Contrastive Mixture of Posteriors for Counterfactual Inference, Data Integration and Fairness

Adam Foster [1] **Árpi Vezér** [2] **Craig A. Glastonbury** [2 3] **Páidí Creed** [2] **Sam Abujudeh** [2] **Aaron Sim** [2]

Abstract

Learning meaningful representations of data that can address challenges such as batch effect correction and counterfactual inference is a central problem in many domains including computational biology. Adopting a Conditional VAE framework, we show that marginal independence between the representation and a condition variable plays a key role in both of these challenges. We propose the Contrastive Mixture of Posteriors (CoMP) method that uses a novel misalignment penalty defined in terms of mixtures of the variational posteriors to enforce this independence in latent space. We show that CoMP has attractive theoretical properties compared to previous approaches, and we prove counterfactual identifiability of CoMP under additional assumptions. We demonstrate state-of-the-art performance on a set of challenging tasks including aligning human tumour samples with cancer cell-lines, predicting transcriptome-level perturbation responses, and batch correction on single-cell RNA sequencing data. We also find parallels to fair representation learning and demonstrate that CoMP is competitive on a common task in the field.

1. Introduction

Large scale datasets describing the molecular properties of cells, tissues and organs in a state of health and disease are commonplace in computational biology. Referred to collectively as ‘omics data, thousands of features are measured per sample and, as single-cell methodologies have developed, it is now typical to measure such features across 10^5–10^6 observations (Svensson et al., 2018; Regev et al., 2017). Given these two properties of ‘omics data, the need for scalable algorithms to learn meaningful low-dimensional representations that capture the variability of the data has grown. As such, Variational Autoencoders (VAEs) (Kingma & Welling, 2013; Rezende et al., 2014) have become an important tool for solving a range of modelling problems in the biological sciences (Lopez et al., 2018; Way & Greene, 2018; Wang & Gu, 2018; Grønbech et al., 2020; Lotfollahi et al., 2019a;b). One such problem is utilising representations for counterfactual inference, e.g. predicting how a certain cell or cell-type, observed in the control group, would have behaved when exposed to a drug (Lotfollahi et al., 2019a;b; Amodio et al., 2018). Another key problem is removing batch effects—spurious shifts in observations due to differing experimental conditions—from data in order to integrate or compare multiple datasets (Lopez et al., 2018; Johnson et al., 2007; Leek & Storey, 2007; Haghverdi et al., 2018; Warren et al., 2021).

Our approach to these problems is to learn a VAE representation that is marginally independent of a condition variable (e.g. experimental batch, stimulated vs. control). Figure 1 [CoMP] illustrates what this looks like in practice: the complete overlap of the cell populations from different conditions in the latent space. For data integration, the resulting representation perfectly integrates distinct batches, assuming there are no population-level differences between them. To predict the effects of interventions, following Lotfollahi et al. (2019a), we encode control data to representation space, and decode it back to the original space under the stimulated condition. Alignment of control and stimulated cells in representation space isolates the effects of interventions to the decoder network, and is a necessary condition for the encode–swap–decode algorithm to provide correct predictions. This same independence constraint also occurs in fair representation learning, where we seek a representation that cannot be used to recover a sensitive attribute (Zemel et al., 2013; Louizos et al., 2015).

Neither the standard VAE nor the conditional VAE (CVAE) (Sohn et al., 2015) are typically successful at learning representations that achieve this desired independence, as shown in Figure 1. Existing methods use a penalty to

[1] Microsoft Research, Cambridge, UK. This work was completed when AF was an intern at BenevolentAI and a DPhil student at the Department of Statistics, University of Oxford. [2] BenevolentAI, 4–8 Maple Street, London, UK [3] Human Technopole, V.le Rita Levi-Montalcini, 1, 20157 Milano, Italy. Correspondence to: Adam Foster <adam.e.foster@microsoft.com>.

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

encourage the CVAE to learn representations that overlap correctly in latent space, with Maximum Mean Discrepancy (MMD) (Gretton et al., 2012) being the most common, applied in the VFAE (Louizos et al., 2015) and trVAE (Lotfollahi et al., 2019a). These methods, however, suffer from a number of drawbacks: conceptually, they introduce an extraneous discrepancy measure that is not a part of the variational inference framework; practically, they require the choice of, and hyperparameter tuning for, an MMD kernel; empirically, whilst trVAE is a significant improvement over an unconstrained CVAE, Figure 1 [trVAE] shows that it may fail to exactly align different conditions.

To overcome these difficulties, we introduce *Contrastive Mixture of Posteriors (CoMP)*, a new method for learning aligned representations in a CVAE framework. Our method features the novel CoMP misalignment penalty that compels the CVAE to remove batch effects. Inspired by contrastive learning (van den Oord et al., 2018; Chen et al., 2020), the penalty encourages representations from different conditions to be close, whilst representations from the same condition are spread out. To achieve this, we approximate the requisite marginal distributions using mixtures of the variational posteriors themselves, leading to a penalty that does not require an extraneous discrepancy measure or a separately tuned kernel. We prove that the CoMP penalty is a stochastic upper bound on a weighted sum of KL divergences, so minimising the penalty minimises a well-established statistical divergence measure. As shown in Figure 1 [CoMP], our method can achieve visually perfect alignment on a number of real-world biological datasets.

Theoretically, counterfactual inference provides the formal framework to discuss data integration (Bareinboim & Pearl, 2016), perturbation response prediction, and fairness (Kusner et al., 2017). We demonstrate that the constrained CVAE approach is *not* always able to compute counterfactuals, even with infinite data. However, introducing additional assumptions, including non-Gaussianity of the latent distribution, we prove counterfactual identifiability and model consistency in our framework. This begins to provide theoretical grounding, not only for CoMP, but for related methods (Louizos et al., 2015; Lotfollahi et al., 2019a).

We apply CoMP to three challenging biological problems[1]: 1) aligning gene expression profiles between tumours and their corresponding cell-lines (Warren et al., 2021), 2) estimating the gene expression profile of an unperturbed cell as if it *had* been treated with a chemical perturbation (Lotfollahi et al., 2019b), 3) data integration with single-cell RNA-seq (Korsunsky et al., 2019). We show that CoMP outperforms existing methods, achieving state-of-the-art performance on these tasks. We also show that CoMP can learn a representation that is fully independent of a protected attribute (gender) whilst maintaining useful information for other prediction tasks on the UCI Adult Income dataset (Dua & Graff, 2017). CoMP represents a conceptually simple and empirically powerful method for learning aligned representation, opening the door to answering high-value questions in biology and beyond.

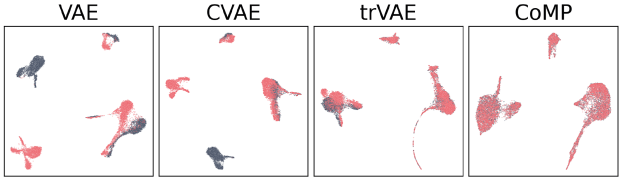

Figure 1. Latent representations of a single-cell gene expression dataset under two conditions: stimulated (red) and not stimulated (black). Full details in Section 6.2. From fully disjointed (VAE) to a well-mixed pair of distributions (CoMP).

2. Background

2.1. Variational Autoencoders and extensions

We begin by assuming that we have n observations $\mathbf{x}_1, \ldots, \mathbf{x}_n$ of an underlying data distribution. For example, $\mathbf{x}_i$ may represent the gene expression profiles of n cells. Variational autoencoders (VAEs) (Kingma & Welling, 2013; Rezende et al., 2014) explain the high-dimensional observations $\mathbf{x}_i$ using low dimensional representations $\mathbf{z}_i$. The standard VAE places a Gaussian prior $\mathbf{z} \sim p(\mathbf{z})$ on the latent variable, and learns a generative model $p_\theta(\mathbf{x}|\mathbf{z})$ that reconstructs $\mathbf{x}$ using $\mathbf{z}$, alongside an inference network $q_\phi(\mathbf{z}|\mathbf{x})$ that encodes $\mathbf{x}$ to $\mathbf{z}$. Both θ and ϕ are trained jointly by maximising the ELBO, a lower bound on marginal likelihood given by $\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} [\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{KL}\left[q_\phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})\right]$. This can be maximised using stochastic optimisers (Robbins & Monro, 1951; Kingma & Ba, 2014).

So far, we have assumed that the only data available are the observations $\mathbf{x}_1, \ldots, \mathbf{x}_n$, but in many practical applications we may have additional information such as a condition label for each observation. For example, in gene knockout studies, we have information about which gene was targeted for deletion in each cell; in multi-batch experiments we have information about which experimental batch each samples was collected in. Thus, we augment our data by considering data pairs $(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_n, c_n)$ where $\mathbf{x}$ is a high-dimensional observation, and c is a label indicating the condition or experimental batch that $\mathbf{x}$ was collected under.

Whilst VAEs are theoretically able to model the pairs $(\mathbf{x}_i, c_i)$, it makes sense to build a model that explicitly distinguishes between the $\mathbf{x}$ and c. The simplest model is the Conditional VAE (CVAE) (Sohn et al., 2015). In this model,

[1] Source code for the experiments is provided at https://github.com/BenevolentAI/CoMP.

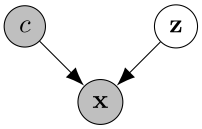

Figure 2. Structural Equation Model for observation $\mathbf{x}$ under known condition c with unobserved latent variable $\mathbf{z}$. In this model, $\mathbf{z}$ and c are independent in the prior.

a conditional generative model $p_\theta(\mathbf{x}|\mathbf{z}, c)$ and a conditional inference network $q_\phi(\mathbf{z}|\mathbf{x}, c)$ are trained using a modified ELBO. A key observation for our work is that the CVAE has many different ways to model the data. For example, it can completely ignore the condition c in p_θ and q_ϕ, reducing to the original VAE. Assuming that $\mathbf{x}$ is not independent of c, this failure mode of the CVAE would be apparent on a visualisation of the representations. For example, different values of c might be visible as separate latent clusters, as shown in Figure 1 [CVAE].

2.2. Counterfactual inference

If $(\mathbf{x}_i, c_i)$ represents an RNA transcript and the gene knock-out applied to the cell, a natural question to ask is "How would the transcript have differed if a different knock-out c' had been applied?" In general, *counterfactual inference* is necessary to answer questions of the form "How would the data have changed if c_i had been replaced by c'?" In this paper, we assume access to unpaired data, meaning that each cell $\mathbf{x}_i$ is observed only in one condition c_i. Answering counterfactual questions with such data is a notoriously difficult task, because they naturally refer to unobservable data (Pearl, 2009). A principled approach to such questions is to adopt the framework of Structural Equation Models (Bollen, 2005; Pearl, 2009). In this paper, we assume that the data generating process is given as in Figure 2. If this model is correct, counterfactual inference in the Pearl framework (Pearl, 2009) can then be performed by: 1) *abduction*: inferring the latent $\mathbf{z}$ from $\mathbf{x}$ and c using $p(\mathbf{z}|\mathbf{x}, c)$, 2) *action*: swap c for c', 3) *prediction*: use $p(\mathbf{x}|\mathbf{z}, c')$ to obtain a predictive distribution for the counterfactual. Thus, the counterfactual distribution of $\mathbf{x}_i$ observed with condition c_i but predicted for condition c' is given by

$$p\left(\mathbf{x}_{c=c'}|\mathbf{x}_i, c_i\right) = \int p(\mathbf{z}|\mathbf{x}_i, c_i)p(\mathbf{x}|\mathbf{z}, c')\, d\mathbf{z}. \qquad (1)$$

3. The constraint $\mathbf{z} \perp\!\!\!\perp c$

Our high-level approach is to learn a CVAE model with the constraint that $\mathbf{z} \perp\!\!\!\perp c$ under the encoder distribution q_ϕ. Visually, this means that latent representations from different conditions are aligned. To achieve this, we introduce a novel training penalty that penalises misalignment between different conditions (Section 4). We first discuss how this general approach applies to data integration, fair representation learning and counterfactual inference.

