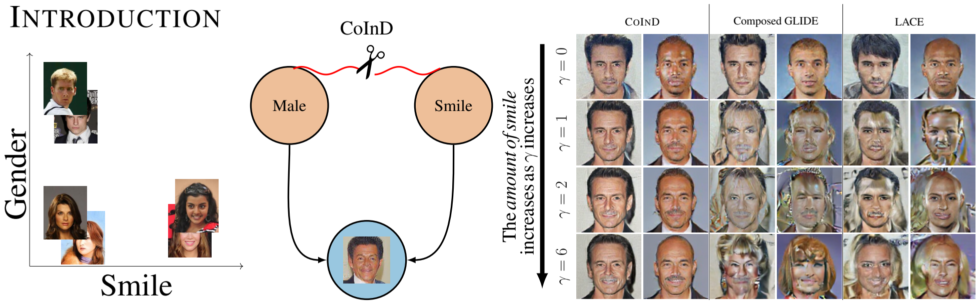
SPURIOUS CORRELATIONS IN DIFFUSION MODELS AND HOW TO FIX THEM

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative models are not immune to spurious correlations. The spuriousness in generative models is defined by their ability to compose attributes faithfully, often referred to as compositionality in generative models. To compose attributes successfully, a model should learn to accurately capture the statistical independence between attributes. This paper shows that standard conditional diffusion models violate this assumption, even when all attribute compositions are observed during training. And, this violation is significantly more severe when only a subset of the compositions is observed. We propose COIND to address this problem. It explicitly enforces statistical independence between the conditional marginal distributions by minimizing Fisher's divergence between the joint and marginal distributions. The theoretical advantages of COIND are reflected in both qualitative and quantitative experiments, demonstrating a significantly more faithful and precise controlled generation of samples for arbitrary compositions of attributes.

1 INTRODUCTION



(a) observational data. (b) Generative model (c) Generated samples

Figure 1: **COIND enables compositionality.** (a) The task of generating unseen compositions of *smiling male* celebrities from only observing three other gender and smile combinations. (c) Diffusion models - Composed GLIDE and LACE - often struggle to accurately compose smile and gender, as they tend to leak gender attributes like hair while controlling for smile, due to their association in the observational data. In contrast, COIND successfully controls for smile by removing any attribute dependencies, as demonstrated in (b).

Many applications of generative models, such as image editing, require explicit and independent control over statistically independent attributes (Brooks et al., 2022). However, failing to learn the conditional independence between attributes results in spurious dependencies, leading to harmful generated content—including stereotypes (Dehdashtian et al., 2025) and biases (Luccioni et al., 2024). To investigate the implicit biases learned by generative models, we examine whether one attribute can be varied independently of others, mathematically put, Do conditional diffusion models implicitly learn conditional independence?

Consider the illustrative example in Fig. 1, which involves generating realistic samples of novel *smiling male* celebrities while retaining explicit control over smile and gender attributes. To generate samples with such compositions, Liu et al.; Du et al.; Nie et al. compose the conditional marginal distributions $p(\text{image} \mid \text{gender} = \textit{male})$, $p(\text{image} \mid \text{smile} = \textit{true})$, refereed as compositional-generation. These marginals are derived either by training separate energy-based models for individual attributes (LACE (Du et al., 2020; Nie et al., 2021)) or by factorizing the joint attribute distribution (Composed GLIDE (Liu et al., 2023)). Both approaches rely on the critical assumption that the conditional marginal distributions are statistically independent. However, as demonstrated in Fig. 1c, both methods inadvertently increase hair length—a feature associated with female celebrities—when controlling for smile. This failure under partial attribute support aligns with observations in Du et al. (2020). These observations naturally raise the following research questions that this paper seeks to answer:

– (RQ1) *Why do standard classifier-free diffusion models fail to generate data with arbitrary compositions of attributes?* We hypothesize that violating the conditional independence assumption, will result in poor image quality, diminished control over the generated image attributes, and, ultimately, failure to adhere to the desired composition. We verify our hypothesis through a case study in § 3.

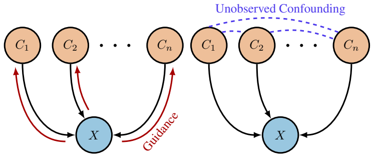
– (RQ2) *How can we explicitly enable diffusion models to generate data with arbitrary compositions of attributes?* We adopt the principle of independent causal mechanisms (Peters et al., 2017) to express the conditional data likelihood in terms of the constituent conditional marginal distributions to ensure that the model does not learn non-existent statistical dependencies from the training data. This framework leads to our proposed objective, CoInD, which combines distribution matching (black arrows in Fig. 1b) with causal independence constraints, breaking any spurious dependency (red line in Fig. 1b), enables independent control over smile and gender attributes, depicted in Fig. 1c

Strong inductive biases, in the form of the conditional independence relations in CoInD, enable compositionality in diffusion models with fine-grained control over conditioned attributes and diversity for unconditioned attributes. CoInD achieves these goals while being monolithic and is scalable with the number of attributes, and can be incorporated with just few lines of code.

2 Compositionality in Diffusion Models

We study the model of learning compositional functions from limited data. As compositionality is a property of data generating process (Wiedemer et al., 2024). We consider the case, where the samples are generated by independently varying factors, and have access to the labels of these factors. In this section, we formally define the assumption in the data generation process. **Notations.** We use bold lowercase and uppercase characters to denote vectors (e.g., $\boldsymbol{a}$) and matrices (e.g., $\boldsymbol{A}$) respectively. Random variables are denoted by uppercase Latin characters (e.g., X). With a slight abuse of notation, we refer to $p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_i)$ as marginal, and joint as $p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C)$.

Data Generation Process. The data generation process consists of observed data $\boldsymbol{X}$ (e.g., images) and its attribute variables $C_1, C_2, \dots, C_n$. To have explicit control over these attributes during generation, they should vary independently of each other (Mathieu et al., 2016). Note that the attributes that we wish to control in practice may be causally related to each other. But, we limit our work to only those causal graphs where they are not causally related to each other as shown in Fig. 2a. Each C_i assumes values from a set $\mathcal{C}_i$ and the joint set $\mathcal{C} = \mathcal{C}_1 \times \dots \times \mathcal{C}_n$ is referred to as the *attribute space*. These attributes generate, $\boldsymbol{X}$ according to the causal graph described in Fig. 2a. Functionally, $\boldsymbol{X} = f(C_1, \dots, C_n, \boldsymbol{U_X})$ where f is the function that generates $\boldsymbol{X}$, $\boldsymbol{U_X}$ collectively denotes the unobserved exogenous variables that affect $\boldsymbol{X}$. Outside of the graphical assumptions in Fig. 2a, we also assume that f is invertible w.r.t. the attributes such that it is possible to estimate $C_1, \dots, C_n$ from $\boldsymbol{X}$. As a result, $C_1, \dots, C_n$ are mutually independent given $\boldsymbol{X}$.



(a) True underlying causal model (b) Causal model during training

Figure 2: (a) $C_1, C_2, \dots, C_n$ vary freely and independently in the underlying causal graph. (b) However, they become dependent during training due to unknown and unobserved confounding factors.

The attribute space in our problem statement has the following properties. **(1)** The attribute space observed during training $\mathcal{C}_{\text{train}}$ covers $\mathcal{C}$ in the following sense:

Definition 1 (Support Cover)**.** *Let $\mathcal{C} = \mathcal{C}_1 \times \dots \times \mathcal{C}_n$ be the Cartesian product of n finite sets $\mathcal{C}_1, \dots, \mathcal{C}_n$. Consider a subset $\mathcal{C}_{train} \subset \mathcal{C}$. Let $\mathcal{C}_{train} = \{(c_{1j}, \dots, c_{nj}) : c_{ij} \in \mathcal{C}_i, 1 \leq i \leq n, 1 \leq j \leq m\}$ and $\tilde{\mathcal{C}}_i = \{c_{ij} : 1 \leq j \leq m\}$ for $1 \leq i \leq n$. The Cartesian product of these sets is $\tilde{\mathcal{C}}_{train} = \tilde{\mathcal{C}}_1 \times \dots \times \tilde{\mathcal{C}}_n$. We say $\mathcal{C}_{train}$ covers $\mathcal{C}$ iff $\mathcal{C} = \tilde{\mathcal{C}}_{train}$.*

Informally, this assumption implies that **(1)** Every possible value that C_i can assume is present in the training set, and open-set attribute compositions do not fall under this definition. **(2)** For every ordered tuple $c \in \mathcal{C}_{\text{train}}$, there is another $c' \in \mathcal{C}_{\text{train}}$ such that c and c' differ on only one attribute.

Preliminaries on Score-based Models In this work, we train conditional score-based models (Song et al., 2021) using classifier-free guidance (Ho & Salimans, 2022) to generate data corresponding by composing attributes. Score-based models learn the score of the observed data distributions $p_{\text{train}}(\boldsymbol{X})$ and $p_{\text{train}}(\boldsymbol{X} \mid C)$ through score matching (Hyvärinen & Dayan, 2005). Once the score of a distribution is learned, samples can be generated using Langevin dynamics. Liu et al. (2023) proposed the following modifications during sampling to enable compositionality, assuming that the model learns the conditional independence relations from the data-generation process. Refer to App. A for more details on score-based models, including exact formulation.

Compositional Sampling: $C_1 = c_1 \wedge C_2 = c_2$ generates data where attributes C_1 and C_2 takes values c_1 and c_2 respectively. Since, $p_{\boldsymbol{\theta}}(C_1 \wedge C_2 \mid \boldsymbol{X}) = p_{\boldsymbol{\theta}}(C_1 \mid \boldsymbol{X}) p_{\boldsymbol{\theta}}(C_2 \mid \boldsymbol{X})$ samples are generated for the composition $C_1 \wedge C_2$ by sampling from the following score:

$$\nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_1 \wedge C_2) = \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_1) + \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_2) - \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}) \quad (1)$$

This formulation gives can me modified for additional flexibility of controlling for attributes, where γ controls the strength of attribute C_1 w.r.t C_2, $\nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_1 \wedge C_2)$ can be written as:

$$\gamma \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_1) + \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_2) - \gamma \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}) \quad (2)$$

3 Why do diffusion models fail to generate Novel compositions?

To study **(RQ1)**, we consider the task of generating synthetic images from the Colored MNIST dataset with explicit and independent control over the composition of color and digit. We systematically investigate dependencies learned by the model with varying observational support.

(1) Uniform support, all attribute compositions are equally likely to be observed. Although the attributes can vary independently, sometimes they may not do so in the training dataset due to unobserved confounding such as sample selection bias (Storkey, 2008), leading to an attribute shift. In such cases, the underlying causal model during training modifies as shown in Fig. 2b. In practice, all attribute compositions may be observed with unequal probabilities. We refer to this scenario as *(2) non-uniform support*. In more severe cases, this dependence could lead to the training samples consisting of only a subset of all attribute compositions, i.e., $\mathcal{C}_{\text{train}} \subset \mathcal{C}$. We refer to this scenario as *(3) partial support*. To quantify the dependence in the observational data we measure Mutual Information (MI) between attributes. Refer to App. D.4 for visual representation and exact formulation.

