | 99 (6) | 94 (413) | 90 (780) |
| | left of cube | 89 (8) | 89 (64) | 90 (873) | 82 (3094) |
| cube_gray10k | cube | 55 (6) | 72 (34) | 33 (1780) | 39 (3094) |
| | gray obj. | 99 (6) | 96 (62) | 99 (917) | 91 (3094) |
| | small metal cube | 99 (7) | 89 (9) | 91 (14) | 96 (781) |
| | cube ¬small | 99 (5) | 99 (6) | 94 (464) | 89 (780) |
| | left of cube | 90 (9) | 89 (90) | 91 (426) | 81 (3094) |
| cube_region | cube | 81 (7) | 68 (34) | 61 (1324) | 44 (2324) |
| | small cube | 91 (7) | 84 (11) | 67 (1780) | 69 (1546) |
| | cube ¬red | 96 (7) | 85 (10) | 68 (921) | 60 (780) |
| | left of cube | 71 (10) | 52 (90) | 54 (1780) | 50 (3094) |
| cube_red | cube | 60 (6) | 69 (5) | 52 (870) | 51 (1526) |
| | red obj. | 100(5) | 90 (62) | 99 (916) | 92 (2292) |
| | small cube | 72 (7) | 68 (7) | 54 (1780) | 84 (1528) |
| | cube ¬small | 100(6) | 99 (7) | 96 (8) | 89 (780) |
| | left of cube | 89 (9) | 90 (90) | 90 (424) | 83 (3094) |

Table 11: **Complete results on downstream tasks without sparsity-aware learning rule.**