For data integration where c_i indicates experimental batch, spurious shifts may be present in the distribution of $\mathbf{x}$ due to differing experimental conditions, as opposed to true changes in the underlying biology. Using our approach, the latent $\mathbf{z}$ can be used in place of $\mathbf{x}$ for downstream tasks, thereby integrating data from different batches. Intuitively, by enforcing $\mathbf{z} \perp\!\!\!\perp c$ we 'subtract' batch effects, leaving a representation that has the same marginal distribution between batches. Alternatively, we can reconstruct the $\mathbf{x}_i$ as if they arose under a single experimental batch, performing batch correction in the original space. By enforcing $\mathbf{z} \perp\!\!\!\perp c$ in our encoder, we are assuming that there are no population-level differences between batches. This assumption bears a close resemblance to the assumptions used (sometimes implicitly) in latent factor (Leek & Storey, 2007; Stegle et al., 2012) and CVAE (Zuo & Chen, 2021; Lotfollahi et al., 2019a) models for data integration. We discuss our assumptions and ways to relax them in Section 7.

In fair representation learning, the notion of building a representation that cannot be used to recover c has been studied widely in recent literature (Zemel et al., 2013; Louizos et al., 2015; Kusner et al., 2017; Glastonbury et al., 2018). In particular, if we wish to make a predictive rule based on $\mathbf{x}$ that does not discriminate between individuals in different conditions c, we can use a fair representation $\mathbf{z}$, one which contains information from $\mathbf{x}$ but cannot be used to recover c, as an intermediate feature and train our model using $\mathbf{z}$. Being unable to recover c from $\mathbf{z}$ is equivalent to our constraint $\mathbf{z} \perp\!\!\!\perp c$ (see Appendix B).

For counterfactual inference, we can estimate equation (1) by replacing the true data generating distributions with model-based estimates $q_\phi(\mathbf{z}|\mathbf{x}_i, c_i)$ and $p_\theta(\mathbf{x}|\mathbf{z}, c')$, giving

$$\hat{p}(\mathbf{x}_{c=c'}|\mathbf{x}_i, c_i) = \int q_\phi(\mathbf{z}|\mathbf{x}_i, c_i)p_\theta(\mathbf{x}|\mathbf{z}, c')\, d\mathbf{z}. \qquad (2)$$

The failure mode in which different values of c form separate latent clusters, as in Figure 1 [CVAE], can be catastrophic for this application, because it violates assumptions of Figure 2. However, it is not true that the constraint $\mathbf{z} \perp\!\!\!\perp c$ alone is sufficient to guarantee the correct estimation of counterfactuals using (2). We discuss this is Section 5, and prove that, under additional assumptions, it becomes possible to identify counterfactuals using our CVAE approach with $\mathbf{z} \perp\!\!\!\perp c$. Counterfactual inference also provides a more rigorous foundation to discuss both fairness (Chiappa, 2019; Kusner et al., 2017; Zhang et al., 2016; Kilbertus et al., 2017; Zhang & Bareinboim, 2018) and data integration (Bareinboim & Pearl, 2016). As such, these three problems have a deep underlying connection.

4. Contrastive Mixture of Posteriors

Our approach to counterfactual inference, data integration and fair representation learning centres on learning a representation such that the latent variable $\mathbf{z}$ is independent of the condition c under the distribution[2] q, so that the latent clusters with different values of c are perfectly aligned. Building off the CVAE, which rarely achieves this in practice, a number of authors have attempted to use a penalty term to reduce the dependence of $\mathbf{z}$ upon c during training. The most successful methods, such as trVAE (Lotfollahi et al., 2019a), are based on MMD (Gretton et al., 2012). Whilst trVAE and related methods can work well, they require an MMD kernel, not a part of the original model, to be specified and its parameters to be carefully tuned. Experimentally, we observe that MMD-based methods can often struggle when there is complex global structure in the latent space. We also analyse the gradients of MMD penalties, showing that they have some undesirable properties.

We propose a novel method to enforce $\mathbf{z} \perp\!\!\!\perp c$ in a CVAE model. Our penalty is based on posterior distributions obtained from the model encoder itself. That is, we do not introduce any external discrepancy measure, rather we propose a penalty term that arises naturally from the model itself. Our penalty enforces the equality of the marginal distribution $q(\mathbf{z}|c)$ and $q(\mathbf{z}|\neg\, c)$ for each $c \in \mathcal{C}$, where $q(\mathbf{z}|c) = \mathbb{E}_{p(\mathbf{x}|c)}\left[q(\mathbf{z}|\mathbf{x}, c)\right]$ represents the marginal distribution of $\mathbf{z}$ over all points within condition c and

$$q(\mathbf{z}|\neg\, c) = \frac{\sum_{c' \in \mathcal{C}, c' \neq c} p(c') q(\mathbf{z}|c')}{\sum_{c' \in \mathcal{C}, c' \neq c} p(c')} \tag{3}$$

represents the marginal distribution of $\mathbf{z}$ over all conditions not equal to c. In Appendix B, we show that the statement '$q(\mathbf{z}|c) = q(\mathbf{z}|\neg\, c)$ for each c' is equivalent to the statement '$\mathbf{z} \perp\!\!\!\perp c$ under the distribution $p(\mathbf{x}, c)q(\mathbf{z}|\mathbf{x}, c)$'. Therefore, enforcing $\mathbf{z} \perp\!\!\!\perp c$ is the same as enforcing $q(\mathbf{z}|c)$ and $q(\mathbf{z}|\neg, c)$ to be equal for every c.

To encourage greater overlap between $q(\mathbf{z}|c)$ and $q(\mathbf{z}|\neg\, c)$, we can encourage points with the condition c to be in areas of high density under the representation distribution for *other* conditions, i.e. areas in which $q(\mathbf{z}|\neg\, c)$ is also high. To encourage this, we can add the penalty term $\mathcal{P}_0(\mathbf{z}_i, c_i) = -\log q(\mathbf{z}_i|\neg\, c_i)$ to the objective for the data pair $(\mathbf{x}_i, c_i)$. When we minimise $\mathcal{P}_0$, this brings the representations of samples under condition c_i towards regions of high density under $q(\mathbf{z}|\neg\, c_i)$.

Since the density $q(\mathbf{z}|\neg\, c)$ is not known in closed form, we approximate $q(\mathbf{z}|\neg\, c)$ using other points in the same training batch as $(\mathbf{x}_i, c_i)$. Indeed, suppose we have a batch $(\mathbf{x}_1, c_1), ..., (\mathbf{x}_B, c_B)$. We let I_c denote the subset of indices for which $c_j = c$ and $I_{\neg c}$ denote its complement. We use the approximation

$$\log q(\mathbf{z}_i|\neg\, c_i) \approx \log\left(\frac{1}{|I_{\neg c_i}|} \sum_{j \in I_{\neg c_i}} q(\mathbf{z}_i|\mathbf{x}_j, c_j)\right) \tag{4}$$

and we will show in Theorem 1 that this approximation in fact leads to a valid stochastic bound.

It may happen that the penalty $\mathcal{P}_0$ causes points to become too tightly clustered. Inspired by contrastive learning (van den Oord et al., 2018), we include a second term which promotes higher entropy of the marginal, thereby avoiding tight clusters of points. Combined with $\mathcal{P}_0$, this leads us to a second penalty $\mathcal{P}_1(\mathbf{z}_i, c_i) = \log q(\mathbf{z}_i|c_i) - \log q(\mathbf{z}_i|\neg\, c_i)$. Again, the density $q(\mathbf{z}|c)$ is not known in closed form, but we can approximate it using points within the same training batch in a similar fashion to (4). Combining both approximations to estimate $\mathcal{P}_1$ and then taking the mean of the penalty over the batch gives our *Contrastive Mixture of Posteriors (CoMP) misalignment penalty*

$$\begin{array}{c}\text{CoMP} \\ \text{penalty}\end{array} = \frac{1}{B}\sum_{i=1}^{B} \log\left(\frac{\frac{1}{|I_{c_i}|}\sum_{j \in I_{c_i}} q(\mathbf{z}_i|\mathbf{x}_j, c_i)}{\frac{1}{|I_{\neg c_i}|}\sum_{j \in I_{\neg c_i}} q(\mathbf{z}_i|\mathbf{x}_j, c_j)}\right) \tag{5}$$

where $\mathbf{x}_{1:B}, c_{1:B}, \mathbf{z}_{1:B} \sim \prod_{i=1}^{B} p(\mathbf{x}_i, c_i) q(\mathbf{z}_i|\mathbf{x}_i, c_i)$ is a random training batch of size B, I_c denotes the subset of $\{1, \ldots, B\}$ with condition c and $I_{\neg c} = \{1, \ldots, B\} \setminus I_c$. Our method therefore utilises a training penalty for CVAE-type models that encourages the constraint $\mathbf{z} \perp\!\!\!\perp c$ to hold by using mixtures of the variational posteriors themselves to approximate $q(\mathbf{z}|c)$ and $q(\mathbf{z}|\neg\, c)$.

As hinted at by the definition of $\mathcal{P}_1$, CoMP can be seen as approximating a symmetrised KL-divergence between the distributions $q(\mathbf{z}|c)$ and $q(\mathbf{z}|\neg\, c)$. In fact, the following theorem shows that the CoMP misalignment penalty is a *stochastic upper bound on a weighted sum of KL-divergences.*

Theorem 1. *The CoMP misalignment penalty satisfies*

$$\mathbb{E}\left[\frac{1}{B}\sum_{i=1}^{B} \log\left(\frac{\frac{1}{|I_{c_i}|}\sum_{j \in I_{c_i}} q(\mathbf{z}_i|\mathbf{x}_j, c_i)}{\frac{1}{|I_{\neg c_i}|}\sum_{j \in I_{\neg c_i}} q(\mathbf{z}_i|\mathbf{x}_j, c_j)}\right)\right] \geq \sum_{c \in \mathcal{C}} p(c)\, \mathrm{KL}\left[q(\mathbf{z}|c) || q(\mathbf{z}|\neg\, c)\right]$$

where the expectation is over $\prod_{i=1}^{B} p(\mathbf{x}_i, c_i) q(\mathbf{z}_i|\mathbf{x}_i, c_i)$. The bound becomes tight as $B \to \infty$.

The proof is given in Appendix C. Our result shows that our new penalty directly reduces the KL divergence between each pair $q(\mathbf{z}|c)$, $q(\mathbf{z}|\neg\, c)$ weighted by $p(c)$. As with standard contrastive learning, our method benefits from larger batch sizes. We add the CoMP misalignment penalty to

[2] We drop the θ, ϕ subscripts on p_θ and q_ϕ in this section for conciseness and legibility.

the familiar CVAE objective to give our *complete training objective* for a batch of size B as

$$\mathcal{L}_B^{\text{CoMP}}(\theta,\phi) = \frac{1}{B}\sum_{i=1}^{B}\Bigg[\log\left(\frac{p_\theta(\mathbf{x}_i|\mathbf{z}_i,c_i)p(\mathbf{z}_i)}{q_\phi(\mathbf{z}_i|\mathbf{x}_i,c_i)}\right) - \gamma\log\left(\frac{\frac{1}{|I_{c_i}|}\sum_{j\in I_{c_i}} q_\phi(\mathbf{z}_i|\mathbf{x}_j,c_i)}{\frac{1}{|I_{\neg c_i}|}\sum_{j\in I_{\neg c_i}} q_\phi(\mathbf{z}_i|\mathbf{x}_j,c_j)}\right)\Bigg] \tag{6}$$

with hyperparameter γ that controls the strength of the regularisation we apply to enforce the constraint $\mathbf{z} \perp\!\!\!\perp c$.