To evaluate the generation of the models, we first infer attributes $(\hat{c}_1, \ldots, \hat{c}_n)$ from the generated images $\hat{\boldsymbol{X}}$ using attribute-specific classifiers ϕ_{C_i} and compare them against the expected attributes from the input composition $(c_1, \ldots, c_n)$. We refer to this accuracy as *conformity score* (CS) and is given by $\text{CS}(g) = \mathbb{E}_{p(C)p(U)} \left[\prod_i^n \mathbb{1}(C_i, \phi_{C_i}(g(C, U)))\right]$ where $\mathbb{1}(\cdot, \cdot)$, g, and U are the indicator function, diffusion model, and the stochastic noise in the generation process respectively. We provide more details about conformity score in App. A.2.

Diffusion models learn the dependence in the underlying distribution.

While the underlying causal structure in Fig. 2a assumes independence, diffusion models trained on varying support settings fail to achieve ideal conformity score (high CS). Performance (CS) degrades progressively: models maintain strong results under uniform support (all attribute combinations equally likely), decline under non-uniform support (biased correlations), and worsen significantly under partial support (missing combinations). These findings Tab. 1 demonstrate diffusion models lack inherent compositional bias, instead propagate dependencies as present in their training data.

Support (MI)	JSD ↓	CS↑
Uniform (0.00)	0.16	97.40
Non Uniform (0.33)	0.33	82.60
Partial (1.70)	2.76	17.90

Table 1: JSD and CS of diffusion models under various support settings for the Colored MNIST dataset.

Diffusion models violate conditional independence assumption.

We measure CI violation as the disparity between the conditional joint distribution $p_{\boldsymbol{\theta}}(C \mid \boldsymbol{X})$ and the product of conditional marginal distributions $\prod_i^n p_{\boldsymbol{\theta}}(C_i \mid \boldsymbol{X})$ learned by the implicit classifier of diffusion model using Jensen-Shannon divergence (JSD) as,

$$\text{JSD} = \mathbb{E}_{C,\boldsymbol{X}\sim p_{\text{data}}}\left[D_{\text{JS}}\left(p_{\boldsymbol{\theta}}(C \mid \boldsymbol{X}) \,||\, \prod_i^n p_{\boldsymbol{\theta}}(C_i \mid \boldsymbol{X})\right)\right] \tag{3}$$

where D_{JS} is the Jensen-Shannon divergence and $p_{\boldsymbol{\theta}}$ is obtained by following (Li et al., 2023) and evaluating the implicit classifier learned by the diffusion model. Note that JSD, measure *conditional* independence, whereas MI measure independence between attributes. More details in App. A.1.

Positive JSD values indicate violations of the conditional independence (CI) relation in diffusion models. As quantified in Tab. 1, CI violations worsen from uniform to non-uniform support and become severe under partial support. This degradation strongly correlates with the mutual information (MI) between attributes in training data Pearson's $r = 0.993, p < 0.01$. Our analysis confirms diffusion models directly inherit and magnify spurious correlations from their training distribution rather than learning conditional independence between attributes.

Violation in conditional independence originates from the standard training objective of diffusion models that maximize the likelihood of conditional generation. Under perfect loss, for every observed composition ($C \in \mathcal{C}_{train}$), the model accurately learns $p_{\text{train}}(\boldsymbol{X} \mid C)$, i.e., $p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C) \approx p_{\text{train}}(\boldsymbol{X} \mid C) = p(\boldsymbol{X} \mid C)$, However, learn incorrect marginals, $p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_i) \approx p_{\text{train}}(\boldsymbol{X} \mid C_i) \neq p(\boldsymbol{X} \mid C_i)$. Refer to App. B.1 for complete proof. Informally, $p_{\text{train}}(\boldsymbol{X} \mid C_1 = 1)$, consists of only smiling females. While, the underlying $p(\boldsymbol{X} \mid C_1 = 1)$ contains smiling male, and female celebrities. These incorrect marginals lead to violation in CI.

Based on these observations, we propose CoInD to train diffusion models that explicitly enforce the conditional independence dictated by the underlying causal data-generation process to encourage the model to learn accurate marginal distributions of the attributes.

4 CoInD: Enforcing Conditional Independence Enables Compositionality

In the previous section, we observed that diffusion models violate conditional independence (CI) by learning incorrect marginals. To remedy this, CoInD uses a training objective that explicitly enforces the causal factorization to ensure that the trained diffusion models obey CI. Applying the assumption of $C_1 \perp\!\!\!\perp \ldots \perp\!\!\!\perp C_n \mid \boldsymbol{X}$ mentioned in § 2, we have $p(\boldsymbol{X} \mid C) = \frac{p(\boldsymbol{X})}{p(C)} \prod_i^n \frac{p(\boldsymbol{X}|C_i)p(C_i)}{p(\boldsymbol{X})}$. Note that the invariant $p(\boldsymbol{X} \mid C)$ is now expressed as the product of marginals employed for sampling. Therefore, training the diffusion model by maximizing this conditional likelihood is naturally more suited for learning accurate marginals for the attributes. We minimize the distance between the true conditional likelihood and the learned conditional likelihood as,

$$\mathcal{L}_{\text{comp}} = \mathcal{W}_2\left(p(\boldsymbol{X} \mid C), \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X})}{p_{\boldsymbol{\theta}}(C)} \prod_i \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_i)p_{\boldsymbol{\theta}}(C_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{X})}\right) \tag{4}$$

By applying the triangle inequality and leveraging the Wasserstein distance upper bound via Fisher divergence (Kwon et al., 2022), we derive the following inequality:

$$\mathcal{L}_{\text{comp}} \leq K_1\sqrt{\mathcal{L}_{\text{score}}} + K_2\sqrt{\mathcal{L}_{\text{CI}}} \tag{5}$$

for constants $K_1, K_2 > 0$. A complete derivation of this bound is provided in App. B.2.

Distribution matching:

$$\mathcal{L}_{\text{score}} = \mathbb{E}_{p(\boldsymbol{X},C)}\|\nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C) - \nabla_{\boldsymbol{X}} \log p(\boldsymbol{X} \mid C)\|_2^2 \tag{6}$$

Conditional Independence:

$$\mathcal{L}_{\text{CI}} = \mathbb{E}\|\nabla_{\mathbf{X}} \log p_\theta(\mathbf{X} \mid C) - \nabla_{\mathbf{X}} \log p_\theta(\mathbf{X}) - \sum_i \left[\nabla_{\mathbf{X}} \log p_\theta(\mathbf{X} \mid C_i) - \nabla_{\mathbf{X}} \log p_\theta(\mathbf{X})\right]\|_2^2 \tag{7}$$

Practical Implementation. A computational burden presented by $\mathcal{L}_{\text{CI}}$ in Eq. (7) is that the required number of model evaluations increases linearly with the number of attributes. To mitigate this burden, we approximate the mutual conditional independence with pairwise conditional independence (Hammond & Sun, 2006). Thus, the modified $\mathcal{L}_{\text{CI}}$ becomes,

$$\mathcal{L}_{\text{CI}} = \mathbb{E}_{p(\boldsymbol{X},C)}\mathbb{E}_{j,k}\|\nabla_{\boldsymbol{X}} \log p_\theta(\boldsymbol{X} \mid C_j, C_k) - \nabla_{\boldsymbol{X}} \log p_\theta(\boldsymbol{X} \mid C_j) - \nabla_{\boldsymbol{X}} \log p_\theta(\boldsymbol{X} \mid C_k) + \nabla_{\boldsymbol{X}} \log p_\theta(\boldsymbol{X})\|_2^2$$

The weighted sum of the square of the terms in Eq. (5) has shown stability. Therefore, CoInD's training objective:

$$\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{score}} + \lambda\mathcal{L}_{\text{CI}} \tag{8}$$

where λ is the hyper-parameter that controls the strength of conditional independence. The reduction to the practical version of the upper bound (Eq. (5)) is discussed in extensively in App. C. For guidance on selecting hyper-parameters in a principled manner, please refer to App. C.3. Finally, our proposed approach can be implemented with just a few lines of code, as outlined in Algorithm 1.

5 Experiments: Learning Independent Marginals Enables Compositionality

CoInD encourages diffusion models to learn conditionally independent marginals of attributes, and thereby improve their compositionality. In this section, we design experiments to evaluate CoInD on two questions: (1) *does* CoInD *effectively train diffusion models that obey the underlying causal model?*, and (2) *does* CoInD *improve the compositionality of these models?*

Datasets. We use the following image datasets with labeled attributes for our experiments: **(1) Colored MNIST**, **(2) Shapes3d** dataset (Burgess & Kim, 2018), with six attributes to demonstrate scalability of CoInD. **(3) CelebA**. Refer to App. D.5 for details.

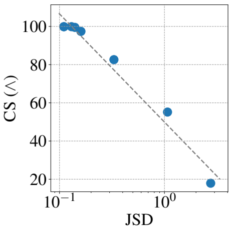
Observed training distributions. We evaluate CoInD on four scenarios where we observe different distributions of attribute compositions during training: (1) Uniform (Fig. 7a). (2) Non-uniform (Fig. 7b). (3) Diagonal partial (Fig. 7c), (4) Orthogonal partial support (Fig. 7d). Refer to App. D.4.

Metrics. We measure the JSD of the trained models to answer the first question. To answer the second question, we measure the confirmity score (CS) described in § 3, following the compositional sampling described in Eq. (1). In addition to the CS we measure R^2 for datasets containing unique ground truth images corresponding to the input composition. For uniform and non-uniform support, we evaluate the generations on the input compositions that span the attribute space $\mathcal{C}$. In other cases, the generations only span unseen compositional support, i.e., $\mathcal{C} \setminus \mathcal{C}_{\text{train}}$.

Baselines. **LACE** (Nie et al., 2021) trains distinct energy-based models (EBMs) for each attribute and combines them following the compositional sampling described in § 2. A similar approach was proposed by (Du et al., 2020). However, in our experimental evaluation for LACE, we train distinct score-based models instead of EBMs. In contrast, **Composed GLIDE** samples from score-based models by factorizing the joint distribution into marginals, assuming these models had implicitly learned conditional independence. Additional details about the baselines are delegated to App. D.3.