| Dataset | Task | Vanilla | $\beta$-VAE | TC-VAE | PACS | DTS |
|---|---|---|---|---|---|---|
| sphere_metal | sphere | 94 | 87 | 70 | 100 | 78 |
| | metal obj. | 99 | 99 | 95 | 100 | 99 |
| | small sphere | 98 | 94 | 86 | 63 | 88 |
| | sphere ¬red | 95 | 88 | 74 | 78 | 82 |
| | left of sphere | 94 | 89 | 81 | 48 | 86 |
| cyl_small | cyl | 97 | 91 | 65 | 99 | 89 |
| | small obj. | 99 | 99 | 92 | 100 | 99 |
| | metal cyl | 99 | 98 | 83 | 58 | 97 |
| | cyl ¬blue | 98 | 93 | 68 | 78 | 90 |
| | left of cyl | 95 | 88 | 77 | 45 | 89 |
| red_cube | red obj. | 98 | 99 | 93 | 100 | 97 |
| | cube | 98 | 98 | 94 | 99 | 97 |
| | small red obj. | 99 | 99 | 95 | 77 | 99 |
| | red obj. ¬x-bin 0 | 96 | 97 | 92 | 90 | 92 |
| | left of red | 95 | 90 | 84 | 59 | 91 |
| blue_large | blue obj. | 98 | 99 | 96 | 100 | 97 |
| | large obj. | 100 | 99 | 97 | 100 | 99 |
| | blue metal obj. | 99 | 99 | 97 | 77 | 99 |
| | blue obj. ¬cube | 99 | 99 | 95 | 79 | 98 |
| | left of blue | 96 | 90 | 87 | 59 | 90 |
| small_rubber | small obj. | 95 | 88 | 76 | 100 | 93 |
| | rubber obj. | 94 | 91 | 88 | 99 | 98 |
| | small cube | 98 | 96 | 88 | 58 | 95 |
| | small obj. ¬green | 95 | 88 | 71 | 76 | 94 |
| | left of small | 96 | 91 | 89 | 62 | 90 |
| metal_purple | metal obj. | 68 | 59 | 25 | 81 | 14 |
| | purple obj. | 66 | 54 | 71 | 99 | 88 |
| | metal sphere | 95 | 81 | 75 | 59 | 90 |
| | metal obj. ¬sphere | 80 | 74 | 47 | 47 | 34 |
| | left of metal | 94 | 90 | 82 | 51 | 76 |
| cube_left | cube | 81 | 84 | 69 | 99 | 89 |
| | small cube | 92 | 92 | 80 | 59 | 94 |
| | cube ¬red | 84 | 84 | 72 | 77 | 91 |
| | left of cube | 77 | 72 | 70 | 51 | 79 |
| green_front | green obj. | 96 | 98 | 96 | 100 | 98 |
| | small green obj. | 99 | 99 | 98 | 77 | 99 |
| | green obj. ¬cyl | 98 | 98 | 97 | 78 | 99 |
| | left of green | 95 | 89 | 87 | 58 | 90 |
| large_cyl | large obj. | 97 | 80 | 78 | 100 | 38 |
| | cyl | 92 | 66 | 74 | 99 | 92 |
| | large red obj. | 99 | 95 | 95 | 76 | 99 |
| | large obj. ¬yellow | 98 | 81 | 83 | 75 | 45 |
| | left of large | 95 | 89 | 88 | 65 | 76 |
| yellow_metal | yellow obj. | 99 | 99 | 98 | 100 | 98 |
| | metal obj. | 99 | 100 | 96 | 100 | 99 |
| | small yellow obj. | 99 | 99 | 98 | 75 | 99 |
| | yellow obj. ¬sphere | 99 | 99 | 98 | 77 | 98 |
| | left of yellow | 96 | 91 | 89 | 58 | 91 |
| multi2 | sphere | 81 | 53 | 41 | 91 | 44 |
| | large obj. | 99 | 86 | 76 | 100 | 65 |
| | large sphere | 96 | 84 | 69 | 58 | 59 |
| | sphere ¬x-bin 0 | 84 | 53 | 36 | 67 | 52 |
| | left of sphere | 93 | 86 | 64 | 42 | 76 |
| multi3 | cube | 59 | 74 | 34 | 92 | 51 |
| | small obj. | 95 | 92 | 65 | 100 | 87 |
| | small cube | 70 | 78 | 49 | 57 | 69 |
| | cube ¬small | 94 | 92 | 75 | 61 | 80 |
| | left of cube | 91 | 84 | 64 | 44 | 81 |
| cube_gray10k | cube | 60 | 68 | 27 | 90 | 47 |
| | gray obj. | 87 | 77 | 91 | 100 | 98 |
| | small metal cube | 88 | 86 | 67 | 59 | 95 |
| | cube ¬small | 96 | 91 | 71 | 61 | 90 |
| | left of cube | 92 | 84 | 60 | 46 | 85 |
| cube_region | cube | 58 | 78 | 52 | 95 | 69 |
| | small cube | 65 | 80 | 64 | 59 | 85 |
| | cube ¬red | 60 | 74 | 55 | 75 | 75 |
| | left of cube | 70 | 54 | 46 | 33 | 64 |
| cube_red | cube | 55 | 64 | 28 | 83 | 51 |
| | red obj. | 88 | 87 | 93 | 100 | 99 |
| | small cube | 74 | 73 | 45 | 57 | 80 |
| | cube ¬small | 95 | 89 | 67 | 61 | 84 |
| | left of cube | 92 | 85 | 59 | 43 | 80 |

# C PROOFS

Here, we prove the main theorems of the paper. Throughout the proof, we consider a binary representation $z \in \{0,1\}^d$. For the proof, we define few definitions. For $j \in [d]$ write $z \oplus e_j$ for $z$ with coordinate $j$ flipped, and for $S \subseteq [d]$ write $z_S \in \{0,1\}^{|S|}$ for the restriction of $z$ to the coordinates in $S$.