In Appendix D, we analyse the training gradient of the CoMP penalty, contrasting it with MMD. We show that, unlike MMD, CoMP gradients have a self-normalising property, allowing one to obtain strong gradients for distant points in a latent space with complex global structure.

5. Theory

5.1. Counterfactual identifiability in CVAE framework

When estimating the counterfactual predictions of equation (1), we replace the true data generating distributions with model-based estimates $q_\phi(\mathbf{z}|\mathbf{x}_i, c_i)$ and $p_\theta(\mathbf{x}|\mathbf{z}, c')$, giving equation (2). Unfortunately, it is possible for a model to fit the training data arbitrarily well, achieving large $p_\theta(\mathbf{x}|c)$, and yet give incorrect counterfactual predictions (see Pearl (2000), Bareinboim et al. (2020)).

In Proposition 5 in Appendix E, we show that this issue is present in the CVAE set-up when the true data generating distribution has $\mathbf{z} \sim N(0, I)$. In this example, the *non-identifiability* arises because we can apply a rotation in the latent space for condition $c = 1$, but not for $c = 0$, leading to different counterfactual predictions. At a more fundamental level, for a correctly specified model, symmetries of the true latent space distribution (the existence of a transformation R such that $R\mathbf{z} \overset{d}{=} \mathbf{z}$) make counterfactuals non-identifiable.

Empirically though, we find that the latent space distribution $q(\mathbf{z})$ is not, in fact, a Gaussian (Sec. 6) and has no apparent global symmetries. We also find in these experiments that counterfactual inference is stable between different training seeds, and that it accords extremely well with counterfactuals that are estimated using held-out cell type information. To explain this phenomenon theoretically, we prove that the non-Gaussianity of the true distribution for $\mathbf{z}$ leads, with additional assumptions, to counterfactual identifiability. Our assumptions include explicitly disallowing linear symmetries of the true latent distribution. See Appendix E for formal specification of our assumptions and the proof.

Theorem 2. *Suppose the true data generating distribution has $\mathbf{z} \sim r(\mathbf{z})$ and linear decoders for each condition. Assume $r(\mathbf{z})$ is non-Gaussian and that Assumption 7 holds. Then counterfactuals are identifiable from unpaired data.*

The theorem shows that when the true data generating distribution for $\mathbf{z}$ is non-Gaussian but the constraint $\mathbf{z} \perp\!\!\!\perp c$ still holds, then a model which best fits the training data also makes correct counterfactual predictions. In other words, estimating (1) by (2) is valid under these conditions.

5.2. Consistency of CoMP under prior misspecification

In the preceding section, we showed that the CVAE framework can identify counterfactuals when the true latent distribution $r(\mathbf{z})$ is non-Gaussian. However, our training objective still contains a Gaussian prior $p(\mathbf{z})$. Our empirical results (Fig. 4) indicate that this is not problematic, as CoMP is well able to learn non-Gaussian latent distributions. To ground this in theory, we show that training with the CoMP objective can recover the true model *provided that* the KL-divergence between the true prior and $N(0, I)$ is controlled.

Theorem 3. *Define $p_{r,\theta}(\mathbf{x}|c) = \mathbb{E}_{r(\mathbf{z})}[p_\theta(\mathbf{x}|\mathbf{z}, c)]$. There exists a constant K_1 such that, if $\mathrm{KL}[r(\mathbf{z})\|p(\mathbf{z})] \leq K_1$ and if the encoder network is sufficiently flexible, then maximising the CoMP objective with infinite data generated under the misspecified model with $\mathbf{z} \sim r(\mathbf{z})$ leads to a θ_∞ that is a maximum point of $\mathbb{E}_{\mathbf{x},c}[\log p_{r,\theta}(\mathbf{x}|c)]$.*

5.3. Evaluating theoretical assumptions in practice

Inspecting the latent distribution $q_\phi(\mathbf{z})$ provides insights on whether the conditions of the theorems hold in practice—if the latent distribution is different from $N(0, I)$, this hints that $r(\mathbf{z})$ has the correct form to make counterfactuals identifiable (Theorem 2) and that $\mathrm{KL}[r(\mathbf{z})\|p(\mathbf{z})]$ is sufficiently small for CoMP to find this non-Gaussian $\mathbf{z}$ through training (Theorem 3). More formally, Normality testing (Razali et al., 2011) could be employed to check non-Gaussianity. Secondly, upon retraining the model with different random seeds, the latent distribution $q_\phi(\mathbf{z})$ should remain the same up to a linear transformation, and counterfactual predictions should remain (approximately) the same.

6. Experiments

We perform experiments on four datasets: 1) Tumour / Cell Line: bulk gene expression profiles of tumours and cancer cell-lines across 39 different cancer types (Warren et al., 2021); 2) Stimulated / untreated single-cell PBMCs: single-cell gene expression (scRNA-seq) profiles of interferon (IFN)-β stimulated and untreated peripheral blood mononuclear cells (PBMCs) (Kang et al., 2018); 3) Single-cell RNA-seq data integration: scRNA-seq profiles of PBMCs that were processed using different library preparation protocols (Korsunsky et al., 2019); 4) UCI Adult Income: personal information of census participants and a binary high / low income label (Dua & Graff, 2017).

Table 1. Tumour / Cell Line experiment results, with $k = 100$, $c =$ Cell Line, and parameter $\alpha = 0.01$ for the kBET and m-kBET metrics. $s_{k,c}$ and $\tilde{s}_{k,c}$ are the two Silhouette Coefficient variants (see Section 6). The top scores are in **bold**.

	Accuracy	s	kBET	$\tilde{s}$	m-kBET
VAE	0.209	0.658	0.974	0.803	0.581
CVAE	0.328	0.554	0.931	0.684	0.571
VFAE	**0.585**	0.168	0.258	0.198	0.188
trVAE	**0.585**	0.096	0.163	0.138	0.123
Celligner	0.578	0.082	0.525	0.568	0.226
CoMP	0.579	**0.023**	**0.160**	**0.094**	**0.101**

The two broad objectives across our experiments are 1) to demonstrate the extent to which the two random variables $\mathbf{z}$ and c are independent, and 2) to quantify useful information retained in $\mathbf{z}$. To benchmark CoMP on the first objective, we use the following pair of k nearest-neighbour metrics: kBET$_{k,\alpha}$ (Büttner et al., 2019), the metric used to evaluate batch correction methods in biology, and a local Silhouette Coefficient (Rousseeuw, 1987) $s_{k,c}$. In both cases a low value close to zero indicates good local mixing of sample representations. As for the second objective, if we have access to a held-out discrete label d_i that represents information one wishes to preserve—in the Tumour / Cell Line case, d_i is the cancer type, while for the scRNA-seq experiments, it refers to cell type—then we calculate kBET and s separately for every fixed-d_i subpopulation and take the mean. We refer to these as the *mean Silhouette Coefficient* $\tilde{s}_{k,c}$ and the *mean kBET* metric m-kBET respectively. These two novel metrics are designed to penalise algorithms that achieve global alignment but mix up different cell types from different conditions, e.g. by aligning stimulated natural killer cells with unstimulated CD14 cells. Low values of both metrics indicate correct alignment of each subpopulation. Full details of the datasets and metrics, along with confidence intervals from multiple experiment runs, are presented in Appendix F.

6.1. Alignment of tumour and cell-line samples

Despite their widespread use in pre-clinical cancer studies, cancer cell-lines are known to have significantly different gene expression profiles compared to their corresponding tumour samples. Here we evaluate the ability of CoMP to subtract out the tumour / cell line condition. This caorm experiments on four datasets; 1) Tumour / Cell Line: bun be seen as both a dataset integration and batch effect correction task. In addition to the set of k nearest neighbour-based mixing evaluations, we train a Random Forest model on the representations of the tumour samples and their cancer-type labels and assess the prediction accuracy on held-out cell lines. To match the results from Warren et al. (2021), the evaluations are performed on the 2D UMAP projections. The results are presented in Table 1.

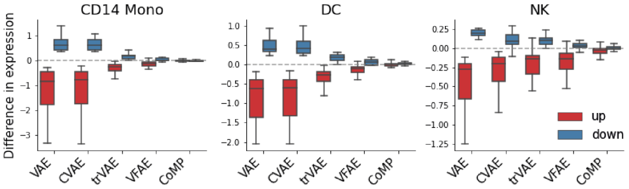

Figure 3. The difference in gene expression values for the top 50 differentially expressed genes (up-regulated: red, down-regulated: blue) between IFN-β stimulated cells and counterfactually stimulated control cells for CD14 monocytes, dendritic cells (DC) and natural killer (NK) cells. See Appendix F for further details.

As expected, both the VAE and CVAE baselines fail at the mixing task; the three explicitly penalised CVAE models and, to a lesser extent, the *Cellinger* method have good mixing performances, with CoMP outperforming the benchmark models by a significant margin on the silhouette coefficient and kBET metric, while successfully maintaining a high accuracy in the cancer-type prediction task. We also see from Figure 4A that CoMP representations have the fewest instances of isolated tumour-only clusters. Finally, from our evaluation on the $\tilde{s}$ and m-kBET metrics, we can deduce that the occurrence of cell lines of one cancer type erroneously clustering around tumours of a different type is less frequent for CoMP compared to the other models. In Appendix F we qualitatively validate this for several example clusters. Overall, we see that CoMP learns to correctly align matching cell type clusters under different conditions *without* any cell type labels being available during training.

6.2. Interventions

Obtaining molecular measurements from biological tissues typically requires destructive sampling, meaning that we are unable to study the gene expression profile of the same cell under multiple experimental conditions. Counterfactual inference (Sec. 2.2) can be used to predict how the molecular status of a destroyed biological sample would have differed if it were measured under different experimental conditions, such as applications of different drugs.