Support	Method	JSD ↓	CS ↑
Uniform	LACE	-	99.04
	Composed GLIDE	0.16	97.40
	CoInD ($\lambda = 0.2$)	0.14	99.50
	CoInD ($\lambda = 1.0$)	**0.11**	**99.86**
Non-uniform	LACE	-	51.70
	Composed GLIDE	0.33	82.60
	CoInD ($\lambda = 1.0$)	**0.11**	**99.84**
Partial	LACE	-	22.94
	Composed GLIDE	2.76	17.90
	CoInD ($\lambda = 1.0$)	**1.07**	**55.16**

(a) Results on Colored MNIST Dataset



(b) JSD vs CS Colored MNIST

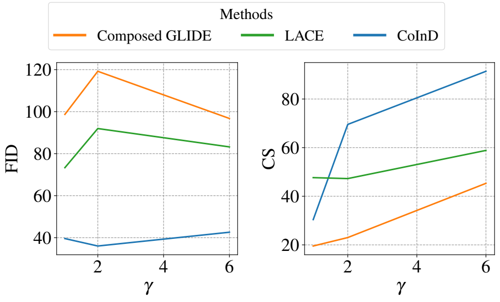
Support	Method	JSD ↓	Composition	
			R^2 ↑	CS ↑
Uniform	LACE	-	0.97	91.19
	Composed GLIDE	0.302	0.94	83.75
	CoInD ($\lambda = 1.0$)	**0.215**	**0.98**	**95.31**
Orthogonal	LACE	-	0.88	62.07
	Composed GLIDE	0.503	0.86	51.56
	CoInD ($\lambda = 1.0$)	**0.287**	**0.97**	**91.10**

(c) Results on Colored MNIST Dataset

Figure 3: **Results on Colored MNIST, Shapes3d dataset.** (a,c) CoInD reduces JSD, and improved CS on Colored MNIST, Shapes3d datasets across supports. (b) Plotting CS against JSD in the log scale of the models trained under different settings reveals a negative correlation. Showcasing reducing CI, will result in faithful generation.

Method	JSD ↓	Composition	
		CS ↑	FID ↓
LACE	-	**47.65**	73.33
Composed GLIDE	0.394	19.53	98.70
CoInD ($\lambda = 100$)	**0.165**	30.40	**39.58**

(a) Results on CelebA Dataset



(b) FID with ↑ γ (c) CS with ↑ γ

Figure 4: **CoInD provides fine-grained control over attributes on CelebA** (a) CoInD generates faithful (CS) and high fidelity (FID) samples while composing "smile","male". (b,c) The effect of γ parameter variation for controlling smile. CoInD demonstrates stable image quality (FID) while improving smile attributes, making it easier for the smile classifier to detect, resulting in improved CS, also evident from samples generated Fig. 1c

– CoInD reduces CI violations (↓ JSD) and enhances faithful generation (↑ CS). As observed in § 3 and illustrated in Fig. 3b, there exists a negative correlation between JSD and CS across various methods and observed support settings. This correlation strongly suggests that violations of CI significantly impair the compositionality of standard diffusion models. Consequently, CoInD, which minimizes the Fisher divergence of these violations, effectively breaks spurious dependencies and improves compositionality on Colored MNIST, Shapes 3d, CelebA across supports.

– CoInD generates diverse samples. It is desirable for generated samples to exhibit diverse values for attributes that are not explicitly controlled; otherwise, the model may reinforce undesirable stereotypes. Therefore, when conditioning on digit, CoInD generates diverse colors of samples, as quantified by the Shannon entropy Tab. 2. Unlike methods that explicitly optimize for diversity, CoInD achieves this as a complementary benefit by breaking dependencies induced by unknown confounders (see App. D.7 for details).

Method	**Entropy**
LACE	1.46
C GLIDE	2.38
CoInD	3.26

Table 2: Entropy of uncontrolled attribute in generation

– CoInD is scalable with attributes. Results on the Shapes3D (Table 3c) demonstrate that CoInD successfully composes even 6 attributes, showcasing strong compositional ability compared to baselines, specifically huge improvements on orthogonal support.

6 Related Work

Our work concerns compositional generalization in generative models, where the goal is to generate data with unseen attribute compositions. One class of approaches seek to achieve compositionality by combining distinct models trained for each attribute (Du et al., 2020; Liu et al., 2021; Nie et al., 2021; Du et al., 2023), which is expensive and scaling linearly with the number of attributes also suffer from incorrect marginals. In contrast, we are interested in monolithic compositional diffusion models that learn compositionality. Liu et al. (2023) studied compositionality broadly and proposed methods to represent compositions in terms of marginal probabilities obtained through factorization of the joint distribution. However, these factorized sampling methods fail since the underlying generative model learns inaccurate marginals.

7 Conclusion

In this work, we study spurious correlation in generative models, formulating it as a compositionality problem. We demonstrate that diffusion models capture spurious dependencies between attributes, propagating biases from observational data. This leads to unfaithful sample generation and feature leakage. To address this, we propose CoInD to enforce conditional independence, breaking spurious dependencies and enabling faithful generation of controlled attributes and diverse unconditional attributes.

REFERENCES

Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. *arXiv preprint arXiv:2211.09800*, 2022.

Chris Burgess and Hyunjik Kim. 3d shapes dataset. https://github.com/deepmind/3dshapes-dataset/, 2018.

Sepehr Dehdashtian, Gautam Sreekumar, and Vishnu Naresh Boddeti. Oasis uncovers: High-quality t2i models, same old stereotypes. *arXiv preprint arXiv:2501.00962*, 2025.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Yilun Du, Shuang Li, and Igor Mordatch. Compositional visual generation with energy based models. *Advances in Neural Information Processing Systems*, 33:6637–6647, 2020.

Yilun Du, Conor Durkan, Robin Strudel, Joshua B Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Sussman Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc. In *International conference on machine learning*, pp. 8489–8510. PMLR, 2023.

Peter J Hammond and Yeneng Sun. The essential equivalence of pairwise and mutual conditional independence. *Probability Theory and Related Fields*, 135(3):415–427, 2006.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL `https://arxiv.org/abs/2207.12598`.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

Dohyun Kwon, Ying Fan, and Kangwook Lee. Score-based generative modeling secretly minimizes the wasserstein distance. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=oPzICxVFqVM`.

Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. *arXiv preprint arXiv:2404.07724*, 2024.

Alexander C. Li, Mihir Prabhudesai, Shivam Duggal, Ellis Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2206–2217, October 2023.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code, 2024. URL `https://arxiv.org/abs/2412.06264`.

Nan Liu, Shuang Li, Yilun Du, Josh Tenenbaum, and Antonio Torralba. Learning to compose visual relations. *Advances in Neural Information Processing Systems*, 34:23166–23178, 2021.

Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B. Tenenbaum. Compositional visual generation with composable diffusion models, 2023. URL `https://arxiv.org/abs/2206.01714`.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision*, 2015.

Sasha Luccioni, Christopher Akiki, Margaret Mitchell, and Yacine Jernite. Stable bias: Evaluating societal representations in diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

Michael F Mathieu, Junbo Jake Zhao, Junbo Zhao, Aditya Ramesh, Pablo Sprechmann, and Yann LeCun. Disentangling factors of variation in deep representation using adversarial training. In *Advances in Neural Processing Systems*, 2016.

Weili Nie, Arash Vahdat, and Anima Anandkumar. Controllable and compositional generation with latent-space energy-based models, 2021. URL `https://arxiv.org/abs/2110.10873`.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

Pablo Sánchez-Moreno, Alejandro Zarzo, and Jesús S Dehesa. Jensen divergence based on fisher's information. *Journal of Physics A: Mathematical and Theoretical*, 45(12):125305, 2012.

Lukas Schott, Julius Von Kügelgen, Frederik Träuble, Peter Vincent Gehler, Chris Russell, Matthias Bethge, Bernhard Schölkopf, Francesco Locatello, and Wieland Brendel. Visual representation learning does not generalize strongly within the same domain. In *International Conference on Learning Representations*, 2020.

Maximilian Seitzer. pytorch-fid: FID Score for PyTorch. `https://github.com/mseitzer/pytorch-fid`, August 2020. Version 0.3.0.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=PxTIG12RRHS`.

Amos Storkey. When training and test sets are different: Characterizing learning transfer. In *Dataset Shift in Machine Learning*. The MIT Press, 2008.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688. Citeseer, 2011.

Thaddäus Wiedemer, Prasanna Mayilvahanan, Matthias Bethge, and Wieland Brendel. Compositional generalization from first principles. *Advances in Neural Information Processing Systems*, 36, 2024.

Appendix

Table of Contents

A PRELIMINARIES OF SCORE-BASED MODELS

Score-based models Score-based models (Song et al., 2021) learn the score of the observed data distribution, $P_{\text{train}}(\boldsymbol{X})$ through score matching (Hyvärinen & Dayan, 2005). The score function $s_{\boldsymbol{\theta}}(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_{\boldsymbol{\theta}}(\mathbf{x})$ is learned by a neural network parameterized by $\boldsymbol{\theta}$.

$$L_{\text{score}} = \mathbb{E}_{\mathbf{x}\sim p_{\text{train}}}\left[\|s_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\text{train}}(\mathbf{x})\|_2^2\right] \tag{9}$$

During inference, sampling is performed using Langevin dynamics:

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \frac{\eta}{2}\nabla_{\mathbf{x}} \log p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}) + \sqrt{\eta}\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0,1) \tag{10}$$

where $\eta > 0$ is the step size. As $\eta \to 0$ and $T \to \infty$, the samples $\mathbf{x}_t$ converge to $p_{\boldsymbol{\theta}}(\boldsymbol{X})$ under certain regularity conditions (Welling & Teh, 2011).

Diffusion models Song & Ermon (2019) proposed a scalable variant that involves adding noise to the data Ho et al. (2020) has shown its equivalence to Diffusion models. Diffusion models are trained by adding noise to the image $\mathbf{x}$ according to a noise schedule, and then neural network, ϵ_θ is used to predict the noise from the noisy image, $\mathbf{x_t}$. The training objective of the diffusion models is given by:

$$L_{\text{score}} = \mathbb{E}_{\mathbf{x}\sim p_{\text{train}}}\mathbb{E}_{t\sim[0,T]}\left\|\epsilon - \epsilon_\theta\left(\mathbf{x}_t, t\right)\right\|^2 \tag{11}$$

Here, the perturbed data $\mathbf{x}_t$ is expressed as: $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x} + \sqrt{1-\bar{\alpha}_t}\epsilon$ where $\bar{\alpha}_t = \prod_{i=1}^{T}\alpha_i$, for a pre-specified noise schedule α_t. The score can be obtained using,

$$s_\theta(\mathbf{x}_t, t) \approx -\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}} \tag{12}$$

Langevin dynamics can be used to sample from the $s_\theta(\mathbf{x}_t, t)$ to generate samples from $p(\boldsymbol{X})$. The conditional score (Dhariwal & Nichol, 2021) is used to obtain samples from the conditional distribution $p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C)$ as:

$$\nabla_{\boldsymbol{X}_t} \log p(\boldsymbol{X}_t \mid C) = \underbrace{\nabla_{\boldsymbol{X}_t} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}_t)}_{\text{Unconditional score}} + \gamma\nabla_{\boldsymbol{X}_t} \log \underbrace{p_{\boldsymbol{\theta}}(C \mid \boldsymbol{X}_t)}_{\text{noisy classifier}}$$

where γ is the classifier strength. Instead of training a separate noisy classifier, Ho & Salimans have extended to conditional generation by training $\nabla_{\boldsymbol{X}_t} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}_t \mid C) = s_\theta(\boldsymbol{X}_t, t, C)$. The sampling can be performed using the following equation:

$$\nabla_{\boldsymbol{X}_t} \log p(\boldsymbol{X}_t \mid C) = (1-\gamma)\nabla_{\boldsymbol{X}_t} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}_t) + \gamma\nabla_{\boldsymbol{X}_t} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}_t \mid C) \tag{13}$$

However, the sampling needs access to unconditional scores as well. Instead of modelling $\nabla_{\boldsymbol{X}_t} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}_t)$, $\nabla_{\boldsymbol{X}_t} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}_t|C)$ as two different models Ho & Salimans have amortize training a separate classifier training a conditional model $s_\theta(\mathbf{x}_t, t, \mathbf{c})$ jointly with unconditional model trained by setting $c = \varnothing$.