We define a *variable support* of a function, which indicates the index of the subset that the function is dependent on:

**Definition C.1** (Variable support of a function)**.** *For $f : \{0,1\}^d \to \mathbb{R}$,*

$$\operatorname{vsupp}(f) \;=\; \big\{\, j \in [d] \;:\; \exists\, z \in \{0,1\}^d \text{ with } f(z) \neq f(z \oplus e_j) \,\big\}.$$

The variable support coincides with the sparsity of a function on the full hypercube (Lemma C.1 below), but the two can differ when the representation does not realize every pattern. For example, if $z = (w_1, w_2, w_1 \wedge w_2)$, the task $f = z_1 + z_2$ is 2-sparse, but on the realized patterns it can equally be written as $(z_1 \vee z_2) + z_3$, whose variable support is $\{1,2,3\}$: on a redundant support the variable support depends on the chosen representation of the task. Full support of the representation (Assumption 1) removes this ambiguity.

**Lemma C.1** (Variable support is a minimal support)**.** *Let $f : \{0,1\}^d \to \mathbb{R}$. Then $f$ is a function of $z_A$ if and only if $A \supseteq \operatorname{vsupp} f$. Consequently the minimal such $A$ is unique and equals* $\operatorname{vsupp} f$.

*Proof.* ($\Leftarrow$) Let $A \supseteq \operatorname{vsupp} f$ and let $z'$ agree with $z$ in $A$ (i.e., $z_A = z'_A$). Let $j_1, \ldots, j_r$ be the coordinates on which $z$ and $z'$ differ; all lie outside $A$, hence outside $\operatorname{vsupp} f$, so $f(u) = f(u \oplus e_{j_t})$ for *every* $u$ and every $t$. Flipping $j_1, \ldots, j_r$ in turn yields a path $z = z^{(0)}, z^{(1)}, \ldots, z^{(r)} = z'$ along which $f$ is constant, leading to $f(z) = f(z')$. Thus $f$ is a function of $z_A$.

($\Rightarrow$) If $f(z) = g(z_A)$ and $j \notin A$, then $f(z) = f(z \oplus e_j)$ for all $z$, implying $j \notin \operatorname{vsupp}(f)$. Therefore, $\operatorname{vsupp}(f) \subseteq A$. □

With the variable support and Lemma C.1 in hand, we now formally define *task sparsity* under a representation $\varphi$.

**Definition C.2** (Sparsity of a task under a representation)**.** *Let $T$ be a task with labels $y = f_T(z^\star)$, $z^\star = \varphi^\star(x)$, and let $z = \varphi(x) \in \{0,1\}^d$ be a representation that is sufficient for $T$, i.e., some head $g : \{0,1\}^d \to \mathbb{R}$ satisfies $g(\varphi(x)) = f_T(\varphi^\star(x))$ for all $x$. The* sparsity of $T$ under $\varphi$ *is the smallest number of coordinates of $z$ that such a head must use:*

$$s_\varphi(T) := \min\big\{\, |\operatorname{vsupp} g| \;:\; g \circ \varphi = f_T \circ \varphi^\star \big\}.$$

*Under the ground-truth representation, $s_{\varphi^\star}(T) = |\operatorname{vsupp} f_T|$, the number of latent factors the task depends on.*

The sparsity of a task depends on the representation. For instance, the task $f_T = z_1^\star \oplus z_2^\star$ has $s_{\varphi^\star}(T) = 2$, but under the representation $z = (z_1^\star,\, z_1^\star \oplus z_2^\star,\, z_3^\star)$ the head $g(z) = z_2$ computes it, so $s_\varphi(T) = 1$; conversely, the task $z_2^\star = z_1 \oplus z_2$ becomes denser under this representation. Theorem 3 asserts that, for suitable task distributions, the average $\mathcal{L}(\varphi)$ is nonetheless minimized only by representations equivalent to $\varphi^\star$.

### C.1 Proof of Theorem 1

We prove Theorem 1 by explicitly constructing, for any $k$-sparse $f$, a labeled subspace $U$ that satisfies Assumptions 1 and 2, together with a task $f'$ that is not distinguishable from $f$ on $U$.