To assess CoMP's utility in counterfactual inference, we trained it on scRNA-seq data from PBMCs that were either stimulated with IFN-β or left untreated (control) (Kang et al., 2018). It is clear from Figure 4B that IFN-β stimulation causes clear shifts in the latent space of a standard VAE between stimulated and control cells from the same cell type. Noticeably, the CD14 and CD16 monocyte and dendritic cell (DC) populations see greater shifts in their gene expression after stimulation. The CVAE fails to align these particular cell types in the latent space, while trVAE, VFAE and CoMP perform better. However, stimulated and control cells are

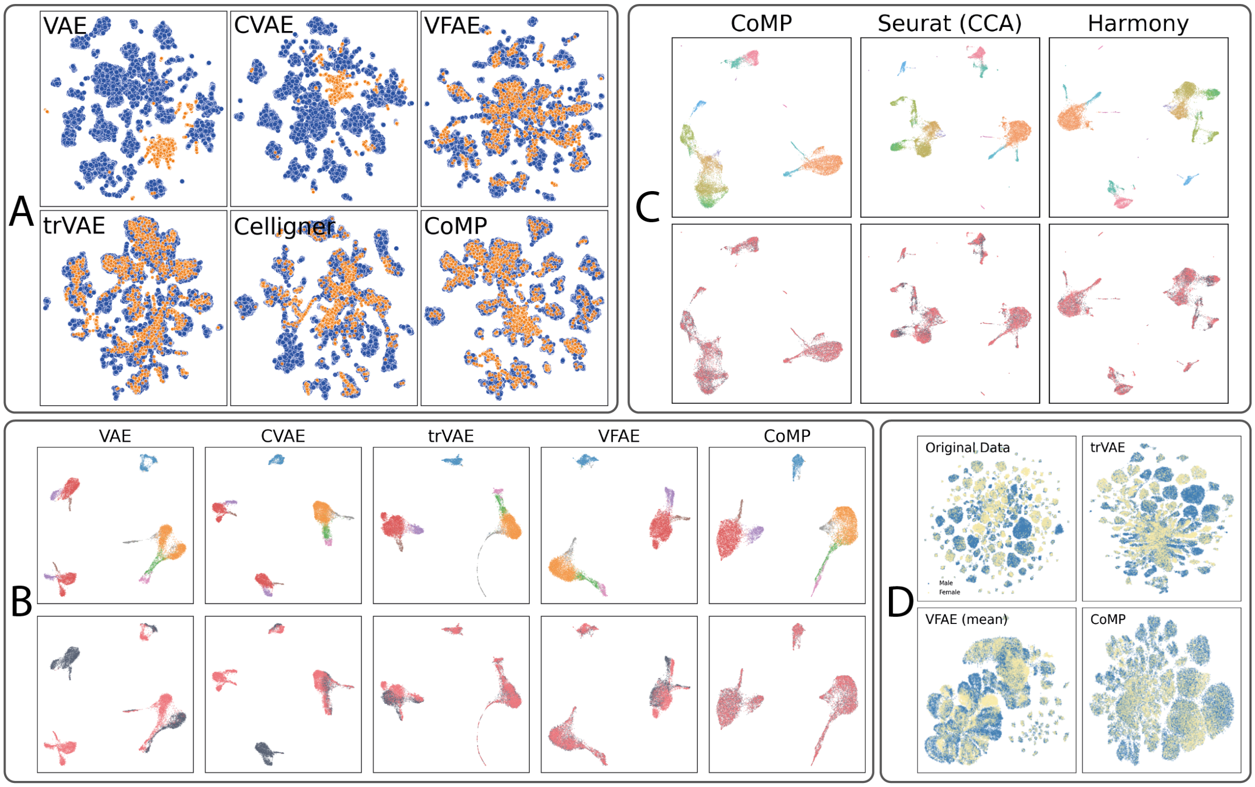

Figure 4. 2D UMAP projection of posterior means of $\mathbf{z}_i$. **A**: Tumour / Cell Line data. Tumours (blue) and cell lines (orange). **B**: Stimulated and control PBMC scRNA-seq data with colours highlighting immune cell types (top) and the IFN-β condition (bottom). **C**: PBMC scRNA-seq data processed using different protocols, with colours indicating immune cell type (top) and 5-prime or 3-prime V2 library construction protocol (bottom); **D**: UCI Adult Income dataset, coloured by gender with male (blue) and female (yellow).

best aligned in the latent space derived from CoMP (see metrics presented in Appendix F).

CoMP suppresses latent space shifts caused by perturbations, but captures perturbation information in the decoder network. This can then be used to predict gene expression levels of a cell type with and without the perturbation (counterfactual inference). To validate this empirically, we perform a counterfactual prediction task for a IFN-β control-to-stimulation variable swap, i.e. the gene expression profiles for control cells were encoded to the latent space, then reconstructed through the decoder with the condition $c \mapsto$ ‘stimulated’. This is a direct application of equation (2). We use held-out cell type labels to evaluate our predictions. Figure 3 shows how the profiles of (actual) stimulated cells differ from the counterfactual predictions for a selection of cell types (see Appendix F for the complete set of results). We see that baseline models tend to systematically underestimate the expression of genes up-regulated by stimulation and overestimate those down-regulated. CoMP outperforms all other models by accurately predicting the expression alterations brought about by stimulation.

Table 2. scRNA-seq data integration experiment results, with $k = 100$, $c =$ Protocol, and $\alpha = 0.05$ for kBET and m-kBET. $s_{k,c}$ and $\tilde{s}_{k,c}$ are the two Silhouette Coefficient variants (see Section 6). The top scores are in **bold**.

	s	kBET	$\tilde{s}$	m-kBET
Seurat CCA	0.0176	0.436	0.022	0.356
Harmony	0.0158	0.318	0.013	0.245
CoMP	**0.0004**	**0.164**	**0.0011**	**0.120**

6.3. Data integration of scRNA-seq data

The scale and complexity of single-cell ‘omics datasets has increased rapidly in recent years (Mereu et al., 2020; Luecken et al., 2021). Efforts such as the Human Cell Atlas (HCA) (Regev et al., 2017) require the collaboration of scientists from all around the world, each performing their own experiments and contributing their datasets to meet this goal. Processing these cells in different laboratories, with different protocols and technologies gives rise to distinct batch effects—unwanted technical variation observed in the data–that can obscure the biological variation that scientists seek to characterise and interpret. Being able to integrate diverse single-cell datasets, while accounting for

Table 3. UCI Adult Income experiment results with $k = 1000$, $c =$ Male for silhouette score s, and $k = 100$, $\alpha = 0.01$ for kBET. A lower gender prediction accuracy is better; 0.675 is the lowest achievable. VFAE-s is VFAE version taken directly from (Louizos et al., 2015) with sampled latents, and VAE-m is our implementation where we take the posterior means.

	Gender Acc	Income Acc	s	kBET
Original data	0.796	**0.849**	0.067	0.786
VAE	0.764	0.812	0.054	0.748
CVAE	0.778	0.819	0.054	0.724
VFAE-s	0.680	0.815	-	-
VFAE-m	0.789	0.805	0.046	0.571
trVAE	0.698	0.808	0.066	0.731
CoMP	**0.679**	0.805	**0.011**	**0.451**

unwanted technical variation, has been described as a major challenge in single-cell data science (Lähnemann et al., 2020; Eisenstein, 2020; Luecken et al., 2021).

To assess CoMP's performance in integrating single-cell data with batch effects, we trained it on scRNA-seq data of PBMCs processed with different library preparation protocols (Korsunsky et al., 2019). Here, we compare CoMP to two widely used single-cell integration approaches: Seurat (CCA) (Stuart et al., 2019) and Harmony (Korsunsky et al., 2019). As can be seen qualitatively in Figure 4C and quantitatively in Table 2 (with further metrics presented in Appendix F), CoMP provides improved mixing in the latent space for cells processed under different protocols when compared to Seurat and Harmony, while maintaining latent embeddings that represent distinct cell types.

6.4. Fair classification

The goal for this fair classification task is to learn a representation on the Adult Income dataset that is not predictive of an individual's gender whilst still being predictive of their income. We compute a baseline by predicting gender and income labels directly from the input data and compare our method to the published results for the VFAE (Louizos et al., 2015) and the trVAE. We also include results for a standard VAE and CVAE.

CoMP achieves a gender accuracy that is close to random (67.5%), tying with the VFAE results from (Louizos et al., 2015) whilst also remaining competitive with the other methods on income accuracy (Table 3). CoMP also outperforms all methods on the nearest neighbour and silhouette metrics. Latent space mixing between males and females can be seen qualitatively in the 2D UMAP projection (Figure 4D).

7. Discussion & Related Work

Data integration We have proposed CoMP as a new method for data integration and shown excellent performance on integrating bulk (Sec. 6.1) and single-cell (Sec. 6.3) RNA-seq data. CoMP uses the constraint $\mathbf{z} \perp\!\!\!\perp c$ to perform data integration, assuming that different batches should be aligned in latent space. Without additional prior knowledge, *some* assumption is needed to conduct data integration. Models such as PEER (Stegle et al., 2012) or SVA (Leek & Storey, 2007) estimate batch effects as latent factors. These models all assume that latent batch effects are independent of any biological variation as they are subsequently regressed out of the data. Most similar to CoMP are generative models such as scVI (Lopez et al., 2018), scMVAE (Zuo & Chen, 2021) and trVAE (Lotfollahi et al., 2019a), the latter two use a CVAE framework and trVAE directly encourages $\mathbf{z} \perp\!\!\!\perp c$ using an MMD penalty. Methods that incorporate linear batch correction, such as ComBat (Johnson et al., 2007), CCA (Correa et al., 2010) and Harmony (Korsunsky et al., 2019), instead assume that batch effects can be isolated to linear components of the data, which can then be removed. Other approaches (Haghverdi et al., 2018; Warren et al., 2021) use matching of mutual nearest neighbours to reduce statistical dependency between different conditions, improving alignment.

If there are phenotypical differences between batches, it is possible that CoMP 'over-corrects'. One approach to alleviate this is to focus on the weight γ in equation (6) that scales the CoMP penalty. When this is small, the model will focus more on providing accurate representations, and the force to perfectly align representations will be smaller. More rigorously, γ can be treated as a Lagrange multiplier, imposing a constraint $\sum_c p(c)\text{KL}\left[q(\mathbf{z}|c)\|q(\mathbf{z}|\neg c)\right] \leq L_\gamma$ for some constant L_γ. Thus, CoMP ensures that there is some level of alignment between different conditions, but may converge to a solution in which the constraint is not exactly zero. Empirically, our mean-kBET and mean-silhouette scores are designed to check that CoMP correctly aligns matching cell types across batches; we find it does so in our experiments. More generally, these scores could be helpful to diagnose over-correction.

Counterfactual inference CoMP proved to be extremely successful at predicting the response of cells to IFN-β stimulation. Predicting cell-level response to intervention has been previously tackled using VAEs by scGen (Lotfollahi et al., 2019b) and trVAE (Lotfollahi et al., 2019a). Although not always referred to as such, this problem is an example of counterfactual inference. Both scGen and trVAE can be interpreted as approaching counterfactual inference by enforcing alignment between different conditions in latent space with a linear translation and MMD penalty respectively. Johansson et al. (2016) estimated counterfactuals using representation learning, and used the discrepancy measure of Mansour et al. (2009) to encourage latent alignment between conditions.

We showed that latent alignment, $\mathbf{z} \perp\!\!\!\perp c$, is *not* always sufficient to perform valid counterfactual inference with a CVAE. Our identifiability and consistency results (Sec. 5) showed conditions under which counterfactual inference within the CVAE framework is valid. These results impact, not just CoMP, but other methods that use the CVAE framework for counterfactual inference, providing a blueprint to prove their counterfactual correctness. Theorems 2 and 3 do rest on assumptions; in Sec. 5.3 we discussed practical methods to try and check them. One particular limitation is that Theorem 2 assumes a linear decoder, whilst our experiments show that CoMP works well with non-linear decoders (shallow MLPs). Further work may look to generalise our results and weaken the assumptions that they make.

Fair representations CoMP proved effective at creating representations that cannot be used to infer the protected condition (Sec. 6.4). The VFAE (Louizos et al., 2015) adopts a closely related approach to ours, using a CVAE and an MMD penalty to enforce $\mathbf{z} \perp\!\!\!\perp c$. Understanding fairness through counterfactuals (Kusner et al., 2017) highlights the relevance of our theoretical analysis to fairness, showing that, under some conditions, CVAE approaches can go beyond alignment in latent space and directly target counterfactual notions of fairness.

Conclusion Marginal independence between the representation $\mathbf{z}$ and condition c is a mathematical thread linking data integration, counterfactual inference and fairness. We proposed CoMP, a novel method to enforce this independence requirement in practice. We saw that CoMP has several attractive properties. First, CoMP only uses the variational posteriors, requiring no additional discrepancy measures such as MMD. Second, the CoMP penalty is an upper bound on a weighted sum of KL divergences, making the connection with a well-known divergence measure. Third, we analysed CoMP theoretically as a means of performing counterfactual inference, showing identifiabilty and consistency under certain conditions. Empirically, we demonstrated CoMP's performance when applied to three biological and one fair representation learning dataset. These biological datasets are of critical importance in drug discovery, for example matching cell-lines to tumours for effective pre-clinical assay development of anti-cancer compounds. Overall, CoMP has the best in class performance on all tasks across a range of metrics and therefore broad utility across multiple challenging domains.