In the general case of classifier-free guidance, a single model can be effectively trained to accommodate all subsets of attribute distributions. During the training phase, each attribute c_i is randomly set to $\varnothing$ with a probability p_{uncond}. This approach ensures that the model learns to match all possible subsets of attribute distributions. Essentially, through this formulation, we use the same network to model all the possible subsets of conditional probability.

Once trained, the model can generate samples conditioned on specific attributes, such as c_i and c_j, by setting all other conditions to $\varnothing$. The conditional score is then computed as, $\nabla_{\boldsymbol{X}_t} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}_t|c_i, c_j) = s_\theta(\mathbf{x}_t, \mathbf{c}^{i,j})$, where $\mathbf{c}^{i,j}$ represents the condition vector with all values other than i and j set to $\varnothing$. This method allows for flexible and efficient sampling across various attribute combinations.

Estimating Guidance Once the diffusion model is trained, we investigate the implicit classifier, $p_\theta(C|\boldsymbol{X})$, learned by the model. This will give us insights into the learning process of the diffusion models. (Li et al., 2023) have shown a way to calculate $p_\theta(C_i = c_i \mid \boldsymbol{X} = \mathbf{x})$, borrowing equation (5), (6) from (Li et al., 2023).

$$p_\theta(C_i = c_i \mid \mathbf{x}) = \frac{p(c_i)\, p_\theta(\mathbf{x} \mid c_i)}{\sum_k p(c_k)\, p_\theta(\mathbf{x} \mid c_k)}$$

$$p_\theta(C_i = c_i \mid \mathbf{x}) = \frac{\exp\{-\mathbb{E}_{t,\epsilon}[\|\epsilon - \epsilon_\theta(\mathbf{x}_t, t, \mathbf{c}^i)\|^2]\}}{\mathbb{E}_{C_i}\left[\exp\{-\mathbb{E}_{t,\epsilon}[\|\epsilon - \epsilon_\theta(\mathbf{x}_t, t, \mathbf{c}^i)\|^2]\}\right]} \tag{14}$$

Likewise, we can extend it to joint distribution by

$$p_\theta(C_i = c_i, C_j = c_j \mid \mathbf{x}) = \frac{\exp\{-\mathbb{E}_{t,\epsilon}[\|\epsilon - \epsilon_\theta(\mathbf{x}_t, t, \mathbf{c}^{i,j})\|^2]\}}{\mathbb{E}_{C_i,C_j}\left[\exp\{-\mathbb{E}_{t,\epsilon}[\|\epsilon - \epsilon_\theta(\mathbf{x}_t, t, \mathbf{c}^{i,j})\|^2]\}\right]} \tag{15}$$

Practical Implementation The authors Li et al.. have showed many axproximations to compute $\mathbb{E}_{t,\epsilon}$. However, we use a different approximation inspired by Kynkäänniemi et al. (2024), where we sample 5 time-steps between [300,600] instead of these time-steps spread over the [0, T].

A.1 Computing JSD

We are interested in understanding the causal structure learned by diffusion models. Specifically, we aim to determine whether the learned model captures the conditional independence between attributes, allowing them to vary independently. This raises the question: *Do diffusion models learn the conditional independence between attributes?* The conditional independence is defined by:

$$p_{\boldsymbol{\theta}}(C_i, C_j \mid \boldsymbol{X}) = p_{\boldsymbol{\theta}}(C_i \mid \boldsymbol{X}) p_{\boldsymbol{\theta}}(C_j \mid \boldsymbol{X}) \tag{16}$$

We aim to measure the violation of this equality using the Jensen-Shannon divergence (JSD) to quantify the divergence between two probability distributions:

$$\text{JSD} = \mathbb{E}_{p_{\text{data}}}\left[D_{\text{JS}}\left(p_{\boldsymbol{\theta}}(C \mid \boldsymbol{X}) \mid\mid p_{\boldsymbol{\theta}}(C_i \mid \boldsymbol{X}) p_{\boldsymbol{\theta}}(C_j \mid \boldsymbol{X})\right)\right] \tag{17}$$

The joint distribution, $p_{\boldsymbol{\theta}}(C_i, C_j \mid \boldsymbol{X})$, and the marginal distributions, $p_{\boldsymbol{\theta}}(C_i \mid \boldsymbol{X})$ and $p_{\boldsymbol{\theta}}(C_j \mid \boldsymbol{X})$, are evaluated at all possible values that C_i and C_j can take to obtain the probability mass function (pmf). The probability for each value is calculated using Equation Eq. (15) for the joint distribution and Equation Eq. (14) for the marginals.

Practical Implementation For the diffusion model with multiple attributes, the violation in conditional *mutual* independence should be calculated using all subset distributions. However, we focus on pairwise independence. We further approximate this in our experiments by computing JSD between the first two attributes, C_1 and C_2. We have observed that computing JSD between any attribute pair does not change our examples' conclusion.

A.2 Conformity Score (CS)

To measure the CS, we first infer attributes $(\hat{c}_1, \ldots, \hat{c}_n)$ from the generated images $\hat{\boldsymbol{X}}$ using attribute-specific classifiers ϕ_{C_i} and compare them against the expected attributes from the input composition $(c_1, \ldots, c_n)$. We refer to this accuracy as *conformity score* (CS) and is given by

$$\text{CS}(g) = \mathbb{E}_{p(C)p(U)}\left[\prod_i^n \mathbb{1}(C_i, \phi_{C_i}(g(C, U)))\right] \tag{18}$$

where $\mathbb{1}(\cdot,\cdot)$, g, and U are the indicator function, diffusion model, and the stochastic noise in the generation process respectively

To obtain a attribute-specific classifier, we train a single ResNet-18 (He et al., 2016) classifier with multiple classification heads, one corresponding to each attribute, and trained on the full support. The effectiveness of the classifier in predicting the attributes is reported in App. D.6

B Proofs for Claims

In this section, we detail the mathematical derivations for violation of conditional independence in diffusion models in App. B.1, and then derive the final loss function of CoInD in App. B.2.

B.1 Standard diffusion model objective is not suitable for compositionality

This section proves that the violation in conditional independence in diffusion models is due to learning incorrect marginals, $p_{\text{train}}(\boldsymbol{X} \mid C_i)$ under $C_i \not\perp\!\!\!\perp C_j$. We leverage the causal invariance property: $p_{\text{train}}(\boldsymbol{X} \mid C) = p_{\text{true}}(\boldsymbol{X} \mid C)$, where p_{train} is the training distribution and p_{true} is the true underlying distribution.

Consider the training objective of the score-based models in classifier free formulation Eq. (9). For the classifier-free guidance, a single model $s_{\boldsymbol{\theta}}(\mathbf{x}, C)$ is effectively trained to match the score of all subsets of attribute distributions. Therefore, the effective formulation for classifier-free guidance can be written as,

$$L_{\text{score}} = \mathbb{E}_{\mathbf{x}\sim p_{\text{train}}}\mathbb{E}_S\left[\|\nabla_{\mathbf{x}} \log p_{\boldsymbol{\theta}}(\mathbf{x} \mid c_S) - \nabla_{\mathbf{x}} \log p_{\text{train}}(\mathbf{x} \mid c_S)\|_2^2\right] \tag{19}$$

where S is the power set of attributes.

From the properties of Fisher divergence, $L_{\text{score}} = 0$ iff $p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid c_S) = p_{\text{train}}(\boldsymbol{X} \mid c_S), \forall S$. In the case of marginals, $p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_i)$ i.e. $S = \{C_i\}$ for some $1 \le i \le n$,

$$\begin{aligned}
p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_i) &= p_{\text{train}}(\boldsymbol{X} \mid C_i) \\
&= \sum_{C_{-i}} p_{\text{train}}(\boldsymbol{X} \mid C_i, C_{-i}) p_{\text{train}}(C_{-i} \mid C_i) \\
&= \sum_{C_{-i}} p_{\text{true}}(\boldsymbol{X} \mid C_i, C_{-i}) p_{\text{train}}(C_{-i} \mid C_i) \\
&\neq \sum_{C_{-i}} p_{\text{true}}(\boldsymbol{X} \mid C_i, C_{-i}) p_{\text{true}}(C_{-i}) = p_{\text{true}}(\boldsymbol{X} \mid C_i) \\
\implies p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_i) &\neq p_{\text{true}}(\boldsymbol{X} \mid C_i)
\end{aligned} \tag{20}$$

Where $C_{-i} = \prod_{\substack{j=1 \\ j\neq i}}^{n} C_j$, which is every attribute except C_i. Therefore, the objective of the score-based models is to maximize the likelihood of the marginals of training data and not the true marginal distribution, which is different from the training distribution when $C_i \not\perp\!\!\!\perp C_j$.