**Theorem C.1** (Impossibility without task structure)**.** *Let $k < d$ and let $f$ be a $k$-sparse task with $\operatorname{vsupp} f = S$, $|S| = k$, w.l.o.g. $S = [k]$. Suppose every pattern of $z^\star_{[d]\setminus\{k+1\}}$ is realized on the slice $\{z^\star_{k+1} = 0\}$ and $p(z^\star_{k+1} = 1) > 0$ (both implied by full support). Set*

$$U := \{x \in X : z^\star_{k+1}(x) = 0\}, \qquad f'(z) := f(z) + z_{k+1}.$$

*Then:*

- *(a) **(Sparse identifiability holds.)** The pair $(f, U)$ satisfies Assumption 2: any $k$-sparse predictor agreeing with $f$ on $U$ equals $f$ on $\mathcal{Z}$.*
- *(b) **(Unrestricted learning fails.)** $f' = f$ on $U$; $f' \neq f$ exactly on $\{z^\star_{k+1} = 1\}$, an event of positive probability; and $\operatorname{vsupp} f' = S \cup \{k+1\}$, so $f'$ is not $k$-sparse.*
- *(c) Consequently, labeled samples from $U$ have the same distribution under $f$ and $f'$, so no predictor chosen from the unrestricted class of all functions on the basis of labeled samples in $U$ can determine $f$ on $X$.*

*Proof.* (a) Let $g$ be $k$-sparse with $g = f$ on $U$, and set $\delta := g - f$. Since every pattern of the remaining coordinates is realized on the slice, $\delta$ vanishes identically on $\{z : z_{k+1} = 0\}$ as a function. Suppose $g \neq f$, and pick $w \in \mathcal{Z}$ with $g(w) \neq f(w)$; necessarily $w_{k+1} = 1$. Then

$$g(w \oplus e_{k+1}) = f(w \oplus e_{k+1}) = f(w) \neq g(w),$$

hence $k+1 \in \operatorname{vsupp} g$. Moreover, every $i \in S$ remains in $\operatorname{vsupp} g$, as $g = f$ on $U$ and $\operatorname{supp} f = S$. Therefore $|\operatorname{vsupp} g| \geq |S| + 1 = k + 1$, contradicting by Lemma C.1 that $g$ is $k$-sparse. Therefore, $g = f$.

(b) Agreement on $U$ and disagreement on $\{z^\star_{k+1} = 1\}$ are immediate from the definition, since $z_{k+1} \equiv 0$ on $U$. Flipping coordinate $k+1$ changes $f'$ by $\pm 1$ at every point, and flipping $i \in S$ at a point of the slice changes $f'$ exactly as it changes $f$; hence $\operatorname{vsupp} f' = S \cup \{k+1\}$, and $f'$ depends on $k+1$ variables, so $f'$ is not $k$-sparse by Lemma C.1.

(c) By (a), the labels $y = f(z^\star(x))$ and $y' = f'(z^\star(x))$ coincide for every $x \in U$, so the learning rule receives identical data for two tasks and outputs the same predictor. By (b), that predictor fails to predict on at least one of the two tasks. □

### C.2 Proof of Theorem 2

The proof of Theorem 2 is immediate from Assumption 2.

**Theorem C.2** (OOD generalization under Sparse Representations)**.** *Let $z^\star = \varphi^\star(x)$ denote the latent representation satisfying Assumption 1. Suppose the downstream task is generated as*

$$y = f(z^\star),$$

*where $f$ is $k$-sparse and $x$ is sampled from subspace $U$ following Assumption 2.*

*Then the learning rule that finds the minimal variable support among the predictors consistent with the labeled samples in $U$, i.e.,*

$$\hat{f} \in \arg\min_{\hat{f}} |\operatorname{vsupp} \hat{f}| \quad \text{s.t. } y = \hat{f}(\varphi^\star(x)) \ \ \forall x \in U,$$

*uniquely recovers $f$ and therefore generalizes to the full distribution $P_X$.*