Acknowledgements

AF gratefully acknowledges funding from EPSRC grant no. EP/N509711/1. AF would like to thank Jake Fawkes for helpful comments on the counterfactual inference aspects of this paper. We would like to thank reviewers who reviewed this paper for NeurIPS 2021 and ICML 2022 and provided constructive feedback.

References

Aitkin, M. Posterior Bayes Factors. *Journal of the Royal Statistical Society. Series B (Methodological)*, 53(1):111–142, 1991. ISSN 0035-9246. URL https://www.jstor.org/stable/2345730.

Amodio, M., Dijk, D. V., Montgomery, R., Wolf, G., and Krishnaswamy, S. Out-of-sample extrapolation with neuron editing. *arXiv: Quantitative Methods*, 2018.

Bareinboim, E. and Pearl, J. Causal inference and the data-fusion problem. *Proceedings of the National Academy of Sciences*, 113(27):7345–7352, 2016.

Bareinboim, E., Correa, J. D., Ibeling, D., and Icard, T. On Pearl's hierarchy and the foundations of causal inference. *ACM Special Volume in Honor of Judea Pearl*, 2(3):4, 2020.

Bollen, K. A. *Structural equation models*. Wiley, 2005.

Büttner, M., Miao, Z., Wolf, F. A., Teichmann, S. A., and Theis, F. J. A test metric for assessing single-cell rna-seq batch correction. *Nature methods*, 16(1):43–49, 2019.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

Chiappa, S. Path-specific counterfactual fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 7801–7808, 2019.

Comon, P. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

Correa, N. M., Adali, T., Li, Y.-O., and Calhoun, V. D. Canonical correlation analysis for data fusion and group inferences. *IEEE signal processing magazine*, 27(4):39–50, 2010.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Eisenstein, M. Single-cell rna-seq analysis software providers scramble to offer solutions. *Nature biotechnology*, 38(3):254–257, 2020.

Fearnhead, P. and Prangle, D. Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(3):419–474, 2012. ISSN 1467-9868. doi: https://doi.org/10.1111/j.1467-9868.2011.01010.x.

URL https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2011.01010.x.

Foster, A., Jankowiak, M., O'Meara, M., Teh, Y. W., and Rainforth, T. A unified stochastic gradient approach to designing bayesian-optimal experiments. In *International Conference on Artificial Intelligence and Statistics*, pp. 2959–2969. PMLR, 2020.

Gerhard, D., Hunger, S., Lau, C., Maris, J., Meltzer, P., Meshinchi, S., Perlman, E., Zhang, J., Guidry-Auvil, J., and Smith, M. Therapeutically applicable research to generate effective treatments (target) project: Half of pediatric cancers have their own" driver" genes. In *PEDIATRIC BLOOD & CANCER*, volume 65, pp. S45–S45. WILEY 111 RIVER ST, HOBOKEN 07030-5774, NJ USA, 2018.

Ghandi, M., Huang, F., Jané-Valbuena, J., Kryukov, G., Lo, C., McDonald, E., Barretina, J., Gelfand, E., Bielski, C., Li, H., Hu, K., Andreev-Drakhlin, A. Y., Kim, J., Hess, J., Haas, B., Aguet, F., Weir, B., Rothberg, M., Paolella, B., Lawrence, M., Akbani, R., Lu, Y., Tiv, H. L., Gokhale, P., de Weck, A., Mansour, A. A., Oh, C., Shih, J., Hadi, K., Rosen, Y., Bistline, J., Venkatesan, K., Reddy, A., Sonkin, D., Liu, M., Lehár, J., Korn, J., Porter, D., Jones, M., Golji, J., Caponigro, G., Taylor, J. E., Dunning, C., Creech, A. L., Warren, A., McFarland, J. M., Zamanighomi, M., Kauffmann, A., Stransky, N., Imieliński, M., Maruvka, Y., Cherniack, A., Tsherniak, A., Vazquez, F., Jaffe, J., Lane, A. A., Weinstock, D., Johannessen, C., Morrissey, M. P., Stegmeier, F., Schlegel, R., Hahn, W., Getz, G., Mills, G., Boehm, J., Golub, T., Garraway, L., and Sellers, W. Next-generation characterization of the Cancer Cell Line Encyclopedia. *Nature*, 569:503–508, 2019.

Glastonbury, C. A., Ferlaino, M., Nellåker, C., and Lindgren, C. M. Adjusting for confounding in unsupervised latent representations of images. *arXiv preprint arXiv:1811.06498*, 2018.

Goldman, M. J., Craft, B., Hastie, M., Repečka, K., McDade, F., Kamath, A., Banerjee, A., Luo, Y., Rogers, D., Brooks, A. N., et al. Visualizing and interpreting cancer genomics data via the xena platform. *Nature biotechnology*, 38(6):675–678, 2020.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

Grønbech, C. H., Vording, M. F., Timshel, P. N., Sønderby, C. K., Pers, T. H., and Winther, O. scvae: Variational auto-encoders for single-cell gene expression data. *Bioinformatics*, 36(16):4415–4422, 2020.

Haghverdi, L., Lun, A., Morgan, M. D., and Marioni, J. Batch effects in single-cell rna-sequencing data are corrected by matching mutual nearest neighbors. *Nature Biotechnology*, 36:421–427, 2018.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. beta-vae: Learning basic visual concepts with a constrained variational framework. *International Conference on Learning Representations*, 2017.

Hyvarinen, A. Fast ica for noisy data using gaussian moments. In *1999 IEEE international symposium on circuits and systems (ISCAS)*, volume 5, pp. 57–61. IEEE, 1999.

Hyvarinen, A., Karhunen, J., and Oja, E. Independent component analysis and blind source separation, 2001.

Johansson, F., Shalit, U., and Sontag, D. Learning representations for counterfactual inference. In *International conference on machine learning*, pp. 3020–3029, 2016.

Johnson, W., Li, C., and Rabinovic, A. Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, 8 1:118–27, 2007.

Kang, H. M., Subramaniam, M., Targ, S., Nguyen, M., Maliskova, L., McCarthy, E., Wan, E., Wong, S., Byrnes, L., Lanata, C. M., et al. Multiplexed droplet single-cell rna-sequencing using natural genetic variation. *Nature biotechnology*, 36(1):89, 2018.

Khemakhem, I., Kingma, D., Monti, R., and Hyvarinen, A. Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, pp. 2207–2217. PMLR, 2020.

Kilbertus, N., Rojas Carulla, M., Parascandolo, G., Hardt, M., Janzing, D., and Schölkopf, B. Avoiding discrimination through causal reasoning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/f5f8590cd58a54e94377e6ae2eded4d9-Paper.pdf.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Korsunsky, I., Millard, N., Fan, J., Slowikowski, K., Zhang, F., Wei, K., Baglaenko, Y., Brenner, M., Loh, P.-r., and Raychaudhuri, S. Fast, sensitive and accurate integration of single-cell data with harmony. *Nature methods*, 16(12):1289–1296, 2019.

Kusner, M. J., Loftus, J. R., Russell, C., and Silva, R. Counterfactual fairness. *arXiv preprint arXiv:1703.06856*, 2017.

Lacoste, A., Luccioni, A., Schmidt, V., and Dandres, T. Quantifying the carbon emissions of machine learning. *arXiv preprint arXiv:1910.09700*, 2019.

Lähnemann, D., Köster, J., Szczurek, E., McCarthy, D. J., Hicks, S. C., Robinson, M. D., Vallejos, C. A., Campbell, K. R., Beerenwinkel, N., Mahfouz, A., et al. Eleven grand challenges in single-cell data science. *Genome biology*, 21(1):1–35, 2020.

Leek, J. and Storey, J. D. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genetics*, 3, 2007.

Li, K. and Malik, J. Implicit maximum likelihood estimation. *arXiv preprint arXiv:1809.09087*, 2018.

Lopez, R., Regier, J., Cole, M. B., Jordan, M. I., and Yosef, N. Deep generative modeling for single-cell transcriptomics. *Nature methods*, 15(12):1053–1058, 2018.

Lotfollahi, M., Naghipourfar, M., Theis, F. J., and Wolf, F. A. Conditional out-of-sample generation for unpaired data using trVAE. *arXiv preprint arXiv:1910.01791*, 2019a.

Lotfollahi, M., Wolf, F. A., and Theis, F. J. scGen predicts single-cell perturbation responses. *Nature methods*, 16 (8):715, 2019b.

Louizos, C., Swersky, K., Li, Y., Welling, M., and Zemel, R. The variational fair autoencoder. *arXiv preprint arXiv:1511.00830*, 2015.

Luecken, M. D., Büttner, M., Chaichoompu, K., Danese, A., Interlandi, M., Mueller, M. F., Strobl, D. C., Zappia, L., Dugas, M., Colomé-Tatché, M., et al. Benchmarking atlas-level data integration in single-cell genomics. *Nature methods*, pp. 1–10, 2021.

Mansour, Y., Mohri, M., and Rostamizadeh, A. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

Mathieu, E., Rainforth, T., Siddharth, N., and Teh, Y. W. Disentangling disentanglement in variational autoencoders. In *In International Conference on Machine Learning*, pp. 4402–4412. PMLR, 2019.

Mereu, E., Lafzi, A., Moutinho, C., Ziegenhain, C., McCarthy, D. J., Álvarez-Varela, A., Batlle, E., Grün, D., Lau, J. K., Boutet, S. C., et al. Benchmarking single-cell rna-sequencing protocols for cell atlas projects. *Nature Biotechnology*, 38(6):747–755, 2020.

Moneta, A., Entner, D., Hoyer, P. O., and Coad, A. Causal inference by independent component analysis: Theory and applications. *Oxford Bulletin of Economics and Statistics*, 75(5):705–730, 2013.

Pearl, J. *Causality: Models, reasoning and inference*. Cambridge University Press, 2000.

Pearl, J. *Causality*. Cambridge university press, 2009.

Razali, N. M., Wah, Y. B., et al. Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests. *Journal of statistical modeling and analytics*, 2(1):21–33, 2011.

Regev, A., Teichmann, S. A., Lander, E. S., Amit, I., Benoist, C., Birney, E., Bodenmiller, B., Campbell, P., Carninci, P., Clatworthy, M., et al. Science forum: the human cell atlas. *Elife*, 6:e27041, 2017.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014.

Robbins, H. and Monro, S. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Robert, C. P. and Rousseau, J. How Principled and Practical Are Penalised Complexity Priors? *Statistical Science*, 32 (1):36–40, February 2017. ISSN 0883-4237, 2168-8745.

Rolinek, M., Zietlow, D., and Martius, G. Variational autoencoders pursue pca directions (by accident). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12406–12415, 2019.

Rousseeuw, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

Shimizu, S., Hoyer, P. O., Hyvärinen, A., Kerminen, A., and Jordan, M. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(10), 2006.

Sohn, K., Lee, H., and Yan, X. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pp. 3483–3491, 2015.

Stegle, O., Parts, L., Piipari, M., Winn, J., and Durbin, R. Using probabilistic estimation of expression residuals (peer) to obtain increased power and interpretability of gene expression analyses. *Nature protocols*, 7(3):500–507, 2012.