B.2 Step-by-step derivation of CoInD in § 4

The objective is to train the model by explicitly modeling the joint likelihood following the causal factorization. The minimization for this objective can be written as,

$$\mathcal{L}_{\text{comp}} = \mathcal{W}_2\left(p(\boldsymbol{X} \mid C), \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X})}{p_{\boldsymbol{\theta}}(C)} \prod_i \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_i) p_{\boldsymbol{\theta}}(C_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{X})}\right) \tag{21}$$

where $\mathcal{W}_2$ is 2-Wasserstein distance. Applying the triangle inequality to Eq. (21) we have,

$$\mathcal{L}_{\text{comp}} \le \underbrace{\mathcal{W}_2\left(p(\boldsymbol{X} \mid C), p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C)\right)}_{\text{Distribution matching}} + \underbrace{\mathcal{W}_2\left(p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C), \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X})}{p_{\boldsymbol{\theta}}(C)} \prod_i^n \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_i) p_{\boldsymbol{\theta}}(C_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{X})}\right)}_{\text{Conditional Independence}} \tag{22}$$

(Kwon et al., 2022) showed that under some conditions, the Wasserstein distance between $p_0(\boldsymbol{X}), q_0(\boldsymbol{X})$ is upper bounded by the square root of the score-matching objective. Rewriting Equation 16 from (Kwon et al., 2022)

$$\mathcal{W}_2\left(p_0(\boldsymbol{X}), q_0(\boldsymbol{X})\right) \le K\sqrt{\mathbb{E}_{p_0(\boldsymbol{X})}\left[||\nabla_{\boldsymbol{X}} \log p_0(\boldsymbol{X}) - \nabla_{\boldsymbol{X}} \log q_0(\boldsymbol{X})||_2^2\right]} \tag{23}$$

Distribution matching Following Eq. (23) result, the first term in Eq. (22), replacing p_0 as p and q_0 as p_θ will result in

$$\begin{aligned}
\mathcal{W}_2\left(p(\boldsymbol{X} \mid C), p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C)\right) &\le K_1\sqrt{\mathbb{E}_{p_0(\boldsymbol{X})}\left[||\nabla_{\boldsymbol{X}} \log p(\boldsymbol{X} \mid C) - \nabla_{\boldsymbol{X}} \log p_\theta(\boldsymbol{X})||_2^2\right]} \\
&= K_1\sqrt{\mathcal{L}_{\text{score}}}
\end{aligned} \tag{24}$$

Conditional Independence Following Eq. (23) result, the second term in Eq. (22), replacing p_0 as p_θ and $q_0(\boldsymbol{X})$ as $\frac{p_{\boldsymbol{\theta}}(\boldsymbol{X})}{p_{\boldsymbol{\theta}}(C)}\prod_i^n \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X}|C_i)p_{\boldsymbol{\theta}}(C_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{X})}$

$$\mathcal{W}_2\left(p_{\boldsymbol{\theta}}(\boldsymbol{X}\mid C), \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X})}{p_{\boldsymbol{\theta}}(C)}\prod_i^n \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X}\mid C_i)p_{\boldsymbol{\theta}}(C_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{X})}\right)$$

$$\leq \sqrt{\mathbb{E}\|\nabla_{\mathbf{X}}\log p_\theta(\boldsymbol{X}\mid C) - \nabla_{\mathbf{X}}\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X})}{p_{\boldsymbol{\theta}}(C)}\prod_i^n \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X}\mid C_i)p_{\boldsymbol{\theta}}(C_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{X})}\|_2^2}$$

Further simplifying and incorporating $\nabla_{\mathbf{X}}\log p_\theta(C_i) = 0$ and $\nabla_{\mathbf{X}}\log p_\theta(C) = 0$ will result in

$$\begin{aligned}
&\mathcal{W}_2\left(p_{\boldsymbol{\theta}}(\boldsymbol{X}\mid C), \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X})}{p_{\boldsymbol{\theta}}(C)}\prod_i^n \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X}\mid C_i)p_{\boldsymbol{\theta}}(C_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{X})}\right)\\
&\leq K_2\sqrt{\underbrace{\mathbb{E}\|\nabla_{\mathbf{X}}\log p_\theta(\boldsymbol{X}\mid C) - \nabla_{\mathbf{X}}\log p_\theta(\boldsymbol{X}) - \sum_i\left[\nabla_{\mathbf{X}}\log p_\theta(\boldsymbol{X}\mid C_i) - \nabla_{\mathbf{X}}\log p_\theta(\boldsymbol{X})\right]\|_2^2}_{\mathcal{L}_{\text{CI}}}}\\
&= K_2\sqrt{\mathcal{L}_{\text{CI}}}
\end{aligned} \tag{25}$$

Substituting Eq. (24), Eq. (25) in Eq. (22) will result in our final learning objective

$$\mathcal{L}_{\text{comp}} \leq K_1\sqrt{\mathcal{L}_{\text{score}}} + K_2\sqrt{\mathcal{L}_{\text{CI}}} \tag{26}$$

where K_1, K_2 are positive constants, i.e., the conditional independence objective $\mathcal{L}_{\text{CI}}$ is incorporated alongside the existing score-matching loss $\mathcal{L}_{\text{score}}$.

Note that Eq. (25) is the Fisher divergence between the joint $p_\theta(\boldsymbol{X}\mid C)$ and the causal factorization $\frac{p_\theta(X)}{p_\theta(C)}\prod_i \frac{p_\theta(\boldsymbol{X}|C_i)p_\theta(C_i)}{p_\theta(\boldsymbol{X})}$. From the properties of Fisher divergence (Sánchez-Moreno et al., 2012), $\mathcal{L}_{\text{CI}} = 0$ iff $p_\theta(\boldsymbol{X}\mid C) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X})}{p_{\boldsymbol{\theta}}(C)}\prod_i^n \frac{p_{\boldsymbol{\theta}}(\boldsymbol{X}|C_i)p_{\boldsymbol{\theta}}(C_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{X})}$ and further implying, $\prod_i p_\theta(C_i\mid\boldsymbol{X}) = p_{\text{train}}(C\mid\boldsymbol{X})$

When $L_{\text{comp}} = 0$: $P_\theta(\boldsymbol{X}\mid C) = P_{\text{train}}(\boldsymbol{X}\mid C) = P(\boldsymbol{X}\mid C)$, and $\prod_i p_\theta(C_i\mid\boldsymbol{X}) = p_{\text{train}}(C\mid\boldsymbol{X})$. This implies that the learned marginals obey the causal independence relations from the data-generation process, leading to more accurate marginals.

B.3 Connection to compositional generation from first principles

Compositional generation from first principles Wiedemer et al. (2024) have shown that restricting the function to a certain compositional form will perform better than a single large model. In this section, we show that, by enforcing conditional independence, we restrict the function to encourage compositionality.

Let $c_1, c_2, \ldots, c_n$ be independent components such that $c_1, c_2, \ldots, c_n \in \mathbb{R}$. Consider an injective function $f : \mathbb{R}^n \to \mathbb{R}^d$ defined by $f(c) = x$. If the components, c are conditionally independent given x the cumulative functions, F must satisfy the following constraint:

$$F_{C_i,C_j,\ldots,C_n|X=x}(c_i, c_j, \ldots, c_n) = \prod_i F_{C_i|X=x}(c_i) \tag{27}$$

$F^{-1}_{C_i,C_j,\ldots,C_n|X=x}(x) = \inf\{c_i, c_j, \ldots, c_n \mid F(c_i, c_j, \ldots, c_n) \geq x\}$, where $F^{-1}_{c_i,c_j,\ldots,C_n|X=x}$ is a generalized inverse distribution function.

$$\begin{aligned}
f(c_i, c_j, \ldots, c_n) &= (f\circ F^{-1}_{c_i,c_j,\ldots,C_n|X=x})(\prod_i F_{C_i|X=x}(c_i))\\
&= (f\circ F^{-1}_{c_i,c_j,\ldots,C_n|X=x}\circ e)(\sum_i \log F_{C_i|X=x}(c_i))\\
&= g(\sum_i \phi_i(c_i))
\end{aligned}$$

Therefore, we are restricting f to take a certain functional form. However, it is difficult to show that the data generating process, f, meets the rank condition on the Jacobian for the sufficient support assumption Wiedemer et al. (2024), which is also the limitation discussed in their approach. Therefore, we cannot provide guarantees. However, this section provides a functional perspective of CoIND.

C Practical Considerations

To facilitate scalability and numerical stability for optimization, we introduce two approximations to the upper bound of our objective function Eq. (5).

C.1 Scalability of $\mathcal{L}_{\text{CI}}$

A key computational challenge posed by Eq. (7) is that the number of model evaluations grows linearly with the number of attributes. The Eq. (7) is derived from conditional independence formulation as follows:

$$p_\theta(C \mid X) = \prod_i p_\theta(C_i \mid X). \tag{28}$$

By applying Bayes' theorem to all terms, we obtain,

$$\frac{p_\theta(\boldsymbol{X} \mid C)p_\theta(C)}{p_\theta(X)} = \prod_i \frac{p_\theta(\boldsymbol{X} \mid C_i)p_\theta(C_i)}{p_\theta(\boldsymbol{X})} \tag{29}$$

Note that this formulation is equal to the causal factorization. From this, by applying logarithm and differentiating w.r.t. $\boldsymbol{X}$, we derive the score formulation.

$$\nabla_{\boldsymbol{X}} \log p_\theta(\boldsymbol{X} \mid C) = \nabla_{\boldsymbol{X}} \log \sum_i p_\theta(\boldsymbol{X} \mid C_i) - \nabla_{\boldsymbol{X}} \log p_\theta(\boldsymbol{X}) \tag{30}$$

The L_2 norm of the difference between LHS and RHS of the objective in Eq. (30) is given by, which forms our $\mathcal{L}_{\text{CI}}$ objective.

$$\mathcal{L}_{\text{CI}} = \|\nabla_{\boldsymbol{X}} \log p_\theta(\boldsymbol{X} \mid C) - \left(\nabla_{\boldsymbol{X}} \log \sum_i p_\theta(\boldsymbol{X} \mid C_i) - \nabla_{\boldsymbol{X}} \log p_\theta(\boldsymbol{X})\right)\|_2^2 \tag{31}$$

Due to the $\sum_i$, in the equation, the number of model evaluations grows linearly with the number of attributes (n). This $\mathcal{O}(n)$ computational complexity hinders the approach's applicability at scale. To address this, we leverage the results of (Hammond & Sun, 2006), which shows conditional independence is equivalent to pairwise independence under large n to reduce the complexity to $\mathcal{O}(1)$ in expectation. This allows for a significant improvement in scalability while maintaining computational efficiency. Using this result, we modify Eq. (28) to:

$$p_\theta(C_i, C_j \mid X) = p_\theta(C_i \mid X)p_\theta(C_j \mid X). \quad \forall i, j$$

Accordingly, we can simplify the loss function for conditional independence as follows:

$$\mathcal{L}_{\text{CI}} = \mathbb{E}_{p(\boldsymbol{X},C)}\mathbb{E}_{j,k}\|\nabla_X[\log p_\theta(X|C_j, C_k) - \log p_\theta(X|C_j) - \log p_\theta(X|C_k) + \log p_\theta(X)]\|_2^2. \tag{32}$$

In score-based models, which are typically neural networks, the final objective is given as:

$$\mathcal{L}_{\text{CI}} = \mathbb{E}_{p(\boldsymbol{X},C)}\mathbb{E}_{j,k}\|s_{\boldsymbol{\theta}}(\boldsymbol{X}, C_j, C_k) - s_{\boldsymbol{\theta}}(\boldsymbol{X}, C_j) - s_{\boldsymbol{\theta}}(\boldsymbol{X}, C_k) + s_{\boldsymbol{\theta}}(\boldsymbol{X}, \varnothing)\|_2^2 \tag{33}$$

where $s_{\boldsymbol{\theta}}(\cdot) \coloneqq \nabla_{\boldsymbol{X}} \log p_\theta(\cdot)$ is the score of the distribution modeled by the neural network. We leverage classifier-free guidance to train the conditional score $s_{\boldsymbol{\theta}}(\boldsymbol{X}, C_i)$ by setting $C_k = \varnothing$ for all $k \neq i$, and likewise for $s_{\boldsymbol{\theta}}(\boldsymbol{X}, C_i, C_j)$, we set $C_k = \varnothing$ for all $k \notin \{i, j\}$.