*Proof.* Since $y = f(\varphi^\star(x))$ for all $x$, the ground-truth $f$ satisfies the constraint of the program, so any minimizer $\hat{f}$ has $|\operatorname{vsupp} \hat{f}| \leq |\operatorname{vsupp} f|$, and $\hat{f}$ agrees with $f$ on $U$ by feasibility. By Assumption 2, $|\operatorname{vsupp} \hat{f}| \geq |\operatorname{vsupp} f|$, so equality holds and $\hat{f} = f$ on $\mathcal{Z}$. Since $y = f(z^\star(x))$ for every $x$, $\hat{f}$ has zero risk under $P_X$. □

### C.3 Proof of Theorem 3

#### C.3.1 Formalizing the richness of task distribution

Here, we formalize the "sufficiently rich" task distribution of the multi-task dataset used in Theorem 3. Each task $T \sim \mathcal{D}_{\mathcal{T}}$ consists of a support $S_T \subseteq [d]$ and a function $f_T : \mathcal{Z} \to \mathcal{Y}$ of the form $f_T(z^\star) = \tilde{f}_T(z^\star_{S_T})$, where $\mathcal{Y} \subset \mathbb{R}$ is a finite answer set (e.g., $\{0,1\}$ for yes/no questions, $n$ discrete bins for counting questions with maximum $n$ objects), with labels $y = f_T(\varphi^\star(x))$.

**Assumption 3** (Joint distinguishability with a common anchor)**.** *Fix an integer $k$ with $1 \le k < d$, and let $\mathcal{Z} = \{0,1\}^d$. Each task $T \sim \mathcal{D}_{\mathcal{T}}$ consists of a support $S_T \subseteq [d]$ and a function $f_T(u) = \widetilde{f}_T(u_{S_T})$, where $\widetilde{f}_T : \{0,1\}^{S_T} \to \mathcal{Y}$. The task distribution satisfies the following conditions:*

1. $|S_T| = k$ almost surely, and
$$\Pr(S_T = S) > 0 \qquad \text{for every } S \subseteq [d] \text{ with } |S| = k. \tag{14}$$
2. For every $S$ and every pair of distinct inputs $a, b \in \{0,1\}^S$,
$$\Pr\Big(\widetilde{f}_T(a) \neq \widetilde{f}_T(b) \mid S_T = S\Big) > 0. \tag{15}$$
3. For every $S$, there exists an *anchor* $a_S \in \{0,1\}^S$ such that
$$\Pr\Big(\widetilde{f}_T^{-1}\Big(\widetilde{f}_T(a_S)\Big) = \{a_S\} \mid S_T = S\Big) = 1. \tag{16}$$

Joint distinguishability requires the task family to collaboratively distinguish all distinct inputs (different tasks can distinguish different pairs). We additionally require an *anchor* that every task on that support can recognize uniquely.

**Disclaimer.** The auxiliary CLEVR programs used in Section 6 are structured tasks and therefore do *not* generally satisfy this joint distinguishability condition. We do not claim that Theorem 3 theoretically supports the result of our experiments; Theorem 3 should be viewed as a sufficient-condition for the identifiability rather than as a formal characterization of our CLEVR task bank. The experiments test whether the same task sparsity principle is effective outside these idealized conditions.

**Lemma C.2** (Sufficient representations are bijective recoding of $\varphi^\star$)**.** *Let $\mathcal{Z} = \{0,1\}^d$ and let $\varphi^\star : \mathcal{X} \to \mathcal{Z}$ be surjective. Suppose a distribution over task functions $f_T : \mathcal{Z} \to \mathcal{Y}$ satisfies*
$$\Pr_T\big(f_T(u) \neq f_T(u')\big) > 0 \qquad \textit{for every distinct } u, u' \in \mathcal{Z}. \tag{17}$$
*Let $\varphi : \mathcal{X} \to \mathcal{Z}$ be sufficient for almost every task: for almost every $T$, there exists a head $g_T : \mathcal{Z} \to \mathcal{Y}$ such that*
$$g_T(\varphi(x)) = f_T(\varphi^\star(x)) \qquad \textit{for every } x \in \mathcal{X}. \tag{18}$$
*Then $\varphi$ is surjective, and there exists a unique bijection $h : \mathcal{Z} \to \mathcal{Z}$ such that*
$$\varphi = h \circ \varphi^\star. \tag{19}$$
*Moreover, every task has a unique exact head on the learned cube, given by $g_T = f_T \circ h^{-1}$. Conversely, every bijective recoding of $\varphi^\star$ is sufficient for every task.*