Stuart, T., Butler, A., Hoffman, P., Hafemeister, C., Papalexi, E., Mauck III, W. M., Hao, Y., Stoeckius, M., Smibert, P., and Satija, R. Comprehensive integration of single-cell data. *Cell*, 177(7):1888–1902, 2019.

Svensson, V., Vento-Tormo, R., and Teichmann, S. A. Exponential scaling of single-cell rna-seq in the past decade. *Nature protocols*, 13(4):599–604, 2018.

Tomczak, J. and Welling, M. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*, pp. 1214–1223. PMLR, 2018.

van den Oord, A., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Vert, J.-P., Tsuda, K., and Schölkopf, B. A primer on kernel methods. *Kernel methods in computational biology*, 47: 35–70, 2004.

Wang, D. and Gu, J. Vasc: dimension reduction and visualization of single-cell rna-seq data by deep variational autoencoder. *Genomics, proteomics & bioinformatics*, 16 (5):320–331, 2018.

Warren, A., Chen, Y., Jones, A., Shibue, T., Hahn, W. C., Boehm, J. S., Vazquez, F., Tsherniak, A., and McFarland, J. M. Global computational alignment of tumor and cell line transcriptional profiles. *Nature Communications*, 12 (1):1–12, 2021.

Way, G. P. and Greene, C. S. Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders. In *PACIFIC SYMPOSIUM ON BIOCOMPUTING 2018: Proceedings of the Pacific Symposium*, pp. 80–91. World Scientific, 2018.

Weinstein, J., Collisson, E., Mills, G., Shaw, K., Ozenberger, B., Ellrott, K., Shmulevich, I., Sander, C., and Stuart, J. M. The Cancer Genome Atlas Pan-Cancer analysis project. *Nature Genetics*, 45:1113–1120, 2013.

Wolf, F. A., Angerer, P., and Theis, F. J. Scanpy: large-scale single-cell gene expression data analysis. *Genome biology*, 19(1):1–5, 2018.

Zemel, R., Wu, Y., Swersky, K., Pitassi, T., and Dwork, C. Learning fair representations. In *International conference on machine learning*, pp. 325–333. PMLR, 2013.

Zhang, J. and Bareinboim, E. Equality of opportunity in classification: A causal approach. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/ff1418e8cc993fe8abcfe3ce2003e5c5-Paper.pdf.

Zhang, L., Wu, Y., and Wu, X. A causal framework for discovering and removing direct and indirect discrimination. *arXiv preprint arXiv:1611.07509*, 2016.

Zuo, C. and Chen, L. Deep-joint-learning analysis model of single cell transcriptome and open chromatin accessibility data. *Briefings in Bioinformatics*, 22(4):bbaa287, 2021.

A. Additional background

A.1. Priors from posteriors

Given the long-standing debates around the role, selection and treatment of the prior within Bayesian statistics, it is natural that the choice of $p(\mathbf{z})$ in VAEs has come under scrutiny. The simplest alteration to dealing with the VAE prior is the β-VAE (Higgins et al., 2017), which scales the KL term of the ELBO by a hyperparameter β. While many of the traditional arguments concerning the prior revolve around principled points on objectivity, the primary issue for VAEs is the lack of expressiveness of the standard Normal distribution (Mathieu et al., 2019). The shared concern is that the prior is often selected for practical but, ultimately, spurious reasons of technical convenience (e.g. conjugacy, reparametrisation trick).

One solution is to simply replace the prior with the posterior. The apparent simplicity of this approach obscures the multiple issues that arise from double-dipping the data (Robert & Rousseau, 2017; Aitkin, 1991). Nevertheless the idea has endured: from the earlier proposal of posterior Bayes Factors as a solution to Lindley's paradox (Aitkin, 1991), modern Empirical Bayes methods, to likelihood-free models such as the calibration in approximate Bayesian computation models (Fearnhead & Prangle, 2012), invoking the posterior 'before its time' is increasingly performed to anchor statistical models to a more objective foundation.

For VAEs, a well-known proposal is to replace the prior with a mixture of variational posteriors, formed using pseudo-observations $\mathbf{u}_1, ..., \mathbf{u}_K$ (Tomczak & Welling, 2018). This Variational Mixture of Posteriors (VaMP) prior is given by

$$p^{\text{VaMP}}(\mathbf{z}) = \frac{1}{K}\sum_{k=1}^{K} q_\phi(\mathbf{z}|\mathbf{u}_k). \tag{7}$$

This results in a multi-modal prior, with the pseudo-observations learned by stochastic backpropagation along with the other parameters θ, ϕ. As we define in Section 4, the CoMP method adopts a similar non-parametric approach to defining a misalignment penalty.

B. Characterising the constraint $\mathbf{z} \perp\!\!\!\perp c$

To connect different notions of 'alignment in representation space' we recall the key components of the CVAE—the encoder $q_\phi(\mathbf{z}|\mathbf{x}, c)$ and decoder $p_\theta(\mathbf{x}|\mathbf{z}, c)$—and we now drop the θ, ϕ subscripts for conciseness. Recall that the marginal distribution of representations within condition $c \in \mathcal{C}$ is $q(\mathbf{z}|c) = \mathbb{E}_{p(\mathbf{x}|c)}\left[q(\mathbf{z}|\mathbf{x}, c)\right]$, and the marginal distribution of $\mathbf{z}$ over all conditions not equal to c is

$$q(\mathbf{z}|\neg\, c) = \frac{\sum_{c' \in \mathcal{C}, c' \neq c} p(c') q(\mathbf{z}|c')}{\sum_{c' \in \mathcal{C}, c' \neq c} p(c')} \tag{8}$$

in our notation. The following proposition brings together several key notions of alignment in the latent space.

Proposition 4. *The following are equivalent:*

1. $\mathbf{z} \perp\!\!\!\perp c$ *under distribution* q*,*
2. *for every* $c, c' \in \mathcal{C}$*,* $q(\mathbf{z}|c) = q(\mathbf{z}|c')$*,*
3. *for every* $c \in \mathcal{C}$*,* $q(\mathbf{z}|c) = q(\mathbf{z}|\neg\, c)$*,*
4. *the mutual information* $I(\mathbf{z}, c) = 0$ *under distribution* q*,*
5. $\mathbf{z}$ *cannot predict* c *better than random guessing.*

Proof. 1. $\implies$ 2. If $\mathbf{z} \perp\!\!\!\perp c$, then for every $c, c' \in \mathcal{C}$, $q(\mathbf{z}|c) = q(\mathbf{z}) = q(\mathbf{z}|c')$.

2. $\implies$ 3. For $c \in \mathcal{C}$, by the definition of $q(\mathbf{z}|\neg\, c)$ we have

$$q(\mathbf{z}|\neg\, c) = \frac{\sum_{c' \in \mathcal{C}, c' \neq c} p(c') q(\mathbf{z}|c')}{\sum_{c' \in \mathcal{C}, c' \neq c} p(c')} = \frac{\sum_{c' \in \mathcal{C}, c' \neq c} p(c') q(\mathbf{z}|c)}{\sum_{c' \in \mathcal{C}, c' \neq c} p(c')} = q(\mathbf{z}|c) \tag{9}$$

using condition 2.

3. $\implies$ 4. We have by definition of the mutual information under distribution q

$$I(\mathbf{z}, c) = \mathbb{E}_{p(\mathbf{x},c)q(\mathbf{z}|\mathbf{x},c)}\left[\log \frac{p(c)q(\mathbf{z}|c)}{p(c)q(\mathbf{z})}\right] \tag{10}$$

which can be written

$$= \mathbb{E}_{p(\mathbf{x},c)q(\mathbf{z}|\mathbf{x},c)}\left[\log \frac{p(c)q(\mathbf{z}|c)}{p(c)[p(c)q(\mathbf{z}|c) + (1-p(c))q(\mathbf{z}|\neg c)]}\right] \tag{11}$$

applying condition 3. gives

$$= \mathbb{E}_{p(\mathbf{x},c)q(\mathbf{z}|\mathbf{x},c)}\left[\log \frac{p(c)q(\mathbf{z}|c)}{p(c)[p(c)q(\mathbf{z}|c) + (1-p(c))q(\mathbf{z}|c)]}\right] \tag{12}$$

$$= \mathbb{E}_{p(\mathbf{x},c)q(\mathbf{z}|\mathbf{x},c)}\left[\log \frac{p(c)q(\mathbf{z}|c)}{p(c)q(\mathbf{z}|c)}\right] \tag{13}$$

$$= 0. \tag{14}$$

4. $\implies$ 5.[3] Let $Q(c|\mathbf{z})$ be some prediction rule for predicting c using $\mathbf{z}$. By Gibbs' Inequality, we have

$$I(\mathbf{z}, c) \geq \mathbb{E}_{p(\mathbf{x},c)q(\mathbf{z}|\mathbf{x},c)}\left[\log \frac{Q(c|\mathbf{z})}{p(c)}\right]. \tag{15}$$

Since $I(\mathbf{z}, c) = 0$, we have

$$\mathbb{E}_{p(\mathbf{x},c)q(\mathbf{z}|\mathbf{x},c)}\left[\log Q(c|\mathbf{z})\right] \leq \mathbb{E}_{p(\mathbf{x},c)}\left[\log p(c)\right]. \tag{16}$$

Observe that the left hand side above is the expected log-likelihood for the prediction rule Q, whilst the right hand side is the the log-likelihood for random guessing of c using only its marginal distribution $p(c)$. We see that random guessing obtains a log-likelihood which is at least as good as that obtained using the rule Q.

5. $\implies$ 1. Consider the prediction rule

$$Q^*(c|\mathbf{z}) := \frac{p(c)q(\mathbf{z}|c)}{q(\mathbf{z})}. \tag{17}$$

By condition 5., we have

$$\mathbb{E}_{p(\mathbf{x},c)q(\mathbf{z}|\mathbf{x},c)}\left[\log Q^*(c|\mathbf{z})\right] \leq \mathbb{E}_{p(\mathbf{x},c)}\left[\log p(c)\right]. \tag{18}$$

Hence,

$$\mathbb{E}_{p(\mathbf{x},c)q(\mathbf{z}|\mathbf{x},c)}\left[\log \frac{p(c)q(\mathbf{z}|c)}{p(c)q(\mathbf{z})}\right] \leq 0. \tag{19}$$

By Gibbs' Inequality,

$$\mathbb{E}_{p(\mathbf{x},c)q(\mathbf{z}|\mathbf{x},c)}\left[\log \frac{p(c)q(\mathbf{z}|c)}{p(c)q(\mathbf{z})}\right] \geq 0 \tag{20}$$

with equality if and only if $p(c)q(\mathbf{z}|c) = p(c)q(\mathbf{z})$. By (19), equality does hold, so $p(c)q(\mathbf{z}|c) = p(c)q(\mathbf{z})$ meaning $\mathbf{z} \perp\!\!\!\perp c$ under distribution q. □

C. CoMP misalignment penalty

We restate and prove Theorem 1.

Theorem 1. *The CoMP misalignment penalty satisfies*

$$\mathbb{E}\left[\frac{1}{B}\sum_{i=1}^{B} \log\left(\frac{\frac{1}{|I_{c_i}|}\sum_{j\in I_{c_i}} q(\mathbf{z}_i|\mathbf{x}_j, c_i)}{\frac{1}{|I_{\neg c_i}|}\sum_{j\in I_{\neg c_i}} q(\mathbf{z}_i|\mathbf{x}_j, c_j)}\right)\right]$$
$$\geq \sum_{c\in\mathcal{C}} p(c)\,\mathrm{KL}\left[q(\mathbf{z}|c)||q(\mathbf{z}|\neg\, c)\right]$$

where the expectation is over $\prod_{i=1}^{B} p(\mathbf{x}_i, c_i)q(\mathbf{z}_i|\mathbf{x}_i, c_i)$. The bound becomes tight as $B \to \infty$.