C.2 Simplification of Theoretical Loss

In Eq. (5), we showed that the 2-Wassertein distance between the true joint distribution $p(\boldsymbol{X} \mid C)$ and the causal factorization in terms of the marginals $p(\boldsymbol{X} \mid C_i)$ is upper bounded by the weighted sum of the square roots of $\mathcal{L}_{\text{score}}$ and $\mathcal{L}_{\text{CI}}$ as $\mathcal{L}_{\text{comp}} \leq K_1\sqrt{\mathcal{L}_{\text{score}}} + K_2\sqrt{\mathcal{L}_{\text{CI}}}$. In practice, however, we minimized a simple weighted sum of $\mathcal{L}_{\text{score}}$ and $\mathcal{L}_{\text{CI}}$, given by $\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{score}} + \lambda\mathcal{L}_{\text{CI}}$ as shown in Eq. (8) instead of Eq. (5). We used Eq. (8) to avoid the instability caused by larger gradient magnitudes (due to the square root). Eq. (8) also provided the following practical advantages: **(1)** the simplicity of the loss function that made hyperparameter tuning easier, and **(2)** the similarity of Eq. (8) to the loss functions of pre-trained diffusion models allowing us to reuse existing hyperparameter settings from these models. We did not observe any significant difference in conclusion between the models trained on Eq. (5) and Eq. (8) as shown in Tabs. 3 and 4. Both approaches significantly outperformed the baselines.

Support	Method	JSD ↓	∧ (CS) ↑
	LACE	-	99.04
	Composed GLIDE	0.16	97.40
Uniform	Theoretical Eq. (5)	0.12	98.44
	CoInD ($\lambda = 0.2$)	0.14	99.50
	CoInD ($\lambda = 1.0$)	**0.11**	**99.86**
	LACE	-	51.70
	Composed GLIDE	0.33	82.60
Non-uniform	Theoretical Eq. (5)	0.17	96.88
	CoInD ($\lambda = 1.0$)	**0.11**	**99.84**
	LACE	-	22.94
	Composed GLIDE	2.76	17.90
Partial	Theoretical Eq. (5)	1.11	23.44
	CoInD ($\lambda = 1.0$)	**1.07**	**55.16**

Table 3: Results on Colored MNIST ($K_1 = 1, K_2 = 0.1$)

Support	Method	JSD ↓	R^2 ↑
	LACE	-	0.97
	Composed GLIDE	0.302	0.94
Uniform	Theoretical Eq. (5)	0.270	0.98
	CoInD ($\lambda = 1.0$)	**0.215**	**0.98**
	LACE	-	0.88
	Composed GLIDE	0.503	0.86
Partial	Theoretical Eq. (5)	0.450	0.93
	CoInD ($\lambda = 1.0$)	**0.287**	**0.97**

Table 4: Results on Shapes3D ($K_1 = 1, K_2 = 0.1$)

C.3 Choice of Hyperparameter λ

Effect of λ on the Learned Conditional Independence.

CoInD enforces conditional independence between the marginals of the attributes learned by the model by minimizing $\mathcal{L}_{\text{CI}}$ defined in Eq. (33). Here, we investigate the effect of $\mathcal{L}_{\text{CI}}$ on the effectiveness of compositionality by varying its strength through λ in Eq. (8). Figure 5 plots JSD and CS as functions of λ for models trained on the Colored MNIST dataset under the diagonal partial support setting.

When $\lambda = 0$, training relies solely on the score matching loss, resulting in higher conditional dependence between $C_i \mid \boldsymbol{X}$. As λ increases, CS improves since ensuring conditional independence between the marginals also encourages more accurate learning of the true marginals. However, when λ takes large values, the model learns truly independent conditional distribution $C \mid \boldsymbol{X}$ but effectively ignores the input compositions and generates samples based solely on the prior distribution $p_{\boldsymbol{\theta}}(\boldsymbol{X})$. As a result, CS drops.

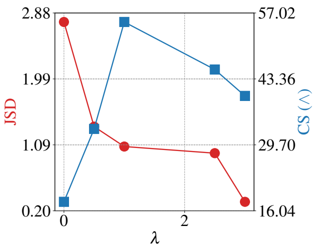


Figure 5: Effect of λ on compositionality under diagonal partial support on the Colored MNIST dataset.

The value for the hyperparameter λ is chosen such that the gradients from the score-matching objective L_{score} and the conditional independence objective L_{CI} are balanced in magnitude. One way to choose λ is by training a vanilla diffusion model and setting $\lambda = \frac{L_{score}}{L_{CI}}$. As a rule of thumb, we recommend the simplified setting: $\lambda = L_{score} \times 4000$. We used two values for λ in our experiments and noticed that they gave similar results, indicating that the approach was stable for various values of λ.

D Experiment Details

D.1 CoInD Algorithm

Algorithm 1 CoInD Training

```
repeat
    $(\mathbf{c}, \mathbf{x}_0) \sim p_{\text{train}}(\mathbf{c}, x)$
    $c_k \leftarrow \varnothing$ with probability $p_{uncond}$    ▷ Set element of index,$k$ i.e, $c_k$ to $\varnothing$ with $p_{uncond} \forall k \in [0, N]$ probability
    $i \sim \text{Uniform}(\{0, \ldots, N\}), j \sim \text{Uniform}(\{0, \ldots, N\} \setminus \{i\})$    ▷ Select two random attribute indices
    $t \sim \text{Uniform}(\{1, \ldots, T\})$
    $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    $x_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}$
    $\mathbf{c}^i, \mathbf{c}^j, \mathbf{c}^{i,j} \leftarrow \mathbf{c}$
    $\mathbf{c}^i \leftarrow \{c_k = \varnothing \mid k \neq i\}, \mathbf{c}^j \leftarrow \{c_k = \varnothing \mid k \neq j\}, \mathbf{c}^{i,j} \leftarrow \{c_k = \varnothing \mid k \notin \{i, j\}\}, \mathbf{c}^{\varnothing} \leftarrow \varnothing$
    $L_{CI} = ||\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c}^i) + \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c}^j) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c}^{i,j}) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c}^{\varnothing})||_2^2$
    Take gradient descent step one
        $\nabla_\theta[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c})\|^2 \; +\lambda L_{CI} \; ]$
until converged
```

To compute pairwise independence in a scalable fashion, we randomly select two attributes, i and j, for a sample in the batch and enforce independence between them. As the score in Eq. (12) is given by $\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}}$. The final equation for enforcing $\mathcal{L}_{\text{CI}}$ will be:

$$L_{CI} = \frac{1}{1 - \bar{\alpha}_t} \left\| \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c}^i) + \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c}^j) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c}^{i,j}) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c}^{\varnothing}) \right\|_2^2$$

We follow Ho et al. (2020) to weight the term by $1 - \bar{\alpha}_t$. This results in an algorithm for CoInD, requiring only a few modifications of lines from (Ho & Salimans, 2022), highlighted below. **Practical Implementation** In our experiments, we have used $p_{\text{uncond}} = 0.3$ and for Shapes3D instead of enforcing $C_i \perp\!\!\!\perp C_j \mid X$, for all i, j enforcing $C_i \perp\!\!\!\perp C_{-i} \mid X$ for all i have led to slightly better results.

D.2 Details of Compositionality Task

Composition Sampling To evaluate the composition, we apply compose all the attributes to generate a respective image.

Consider an image from the Shapes3D dataset. The image is generated by some function, f, with the input $c = [\begin{matrix} 6 & 8 & 4 & 6 & 2 & 11 \end{matrix}]$. The following image can be queried using the expression $C_1 = 6 \wedge \ldots \wedge C_6 = 11$. We follow Equation Eq. (1) to sample from the above composition. To reiterate, for the $\wedge$ composition task on Shapes3D, the sampling equation is given by $\nabla_{\boldsymbol{X}} p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_1 = 6 \wedge \ldots \wedge C_6 = 11)$:

$$\nabla_{\mathbf{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}) + \sum_i \left[\nabla_{\mathbf{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_i) - \nabla_{\mathbf{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X})\right] \quad (34)$$

Similarly, to evaluate the composition for the Colored MNIST dataset.

Evaluations are strictly restricted to unseen compositions under orthogonal partial support for Shapes3D and under diagonal partial support for Colored MNIST. This approach allows us to explore how effectively the model handles unseen image generation. Additionally, we evaluate compositions observed during training with less frequency under non-uniform support.

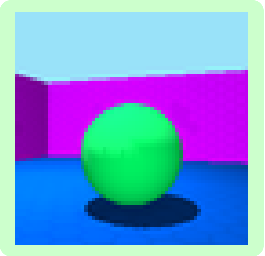


Figure 6: Image from Shapes3d with attributes $c = [6, 8, 4, 6, 2, 11]$

D.3 Training details, Architecture, and Sampling

Training Composed GLIDE & CoInD We train the diffusion model using the DDPM noise scheduler. The model architecture and hyperparameters used for all experiments are detailed in Tab. 5.

Training LACE The LACE method involves training multiple energy-based models for each attribute and sampling according to compositional equations. However, we use score-based models instead. We follow the architecture outlined in Tab. 5 for each attribute to train multiple score-based models. For Colored MNIST, which has two attributes, we create two models—one for each attribute—using the same architecture as other methods, effectively doubling the model size. Similarly, for Shapes3D with six attributes, we develop six models. We reduce the Block Out Channels for each attribute model to fit these into memory while keeping all other hyperparameters consistent. Since we train a single model per attribute, we do not match the joint distribution, preventing us from evaluating it and measuring the JSD.

Sampling To generate samples for a given composition, we sample from equations from App. D.2 using DDIM (Song et al., 2020) with 150 steps.