*Proof.* Suppose $\varphi(x) = \varphi(x')$ but $u := \varphi^\star(x) \neq \varphi^\star(x') =: u'$. By Equation (17), the event $f_T(u) \neq f_T(u')$ has positive probability and therefore contains a task for which exact prediction holds. For this task,
$$f_T(u) = g_T(\varphi(x)) = g_T(\varphi(x')) = f_T(u'), \tag{20}$$
a contradiction. Thus the decoder
$$v : \varphi(\mathcal{X}) \to \mathcal{Z}, \qquad v(\varphi(x)) := \varphi^\star(x), \tag{21}$$
is well-defined. Surjectivity of $\varphi^\star$ makes $v$ surjective. Since $\varphi(\mathcal{X}) \subseteq \mathcal{Z}$ and $|\mathcal{Z}| = 2^d$, it follows that $\varphi(\mathcal{X}) = \mathcal{Z}$ and $v$ is bijective. Setting $h = v^{-1}$ gives $\varphi = h \circ \varphi^\star$; surjectivity of $\varphi^\star$ ensures uniqueness of $h$. For every task, the head $g_T = f_T \circ h^{-1}$ gives exact prediction, and surjectivity of $\varphi$ makes this head unique. The same formula shows that any bijective recoding is sufficient for every task. □

**Theorem C.3** (Identification from average task sparsity). *Let $\varphi^\star : \mathcal{X} \to \mathcal{Z}$ be surjective, and suppose Assumption 3 holds. Consider a sufficient representation $\varphi : \mathcal{X} \to \mathcal{Z}$ which for every task $T$, there exists a head*

$$g_T(\varphi(x)) = f_T(\varphi^\star(x)) \ \text{ for every } x \in \mathcal{X}. \tag{22}$$

*Define the sparsity objective under the representation $\varphi$,*

$$s_\varphi(T) := \min_{g:\, g\circ\varphi = f_T\circ\varphi^\star} \big|\mathrm{vsupp}(g)\big|, \qquad \mathcal{L}(\varphi) := \mathbb{E}_T[s_\varphi(T)]. \tag{23}$$

*Then $\mathcal{L}(\varphi) \geq k$, with equality if and only if there exist a permutation $\sigma$ of $[d]$ and an invertible function $v_i : \{0,1\} \to \{0,1\}$ such that*

$$\varphi_i(x) = v_i(\varphi^\star_{\sigma(i)}(x)) \ \text{ for every } i \in [d] \text{ and } x \in \mathcal{X}. \tag{24}$$

*Proof.* (L(φ) ≥ k). By Lemma C.2, we have bijection $h$ with $\varphi = h^{-1} \circ \varphi^\star$. The unique task heads and their sparsities are

$$g_T(z) = \widetilde{f}_T\big(h(z)_{S_T}\big), \qquad s_\varphi(T) = \big|\mathrm{vsupp}(g_T)\big|. \tag{25}$$

Fix a task satisfying the common-anchor condition, and write $c_T = \widetilde{f}_T(a_S)$. The corresponding preimage of $c_T$ is

$$f_T^{-1}(c_T) = \{u \in \mathcal{Z} : u_{S_T} = a_{S_T}\}, \tag{26}$$

which has size of $2^{d-k}$. By bijectivity of $v$, the learned preimage $g_T^{-1}(c_T)$ also has size of $2^{d-k}$. However, this preimage contains at least $2^{d-|\,\mathrm{vsupp}(g_T)|}$, as single solution will have $2^{d-|\,\mathrm{vsupp}(g_T)|}$ other solutions with the same answer. Therefore $2^{d-k} \geq 2^{d-|\,\mathrm{vsupp}(g_T)|}$, and hence

$$s_\varphi(T) = |\,\mathrm{vsupp}(g_T)| \geq k \qquad \text{almost surely.} \tag{27}$$

(L(φ⋆) = k). Having an anchor $a_{S_T}$ also makes every coordinate of $S_T$ essential for $f_T$, because flipping a coordinate at $a_{S_T}$ changes the answer from $c_T$. Thus $s_{\varphi^\star}(T) = k$ almost surely, so $\mathcal{L}(\varphi^\star) = k$.