[3] We interpret 'better prediction' in condition 5. as achieving a higher expected log-likelihood.

Proof. First, by linearity of the expectation we have

$$\begin{aligned}&\mathbb{E}_{\prod_{i=1}^{B} p(\mathbf{x}_i,c_i)q(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\frac{1}{B}\sum_{i=1}^{B}\log\left(\frac{1}{|I_{c_i}|}\sum_{j\in I_{c_i}} q(\mathbf{z}_i|\mathbf{x}_j,c_i)\right)-\log\left(\frac{1}{|I_{\neg c_i}|}\sum_{j\in I_{\neg c_i}} q(\mathbf{z}_i|\mathbf{x}_j,c_j)\right)\right]\\ &=\mathbb{E}_{\prod_{i=1}^{B} p(\mathbf{x}_i,c_i)q(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log\left(\frac{1}{|I_{c_1}|}\sum_{j\in I_{c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_i)\right)-\log\left(\frac{1}{|I_{\neg c_1}|}\sum_{j\in I_{\neg c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_j)\right)\right].\end{aligned}\tag{21}$$

Focusing on the latter term, Jensen's Inequality gives

$$\mathbb{E}_{\prod_{i=1}^{B} p(\mathbf{x}_i,c_i)q(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[-\log\left(\frac{1}{|I_{\neg c_1}|}\sum_{j\in I_{\neg c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_j)\right)\right]\tag{22}$$

$$\geq\mathbb{E}_{p(\mathbf{x}_1,c_1)q(\mathbf{z}_1|\mathbf{x}_1,c_1)}\left[-\log\left(\mathbb{E}_{\prod_{i>1}^{B} p(\mathbf{x}_i,c_i)q(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\frac{1}{|I_{\neg c_1}|}\sum_{j\in I_{\neg c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_j)\right]\right)\right]\tag{23}$$

$$=\mathbb{E}_{p(\mathbf{x}_1,c_1)q(\mathbf{z}_1|\mathbf{x}_1,c_1)}\left[-\log q(\mathbf{z}_1|\neg\, c_1)\right].\tag{24}$$

For the other term, we take our inspiration from recent work on experimental design (Foster et al., 2020). We have

$$\mathbb{E}_{\prod_{i=1}^{B} p(\mathbf{x}_i,c_i)q(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log\left(\frac{1}{|I_{c_1}|}\sum_{j\in I_{c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_i)\right)\right]\tag{25}$$

$$=\mathbb{E}_{\prod_{i=1}^{B} p(\mathbf{x}_i,c_i)q(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log q(\mathbf{z}_1|c_1)+\log\left(\frac{\frac{1}{|I_{c_1}|}\sum_{j\in I_{c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_i)}{q(\mathbf{z}_1|c_1)}\right)\right]\tag{26}$$

$$=\mathbb{E}_{\prod_{i=1}^{B} p(\mathbf{x}_i,c_i)q(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log q(\mathbf{z}_1|c_1)\right]+\Delta\tag{27}$$

Then applying the tower rule with variable $c_1, |I_{c_1}|$ we have the difference term equal to

$$\Delta=\mathbb{E}_{c_1,|I_{c_1}|}\left[\mathbb{E}_{\prod_{i=1}^{|I_{c_1}|} p(\mathbf{x}_i|c_1)q(\mathbf{z}_1|\mathbf{x}_1,c_1)}\left[\log\left(\frac{\frac{1}{|I_{c_1}|}\sum_{j\in I_{c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_i)}{q(\mathbf{z}_1|c_1)}\right)\right]\right]\tag{28}$$

$$=\mathbb{E}_{c_1,|I_{c_1}|}\left[\mathbb{E}_{\prod_{i=1}^{|I_{c_1}|} p(\mathbf{x}_i|c_1)q(\mathbf{z}_1|\mathbf{x}_1,c_1)}\left[\log\left(\frac{\prod_{i=1}^{|I_{c_1}|} p(\mathbf{x}_i|c_1)\frac{1}{|I_{c_1}|}\sum_{j\in I_{c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_i)}{\prod_{i=1}^{|I_{c_1}|} p(\mathbf{x}_i|c_1)q(\mathbf{z}_1|c_1)}\right)\right]\right]\tag{29}$$

Now observe that $\mathbf{z}_1,...,\mathbf{z}_{|I_{c_1}|}$ are equal in distribution, so we can change the sampling distribution to be over

$$\prod_{i=1}^{|I_{c_1}|} p(\mathbf{x}_i|c_1)\frac{1}{|I_{c_1}|}\sum_{j\in I_{c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_i)\tag{30}$$

which amounts to choosing at random which of the $\mathbf{x}_1,...,\mathbf{x}_{|I_{c_1}|}$ to sample $\mathbf{z}_1$ from. Finally, we observe that

$$\prod_{i=1}^{|I_{c_1}|} p(\mathbf{x}_i|c_1)q(\mathbf{z}_1|c_1)\tag{31}$$

is a normalised distribution over $\mathbf{x}_1,...,\mathbf{x}_{|I_{c_1}|},\mathbf{z}_1$. Thus we can write Δ as the following expected KL divergence

$$\Delta=\mathbb{E}_{c_1,|I_{c_1}|}\left[\mathrm{KL}\left[\prod_{i=1}^{|I_{c_1}|} p(\mathbf{x}_i|c_1)\frac{1}{|I_{c_1}|}\sum_{j\in I_{c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_i)\middle\|\prod_{i=1}^{|I_{c_1}|} p(\mathbf{x}_i|c_1)q(\mathbf{z}_1|c_1)\right]\right].\tag{32}$$

Since the KL divergence is non-negative, we have shown that $\Delta \geq 0$. Therefore

$$\mathbb{E}_{\prod_{i=1}^{B} p(\mathbf{x}_i,c_i)q(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log\left(\frac{1}{|I_{c_1}|}\sum_{j\in I_{c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_i)\right)\right] \geq \mathbb{E}_{p(\mathbf{x}_1,c_1)q(\mathbf{z}_1|\mathbf{x}_1,c_1)}\left[\log q(\mathbf{z}_1|c_1)\right]. \tag{33}$$

Putting these two results together, we have shown that

$$\mathbb{E}_{\prod_{i=1}^{B} p(\mathbf{x}_i,c_i)q(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\frac{1}{B}\sum_{i=1}^{B}\log\left(\frac{1}{|I_{c_i}|}\sum_{j\in I_{c_i}} q(\mathbf{z}_i|\mathbf{x}_j,c_i)\right) - \log\left(\frac{1}{|I_{\neg c_i}|}\sum_{j\in I_{\neg c_i}} q(\mathbf{z}_i|\mathbf{x}_j,c_j)\right)\right] \tag{34}$$

$$\geq \mathbb{E}_{\prod_{i=1}^{B} p(\mathbf{x}_i,c_i)q(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log\frac{q(\mathbf{z}_1|c_1)}{q(\mathbf{z}_1|\neg\, c_1)}\right] \tag{35}$$

$$= \mathbb{E}_{p(\mathbf{x}_1,c_1)q(\mathbf{z}_1|\mathbf{x}_1,c_1)}\left[\log\frac{q(\mathbf{z}_1|c_1)}{q(\mathbf{z}_1|\neg\, c_1)}\right] \tag{36}$$

$$= \sum_{c\in\mathcal{C}} p(c_1)\mathbb{E}_{p(\mathbf{x}_1|c_1)q(\mathbf{z}_1|\mathbf{x}_1,c_1)}\left[\log\frac{q(\mathbf{z}_1|c_1)}{q(\mathbf{z}_1|\neg\, c_1)}\right] \tag{37}$$

$$= \sum_{c\in\mathcal{C}} p(c_1)\mathbb{E}_{q(\mathbf{z}_1|c_1)}\left[\log\frac{q(\mathbf{z}_1|c_1)}{q(\mathbf{z}_1|\neg\, c_1)}\right] \tag{38}$$

$$= \sum_{c\in\mathcal{C}} p(c)\,\mathrm{KL}[q(\mathbf{z}|c)\|q(\mathbf{z}|\neg\, c)]. \tag{39}$$

Finally, as $B \to \infty$, the Strong Law of Large Numbers implies that

$$\log\left(\frac{1}{|I_{c_1}|}\sum_{j\in I_{c_1}} q(\mathbf{z}_1|\mathbf{x}_j,c_i)\right) \to q(\mathbf{z}_i|c_i) \text{ a.s.}, \tag{40}$$

$$\log\left(\frac{1}{|I_{\neg c_i}|}\sum_{j\in I_{\neg c_i}} q(\mathbf{z}_i|\mathbf{x}_j,c_j)\right) \to q(\mathbf{z}_i|\neg\, c_i) \text{ a.s.}, \tag{41}$$

so (under mild technical assumptions) we conclude that the bound becomes tight in this limit. This completes the proof. □

C.1. Combining with the β-VAE

Note that the CoMP penalty may also be combined with the β-VAE objective (Higgins et al., 2017), giving the overall objective for a training batch

$$\mathcal{L}_{B,\beta}^{\text{CoMP}} = \frac{1}{B}\sum_{i=1}^{B}\left[\log p_\theta(\mathbf{x}_i|\mathbf{z}_i,c_i) + \beta\log\frac{p(\mathbf{z}_i)}{q_\phi(\mathbf{z}_i|\mathbf{x}_i,c_i)} - \gamma\log\left(\frac{\frac{1}{|I_{c_i}|}\sum_{j\in I_{c_i}} q_\phi(\mathbf{z}_i|\mathbf{x}_j,c_i)}{\frac{1}{|I_{\neg c_i}|}\sum_{j\in I_{\neg c_i}} q_\phi(\mathbf{z}_i|\mathbf{x}_j,c_j)}\right)\right]. \tag{42}$$

D. Analysing CoMP gradients

We attempt to understand how the CoMP penalty differs from existing penalties in the literature. Specifically, we compare CoMP using a Gaussian posterior family with MMD using a Radial Basis Kernel (Vert et al., 2004). To analyse MMD and CoMP gradients, we focus on the two specific cases that highlight the similarities between these methods, revealing the remaining differences. Specifically, we consider MMD with a simple unnormalised Radial Basis Kernel (Vert et al., 2004)

$$k(\mathbf{z},\mathbf{z}') = e^{-\|\mathbf{z}-\mathbf{z}'\|^2}, \tag{43}$$

and a Gaussian variational posterior family with fixed covariance matrix $\frac{1}{2}I$

$$q(\mathbf{z}|\mathbf{x},c) \propto e^{-\|\mathbf{z}-\boldsymbol{\mu}_{\mathbf{z}}(\mathbf{x},c)\|^2}. \tag{44}$$

We also assume just two conditions $|\mathcal{C}| = 2$. We show that both methods can be interpreted as applying a penalty to each element $\mathbf{z}_i, c_i$ of the training batch. Full derivations are given in the following section. For the MMD penalty for $\mathbf{z}_i, c_i$, the gradient takes the form

$$\nabla_{\mathbf{z}_i} \mathcal{P}_{\text{MMD}}(\mathbf{z}_i, c_i) = \frac{2}{|I_{c_i}|^2} \sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \mathbf{z}_j\|^2} (\mathbf{z}_j - \mathbf{z}_i) - \frac{4}{|I_{\neg c_i}|\,|I_{c_i}|} \sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \mathbf{z}_j\|^2} (\mathbf{z}_j - \mathbf{z}_i), \tag{45}$$

whilst the CoMP penalty gradient takes the form

$$\nabla_{\mathbf{z}_i} \mathcal{P}_{\text{CoMP}}(\mathbf{z}_i, c_i) = \frac{2 \sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2} (\boldsymbol{\mu}_{\mathbf{z}_j} - \mathbf{z}_i)}{B \sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2}} - \frac{2 \sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2} (\boldsymbol{\mu}_{\mathbf{z}_j} - \mathbf{z}_i)}{B \sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2}} \tag{46}$$

where $\boldsymbol{\mu}_{\mathbf{z}_j}$ is the variational mean for $\mathbf{z}_j$. One important feature of the MMD gradients is that, if $\|\mathbf{z}_i - \mathbf{z}_j\|^2$ is large for all $j \neq i$, for instance when the point $\mathbf{z}_i$ is part of an isolated cluster, then the gradient to update the representation $\mathbf{z}_i$ will be small. So if $\mathbf{z}_i$ is already very isolated from the distribution $q(\mathbf{z}|\neg\, c_i)$, then the gradients bringing it closer to points with condition $\neg\, c_i$ will be small. In comparison to the MMD gradient, it can be seen that gradients for CoMP are *self-normalised*. This means that the gradient through $\mathbf{z}_i$ will be large, even when $\mathbf{z}_i$ is very far away from any points with condition $\neg\, c_i$. This, in turn, suggests that that CoMP is likely to be preferable to MMD when we have a number of isolated clusters or interesting global structure in latent space, something which often occurs with biological data. The CoMP approach also bears a resemblance to nearest-neighbour approaches (Li & Malik, 2018). Indeed, for a Gaussian posterior as $\sigma \to 0$, the $\neg\, c_i$ term of the gradient places all its weight on the nearest element of the batch under condition $\neg\, c_i$.