Hyperparameter	Colored MNIST		Shapes3D	
	CoInD & Composed GLIDE	LACE	CoInD & Composed GLIDE	LACE
Optimizer	AdamW	AdamW	AdamW	AdamW
Learning Rate	2.0×10^{-4}	2.0×10^{-4}	2.0×10^{-4}	2.0×10^{-4}
Num Training Steps	50000	100000	100000	100000
Train Noise Scheduler	DDPM	DDPM	DDPM	DDPM
Train Noise Schedule	Linear	Linear	Linear	Linear
Train Noise Steps	1000	1000	1000	1000
Sampling Noise Schedule	DDIM	DDIM	DDIM	DDIM
Sampling Steps	150	150	150	150
Model	U-Net	U-Net	U-Net	U-Net
Layers per block	2	2	2	2
Beta Schedule	Linear	Linear	Linear	Linear
Sample Size	28x3x3	28x3x3	64x3x3	64x3x3
Block Out Channels	[56,112,168]	[56,112,168]	[56,112,168,224]	**[56,112,168]**
Dropout Rate	0.1	0.1	0.1	0.1
Attention Head Dimension	8	8	8	8
Norm Num Groups	8	8	8	8
Number of Parameters	$8.2M$	$8.2M \times 2$	$17.2M$	$8.2M \times 6$

Table 5: Hyperparameters for Colored MNIST and Shapes3D used by CoInD, Composed GLIDE, and LACE

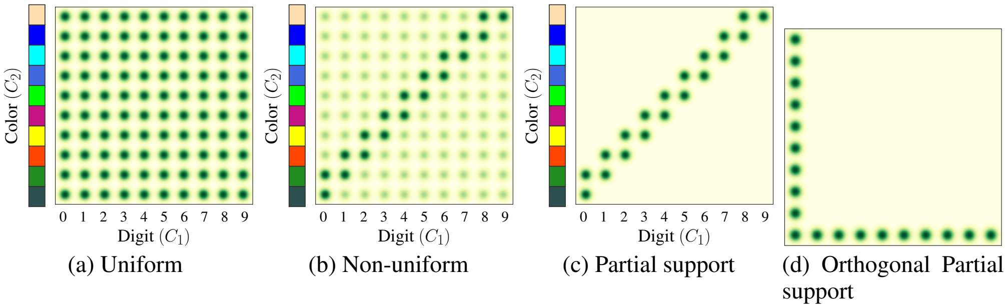
CelebA To generate CelebA images, we scale the image size to 128×128. We use the latent encoder of Stable Diffusion 3 (SD3) to encode the images to a latent space and perform diffusion in the latent space. The architecture is similar to the Colored MNIST and Shapes3D, except that Block out Channels are scaled as [224, 448, 672, 896]. We use a learning rate of 1.0×10^{-4} and train the model for 500,000 steps on one A6000 GPU.

FID Measure To evaluate both the generation quality and how well the generated samples align with the natural distribution of 'smiling male celebrities', we use the FID metric (Seitzer, 2020). Notably, we calculate the FID score specifically on the subset of 'smiling male celebrities,' as our primary objective is to assess the model's ability to generate these unseen compositions. We generate 3000 samples to evaluate FID.

D.4 Analytical Forms of Support Settings

Below are the analytical expressions for the densities under the various support settings that we considered in the paper. Let n_i be the number of categories for the attribute C_i. For non-uniform and diagonal partial support settings, we assume that $n_i = n_j = n, \forall i, j, i \neq j$.

- **Uniform setting:** $p(C_i = c_1) = \frac{1}{n_i}$ and $p(C_i = c_1, C_j = c_2) = p(C_i = c_1)p(C_j = c_2) = \frac{1}{n_i n_j}$.



(a) Uniform (b) Non-uniform (c) Partial support (d) Orthogonal Partial support

Figure 7: **Generative Modeling of compositions under various supports.**

- **Orthogonal support setting:** $p(C_i = c_1, C_j = c_2) = \begin{cases} \frac{1}{n_i+n_j-1}, & c_1 = 0 \text{ or } c_2 = 0 \\ 0, & \text{otherwise} \end{cases}$
- **Non-uniform setting:** $p(C_i = c_1, C_j = c_2) = \begin{cases} a, & c_2 \leq c_1 \leq c_2 + 1 \\ b, & \text{otherwise} \end{cases}$, where $\frac{1}{n^2} \leq b < a \leq \frac{1}{2n-1}$.
- **Diagonal partial support setting:** $p(C_i = c_1, C_j = c_2) = \begin{cases} \frac{1}{2n-1}, & c_2 \leq c_1 \leq c_2 + 1 \\ 0, & \text{otherwise} \end{cases}$.

Computing mutual information between attributes for all supports.

D.5 Datasets

Colored MNIST Dataset In Section § 1, we introduced the Colored MNIST dataset. Here, we will detail the dataset generation process. We selected 10 visually distinct colors [1], taking the value $C_2 \in [0, 9]$. The dataset is constructed by coloring the grayscale images from MNIST by converting them into three channels and applying one of the ten colors to non-zero grayscale values.

The training data is composed of three types of support:

- **Uniform Support**: A digit and a color are randomly selected to create an image.
- **Diagonal Partial Support**: A digit is selected, and during training, it is only assigned one of two colors, $C_2 \in \{d, d+1\}$, except for 9, which only takes one color. This creates a dataset where compositions observed during training are along the diagonal of the $\mathcal{C}$ space, meaning each digit is seen only with its corresponding colors.
- **Non-uniform Support**: All compositions are observed, but combining a digit and its corresponding colors occurs with a higher probability (0.5). The remaining color space is distributed evenly among other colors, resulting in approximately a 0.25 probability for each corresponding color and a 0.0625 probability for each remaining color.

Shapes3D Full support for Shapes3D consists of all samples from the dataset. For orthogonal support, we use the composition split of Shapes3D as described by Schott et al.., whose code is publicly available [2].

D.6 Accuracy of classifiers for Conformity Score (CS)

The effectiveness of the ResNet-18 classifier in predicting the attributes is reported in Table 8 below.

[1] `https://mokole.com/palette.html`

[2] https://github.com/bethgelab/InDomainGeneralizationBenchmark

Feature	Attributes	Possible Values	Accuracy
C_1	Digit	0-9	98.93
C_2	color	10 values	100

(a) Colored MNIST Dataset

Feature	Attributes	Possible Values	Accuracy
C_1	Gender	$\{0,1\}$	98.2
C_2	Smile	$\{0,1\}$	92.1

(b) CelebA Dataset

Feature	Attributes	Possible Values	Accuracy
C_1	floor hue	10 values in [0, 1]	100
C_2	wall hue	10 values in [0, 1]	100
C_3	object hue	10 values in [0, 1]	100
C_4	scale	8 values in [0, 1]	100
C_5	shape	4 values in [0-3]	100
C_6	orientation	15 values in [-30, 30]	100

(c) Shapes3D Dataset

Figure 8: Independent attribute, their possible values, and the classifier accuracy in estimating them for different datasets

D.7 Measuring Diversity in Attributes

To achieve explicit control over certain attributes during the generation process, these attributes must vary independently. Therefore, an ideal generative model must be able to produce samples where all except the controlled attributes take diverse values. This diversity can be measured by the entropy of the uncontrolled attributes in the generated samples, where higher entropy suggests greater diversity. Therefore, the accurate generation of controlled and diverse uncontrolled attributes indicates that the model has successfully learned the correct marginal likelihood of the controlled attributes.

For example, consider the generation of colored MNIST digits. In this case, controllability means that the model has learned that digit and color attributes are independent. When prompted to generate a specific digit (controlled attribute), the model should generate this digit in all possible colors (uncontrolled attribute) with equal likelihood, implying maximum entropy for the color attribute and diverse generation. We measure this entropy by generating samples $x^i \sim p_\theta(X \mid c_1 = 4)$ and passing them through a near-perfect classifier to obtain the color predictions $p(\hat{C_2}) = p(\phi_2(x^i))$. The diversity is then quantified as: $H = \mathbb{E}_{\hat{c_2} \sim p(\hat{C_2})} \left[\log_2 p(\hat{c_2})\right]$

Ensuring diversity through explicit control has applications in bias detection and mitigation in generative models. For example, a biased model may generate images of predominantly male doctors when asked to generate images of “doctors”. Ensuring diversity in uncontrolled attributes like gender or race can limit such biases.

E CoInD for Face Image Generation

In § 5, we demonstrated that CoInD outperforms baseline methods on the unseen compositionality task using synthetic datasets. In App. E.1, we showcase the success of CoInD in generating face images from the CelebA dataset (Liu et al., 2015), where CoInD demonstrates superior control over attributes compared to the baseline. CoInD also allows us to adjust the strength of various attributes and thus provides more fine-grained control over the compositional attributes, as shown in App. E.2.

Problem Setup We choose the CelebA dataset to evaluate CoInD’s ability to generate real-world images. We choose the binary attributes “smiling” and “gender” as the attributes we wish to control. During training, all combinations of these attributes except gender = “male” and smiling = “true” are observed, similar to the orthogonal support shown in Fig. 7d. During inference, the model is tasked to generate images with the attribute combination gender = “male” and smiling = “true”, which was not observed during training.

Metrics Similar to the experiments on the synthetic image datasets in § 5, we assess the accuracy of the generation w.r.t. the input desired attribute combination CS (conformity score). We also measure the violation of the learned conditional independence using JSD. In addition to CS, we compute FID (Fréchet inception distance) between the generated images and the real samples in the CelebA dataset where gender = “male” and smiling = “true”. A lower FID implies that the distribution of generated samples is closer to the real distribution of the images in the validation dataset.

E.1 CoInD can successfully generate real-world face images

Fig. 4a shows the quantitative results of CoInD and Composed GLIDE trained from scratch in the tasks of joint sampling and $\wedge$ composition. Similar to our observations from previous experiments, CoInD achieves better CS in both tasks by learning accurate marginals as demonstrated by lower JSD. When sampled from the joint likelihood, CoInD achieves a nearly $4\times$ improvement in CS over the baseline, while it achieves $> 10\%$ improvement in CS over the baseline, for $\wedge$ compositionality.

E.2 CoInD provides fine-grained control over attributes

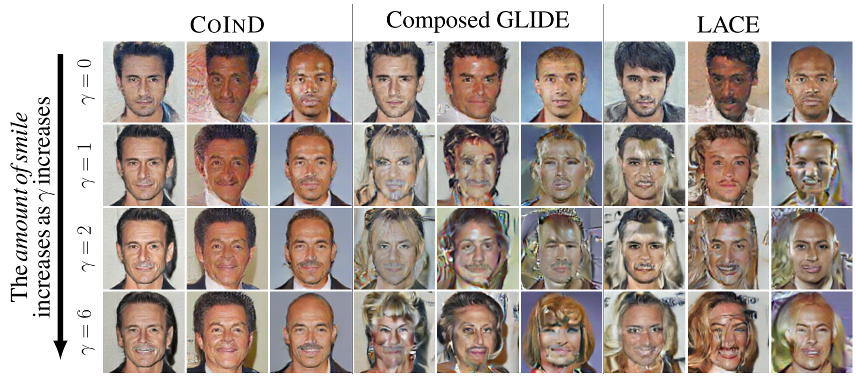


Figure 9: By adjusting γ, CoInD allows us to the vary the *amount of “smile”* in the generated images. However, Composed GLIDE associates the smile attribute with the gender attribute due to their association in the training data. Hence, the images generated by Composed GLIDE contain gender-specific attributes such as long hair and earrings.