(φ⋆ is unique up to trivial transforms). Suppose that $\mathcal{L}(\varphi) = k$.

Suppose $\mathcal{L}(\varphi) = k$. Since $s_\varphi(T) \geq k$ almost surely, we have $s_\varphi(T) = k$ almost surely.

Fix a $k$-subset $S$ and define

$$F_S := \{z : h(z)_S = a_S\}. \tag{28}$$

For almost every task with $S_T = S$, the anchor assumption and bijectivity of $h$ give

$$F_S = g_T^{-1}\big(\widetilde{f}_T(a_S)\big), \qquad |F_S| = 2^{d-k}, \qquad |\,\mathrm{vsupp}\, g_T| = k. \tag{29}$$

Since a preimage of $g_T$ is a union of coordinates faces fixing $J_T$, each of size $2^{d-k}$, it follows that $F_S = \{z : z_{J_T} = b_T\}$ for some $b_T$. The coordinates fixed by a coordinate face are uniquely determined. Thus the common preimage $F_S$ forces

$$\mathrm{vsupp}(g_T) = J_S, \qquad |J_S| = k, \quad \text{for almost every } T \mid S_T = S, \tag{30}$$

for a set $J_S$ independent of $T$.

If $z_{J_S} = z'_{J_S}$, then Equation (30) gives

$$\widetilde{f}_T\big(h(z)_S\big) = \widetilde{f}_T\big(h(z')_S\big) \qquad \text{for almost every } T \mid S_T = S. \tag{31}$$

Joint distinguishability implies $h(z)_S = h(z')_S$. Hence $h_S$ depends only on $z_{J_S}$, and therefore

$$\bigcup_{i\in S} \mathrm{vsupp}(h_i) \subseteq J_S, \qquad \left|\bigcup_{i\in S} \mathrm{vsupp}(h_i)\right| \leq k \quad \text{for every } |S| = k. \tag{32}$$

On the other hand, for every $I \subseteq [d]$, the map $h_I$ depends only on the coordinates in $\bigcup_{i \in I} \operatorname{vsupp} h_i$. Surjectivity of $h$ implies that $h_I$ attains all $2^{|I|}$ binary patterns, and therefore

$$\left| \bigcup_{i \in I} \operatorname{vsupp} h_i \right| \geq |I|. \tag{33}$$

By Hall's marriage theorem (Hall, 1987), Equation (33) yields a permutation $\sigma$ of $[d]$ such that $\sigma(i) \in \operatorname{vsupp} h_i$ for every $i$.

Suppose that $\sigma(j) \in \operatorname{vsupp} h_i$ for some $j \neq i$. Since $k < d$, there exists a $k$-subset $S$ containing $i$ but not $j$. Then

$$\bigcup_{\ell \in S} \operatorname{vsupp} h_\ell \supseteq \{\sigma(\ell) : \ell \in S\} \cup \{\sigma(j)\}, \tag{34}$$

whose right-hand side has $k+1$ elements, contradicting Equation (32). Thus, $\operatorname{vsupp} h_i = \{\sigma(i)\}$, so

$$h_i(z) = v_i(z_{\sigma(i)}) \tag{35}$$

with $v_i$ being an invertible function, leading the form of $\varphi$ stated in the theorem. Conversely, coordinate permutations and invertible element-wise transforms preserve the cardinality of variable supports, so every representation with those trivial transforms attains $\mathcal{L}(\varphi) = k$. □