So far, we studied the capabilities of CoInD to dictate the presence and absence of attributes in the task of controllable image generation. However, there are applications where we desire fine-grained control over the attributes. Specifically, we may want to control the *amount of each attribute* in the generated sample. We can mathematically formulate this task by revisiting the formulation of expressions of attributes in terms of the score functions of marginal likelihood. As an example, the $\wedge$ operation can be written as,

$$\nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_1 \wedge C_2) = \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_1) + \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_2) - \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X})$$

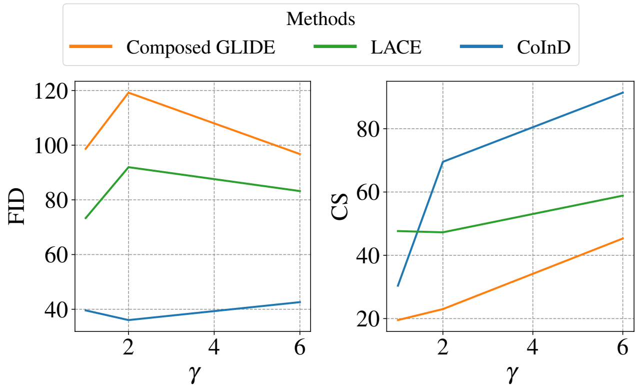
Here, to adjust the amount of attribute added to the generated sample, we can weigh the score functions using some scalar γ, as follows,

$$\nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_1) + \gamma \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X} \mid C_2) - \gamma \nabla_{\boldsymbol{X}} \log p_{\boldsymbol{\theta}}(\boldsymbol{X}) \tag{35}$$

where γ controls for the amount of C_2 attribute.

Fig. 9 shows the effect of increasing γ to adjust the amount of smiling in the generated image. Ideally, we expect increasing γ to increase the amount of smiling without affecting the gender attribute. When $\gamma = 0$ (top row), both CoInD and Composed GLIDE generate images of men who are not smiling. As γ increases, we notice that the samples generated by CoInD show an increase in the amount of smiling, going from a short smile to a wider smile to one where teeth are visible. Note that the training dataset did not include any images of smiling men or fine-grained annotations for the amount of smiling in each image. This conclusion is strengthened by Fig. 10b that shows an increase in CS when γ increases. CS increases when it is easier for the smile classifier to detect the smile. CoInD provides this fine-grained control over the smiling attribute without any effect on the realism of the images, as shown by the minimal changes in FID in Fig. 10a.

In contrast, the images generated by Composed GLIDE show an increase in the amount of smiling while adding gender-specific attributes such as long hair and makeup. We conclude that, by strictly



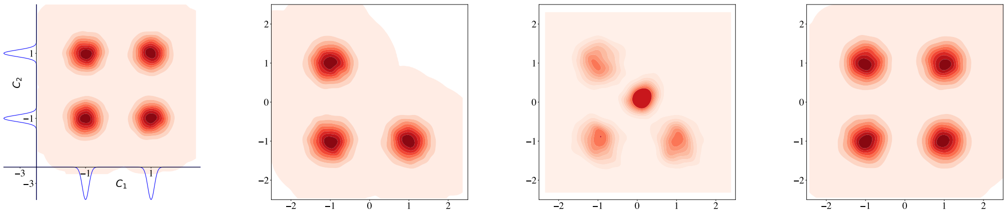
(a) Variation of FID with γ

(b) Variation of CS with γ

Figure 10: **Effect of γ on FID and CS:** Varying the amount of smile in a generated image through γ does not affect the FID of CoInD. However, the smiles in the generated images become more apparent, leading to easier detection by the smile classifier and improved CS.

enforcing a conditional independence loss between the attributes, CoInD provides fine-grained control over the attributes, allowing us to adjust the intensity of the attribute in the image without additional training. As shown in Fig. 4a, CoInD outperforms the baselines for generating unseen compositions. Tuning γ further improves the generation.

F 2D Gaussian: Workings of CoInD in closed form



(a) True underlying data distribution

(b) Training data Orthogonal Support

(c) Conditional distribution learned by vanilla diffusion objective

(d) Conditional distribution learned by CoInD

Figure 11: CoInD respects underlying independence conditions thereby generating true data distribution (d).

The underlying data is generated by two independent attributes, C_1 and C_2. The observed variable $\mathbf{X}$ is defined as:

$$\mathbf{X} = f(\mathbf{C}_1) + f(\mathbf{C}_2) \tag{36}$$

where $f(c_i) = c_i + \sigma\epsilon$, and $\epsilon \sim \mathcal{N}(0, I)$ represents Gaussian noise. For simplicity, C_1 and C_2 are binary variables taking values in $\{-1, +1\}$. The function $f(C_1)$ results in a mixture of Gaussians with means $[-1 \quad 0]$ and $[+1 \quad 0]$. These are represented along the x-axis in Figure 11a. Similarly, $f(C_2)$ produces a mixture of Gaussians with means:$[0 \quad -1]$ and $[0 \quad +1]$. These are displayed along the y-axis in Figure 11a. The combination of C_1 and C_2 independently generates as Eq. (36) This results in a two-dimensional Gaussian mixture, as illustrated in Figure 11. We consider orthogonal support, where attribute combinations of $(C_1, C_2) \in \{(-1,-1), (-1,+1), (+1,-1)\}$, and the model is tasked to generate unseen combination of $(+1,+1)$. Also as a reminder that assumptions mentioned in § 2 are satisfied. (1) C_1, C_2 independently generate $\boldsymbol{X}$, and (2) all possibles values for every attribute are present at-least observed during training. Let score is given by $s_{+1,+1}$ represents $\nabla_{\mathbf{X}} \log p(X \mid C_1 = 1, C_2 = 1)$ and likewise $s_{1,\varnothing}$ represents $\nabla_{\mathbf{X}} \log p(X \mid C_1)$ To sample for the unseen compositions of (1,1) we use Eq. (1) to

$$s_{1,1}(x) = s_{1,\varnothing}(x) + s_{\varnothing,1}(x) - s_{\varnothing,\varnothing}(x) \tag{37}$$

Training diffusion model (score) objective involves computing score functions from the training data, which will give us the following terms in closed form. For example $s_{1,\varnothing}(x)$ is training using only $+1, -1$ combinations present during training. which is nothing but a gaussian at $+1, -1$ and the score of the gaussian is given in closed form.

$$\begin{aligned} s_{1,\varnothing}(x) &= \frac{\mu_{1,-1} - x}{\sigma^2} \\ s_{\varnothing,1}(x) &= \frac{\mu_{1,-1} - x}{\sigma^2} \end{aligned}$$

$s_{\varnothing,\varnothing}(x)$ is a mixture of gaussian with means around 3 Gaussians present during training. The score of the mixture gaussian as:

$$s_{\varnothing,\varnothing}(x) = \frac{\sum_i \mathcal{N}(x; \mu_i, \sigma^2 I) \left[\frac{\mu_i - x}{\sigma^2}\right]}{\sum_i \mathcal{N}(x; \mu_i, \sigma^2 I)}$$

Now leveraging Langevin dynamics Eq. (10) will generate the Fig. 11c as the distribution of $P(X \mid C_1 = +1, C_2 = +1)$ is incorrect (strong red blob between the $(+1, -1), (-1, +1)$ instead of gaussian at (+1,+1)). This is due to incorect modelling of the distributions $s_{1,\varnothing}(x), s_{\varnothing,1}(x), s_{\varnothing,\varnothing}(x)$. However, COIND does not explicitly model $s_{1,\varnothing}$, instead learn joint $s_{-1,-1}(x), s_{+1,-1}(x), s_{-1,+1}(x)$ as Gaussians and then combine them using pairwise conditional independence conditions given as:

$$\begin{aligned} s_{-1,-1}(x) &= s_{+1,\varnothing}(x) + s_{\varnothing,+1}(x) - s_{\varnothing,\varnothing}(x) \\ s_{+1,-1}(x) &= s_{+1,\varnothing}(x) + s_{\varnothing,-1}(x) - s_{\varnothing,\varnothing}(x) \\ s_{-1,+1}(x) &= s_{-1,\varnothing}(x) + s_{\varnothing,+1}(x) - s_{\varnothing,\varnothing}(x) \\ s_{+1,1}(x) &= s_{+1,\varnothing}(x) + s_{\varnothing,+1}(x) - s_{\varnothing,\varnothing}(x) \\ &= s_{+1,-1}(x) + s_{-1,+1}(x) - s_{-1,-1}(x) \\ &= \frac{[\mu_{+1,-1} + \mu_{+1,-1} - \mu_{-1,-1}] - x}{\sigma^2} \end{aligned}$$

This shows the workings of COIND and also demonstrates that conditional independence constraints are necessary to learn the underlying distribution and alos with these constraints, diffusion models generate incorrect interpolation for unseen data distributions as shown in Fig. 11c.

G EXTENSION TO GAUSSIAN SOURCE FLOW MODELS

Diffusion models can be viewed as a specific case of flow-based models where: (1) the source distribution is Gaussian, and (2) the forward process follows a predetermined noise schedule.(Lipman et al., 2024). Can we reformulate COIND in terms of velocity rather than score, thereby generalizing it to accommodate arbitrary source distributions and schedules? When the source distribution is gaussian, score and velocity are related by affine transformation as detailed in Tab. 1 of (Lipman et al., 2024).

$$s^t_\theta(x, C_1, C_2) = a_t x + b_t u^t_\theta(x, C_1, C_2) \tag{38}$$

replacing $s^t_\theta(\cdot)$ into Eq. (33)

$$\begin{aligned} \mathcal{L}_{\text{CI}} &= \mathbb{E}_{p(\boldsymbol{X},C),t\sim U[0,1]}\mathbb{E}_{j,k}\|s^t_{\boldsymbol{\theta}}(x, C_j, C_k) - s^t_{\boldsymbol{\theta}}(x, C_j) - s^t_{\boldsymbol{\theta}}(x, C_k) + s^t_{\boldsymbol{\theta}}(x)\|_2^2 \\ &= \mathbb{E}_{p(\boldsymbol{X},C),t\sim U[0,1]}\mathbb{E}_{j,k}\left[b_t^2\|u^t_{\boldsymbol{\theta}}(x, C_j, C_k) - s^t_{\boldsymbol{\theta}}(x, C_j) - u^t_{\boldsymbol{\theta}}(x, C_k) + u^t_{\boldsymbol{\theta}}(x)\|_2^2\right] \end{aligned}$$

However we can ignore b_t^2, weighting for the time step t.

$$\mathcal{L}_{\text{CI}} = \mathbb{E}_{p(\boldsymbol{X},C),t\sim U[0,1]}\mathbb{E}_{j,k}\left[\|u^t_{\boldsymbol{\theta}}(x, C_j, C_k) - u^t_{\boldsymbol{\theta}}(x, C_j) - u^t_{\boldsymbol{\theta}}(x, C_k) + u^t_{\boldsymbol{\theta}}(x)\|_2^2\right] \tag{39}$$

Therefore, if the source distribution is gaussian and for any arbitrary noise schedule, constraint in score translates directly to velocity constraint as given as Eq. (39).