D.1. Derivations

For an MMD penalty, the simplest form of the Kernel Two-sample Test statistic (Gretton et al., 2012) with batch size B can be written as follows

$$\mathcal{P}_{\text{MMD}} = \sum_{i=1}^{B} \mathcal{P}_{\text{MMD}}(\mathbf{z}_i, c_i) \tag{47}$$

$$= \sum_{i=1}^{B} \left(\frac{1}{|I_{c_i}|^2} \sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \mathbf{z}_j\|^2} - \frac{1}{|I_{\neg c_i}|\,|I_{c_i}|} \sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \mathbf{z}_j\|^2} \right), \tag{48}$$

taking gradients with respect to $\mathbf{z}_i$ gives us

$$\nabla_{\mathbf{z}_i} \mathcal{P}_{\text{MMD}}(\mathbf{z}_i, c_i) = \frac{2}{|I_{c_i}|^2} \sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \mathbf{z}_j\|^2} (\mathbf{z}_j - \mathbf{z}_i) - \frac{2}{|I_{\neg c_i}|\,|I_{c_i}|} \sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \mathbf{z}_j\|^2} (\mathbf{z}_j - \mathbf{z}_i), \tag{49}$$

the gradients of the total penalty are

$$\nabla_{\mathbf{z}_i} \mathcal{P}_{\text{MMD}} = \frac{4}{|I_{c_i}|^2} \sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \mathbf{z}_j\|^2} (\mathbf{z}_j - \mathbf{z}_i) - \frac{4}{|I_{\neg c_i}|\,|I_{c_i}|} \sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \mathbf{z}_j\|^2} (\mathbf{z}_j - \mathbf{z}_i). \tag{50}$$

The CoMP penalty (ignoring normalising constants) is

$$\mathcal{P}_{\text{CoMP}} = \sum_{i=1}^{B} \mathcal{P}_{\text{CoMP}}(\mathbf{z}_i, c_i) \tag{51}$$

$$= \frac{1}{B} \sum_{i=1}^{B} \log \left(\frac{1}{|I_{c_i}|} \sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2} \right) - \log \left(\frac{1}{|I_{\neg c_i}|} \sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2} \right), \tag{52}$$

if we take the gradient with respect to $\mathbf{z}_i$ we obtain

$$\nabla_{\mathbf{z}_i} \mathcal{P}_{\text{CoMP}}(\mathbf{z}_i, c_i) = \frac{2 \sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2} (\boldsymbol{\mu}_{\mathbf{z}_j} - \mathbf{z}_i)}{B \sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2}} - \frac{2 \sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2} (\boldsymbol{\mu}_{\mathbf{z}_j} - \mathbf{z}_i)}{B \sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2}} \tag{53}$$

where $\boldsymbol{\mu}_{\mathbf{z}_j} = \boldsymbol{\mu}(\mathbf{x}_i, c_i)$ is the variational mean for $\mathbf{z}_j$. The gradient of the full penalty with respect to $\boldsymbol{\mu}_{\mathbf{z}_i}$, noting $\mathbf{z}_i = \boldsymbol{\mu}_{\mathbf{z}_i} + \boldsymbol{\epsilon}_i$, is

$$\begin{aligned}\nabla_{\boldsymbol{\mu}_{\mathbf{z}_i}} \mathcal{P}_{\text{CoMP}} =& \frac{2\sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2} (\boldsymbol{\mu}_{\mathbf{z}_j} - \mathbf{z}_i)}{B \sum_{j \in I_{c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2}} - \frac{2\sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2} (\boldsymbol{\mu}_{\mathbf{z}_j} - \mathbf{z}_i)}{B \sum_{j \in I_{\neg c_i}} e^{-\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2}} \\ &+ 2 \sum_{j \in I_{c_i}} \frac{e^{-\|\mathbf{z}_j - \boldsymbol{\mu}_{\mathbf{z}_i}\|^2} (\mathbf{z}_j - \boldsymbol{\mu}_{\mathbf{z}_i})}{B \sum_{k \in I_{c_i}} e^{-\|\mathbf{z}_j - \boldsymbol{\mu}_{\mathbf{z}_k}\|^2}} - 2 \sum_{j \in I_{\neg c_i}} \frac{e^{-\|\mathbf{z}_j - \boldsymbol{\mu}_{\mathbf{z}_i}\|^2} (\mathbf{z}_j - \boldsymbol{\mu}_{\mathbf{z}_i})}{B \sum_{k \in I_{\neg c_j}} e^{-\|\mathbf{z}_j - \boldsymbol{\mu}_{\mathbf{z}_k}\|^2}}.\end{aligned} \tag{54}$$

Finally, to see the connection with nearest neighbour methods, we repeat this analysis with Gaussian posterior with fixed variance σ^2. The gradient term is then

$$\begin{aligned}\nabla_{\mathbf{z}_i} \mathcal{P}_{\text{CoMP}}(\mathbf{z}_i, c_i) = & \frac{\frac{1}{\sigma^2} \sum_{j \in I_{c_i}} e^{-\frac{1}{2\sigma^2}\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2} (\boldsymbol{\mu}_{\mathbf{z}_j} - \mathbf{z}_i)}{B \sum_{j \in I_{c_i}} e^{-\frac{1}{2\sigma^2}\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2}} \\ & - \frac{\frac{1}{\sigma^2} \sum_{j \in I_{\neg c_i}} e^{-\frac{1}{2\sigma^2}\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2} (\boldsymbol{\mu}_{\mathbf{z}_j} - \mathbf{z}_i)}{B \sum_{j \in I_{\neg c_i}} e^{-\frac{1}{2\sigma^2}\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2}}.\end{aligned} \tag{55}$$

As $\sigma \to 0$, we have

$$\frac{e^{-\frac{1}{2\sigma^2}\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_k}\|^2}}{\sum_{j \in I_{\neg c_i}} e^{-\frac{1}{2\sigma^2}\|\mathbf{z}_i - \boldsymbol{\mu}_{\mathbf{z}_j}\|^2}} \to \delta_{k\text{nn}_i} \tag{56}$$

where nn_i is the index of the nearest neighbour to $\mathbf{z}_i$ among the set $\{\mathbf{z}_j : j \in I_{\neg c_i}\}$, i.e.

$$\text{nn}_i = \text{argmin}_{j \in I_{\neg c_i}} \|\mathbf{z}_i - \mathbf{z}_j\| \tag{57}$$

indicating that the gradient between $\mathbf{z}_i$ and $\mathbf{z}_{\text{nn}_i}$ becomes the dominant term in the limit.

E. Theory

We have introduced CoMP, a new approach to enforcing the constraint $\mathbf{z} \perp\!\!\!\perp c$ in a CVAE model. However, it remains to be shown that training a CVAE with CoMP on unpaired data, i.e. on independent samples $(\mathbf{x}_i, c_i)_{i=1}^{\infty}$ from the data generating process, leads to valid counterfactual predictions.

In the language of the Causal Hierarchy Theorem (Pearl, 2000; Bareinboim et al., 2020), our unpaired data may be treated as interventional data, at level 2 of the hierarchy. (This is accurate in biological applications in which data under non-control conditions may be produced by actively intervening on samples with a drug, gene knock-out, etc. It is also formally correct in Fig. 2 in which c has no parents.) Counterfactuals exist on level 3 of the hierarchy, and therefore additional assumptions or constraints are necessary to infer them from our data. To illustrate the problem of *counterfactual identifiability*, we begin with a negative result for counterfactual inference with the CVAE framework.

Proposition 5. *Consider the following generative CVAE models for data* $\mathbf{x}, c$ *with two conditions* $c \in \{0, 1\}$.

1. $\mathbf{z} \sim N(0, I) \qquad \mathbf{x} = A_c\mathbf{z} + \boldsymbol{\varepsilon}$*,*

2. $\mathbf{z} \sim N(0, I) \qquad \mathbf{x} = \begin{cases} A_0\mathbf{z} + \boldsymbol{\varepsilon} \textit{ if } c = 0 \\ A_1R\mathbf{z} + \boldsymbol{\varepsilon} \textit{ if } c = 1, \end{cases}$

where A_0, A_1 *are matrices,* R *is a non-trivial rotation matrix and* $\boldsymbol{\varepsilon}$ *is Gaussian observation noise independent of* $\mathbf{z}$*. Assume* $A_1R \neq A_1$*. Then models 1. and 2. fit unpaired training data* $(\mathbf{x}_i, c_i)_{i=1}^{\infty}$ *equally well, but give different counterfactual predictions.*

Proof. For the two models to fit the training data equally well, we need class-conditional likelihoods $p(\mathbf{x}|c)$ to be equal for each c. For $c = 0$, the models are equal. For $c = 1$, we use the fact that $R\mathbf{z} \overset{d}{=} \mathbf{z}$ since $\mathbf{z}$ is an isotropic Gaussian. This gives

$$p_{\text{model 2.}}(\mathbf{x}|c = 1) = \mathbb{E}_{\mathbf{z} \sim N(0,I)}[p_{A_1}(\mathbf{x}|R\mathbf{z})] = \mathbb{E}_{\mathbf{z} \sim N(0,I)}[p_{A_1}(\mathbf{x}|\mathbf{z})] = p_{\text{model 1.}}(\mathbf{x}|c = 1) \tag{58}$$

where $p_A(\mathbf{x}|\mathbf{z}) = p_{\boldsymbol{\varepsilon}}(\mathbf{x} - A\mathbf{z})$. Thus, both models fit the training data equally well. To see that the models give different counterfactual predictions, we have

$$\hat{p}_{\text{model 1.}}(\mathbf{x}_{c=1}|\mathbf{x}, c=0) = \int p_{\text{model 1.}}(\mathbf{z}|\mathbf{x}, c=0) p_{A_1}(\mathbf{x}|\mathbf{z})\, d\mathbf{z} \tag{59}$$

$$\begin{aligned} \hat{p}_{\text{model 2.}}(\mathbf{x}_{c=1}|\mathbf{x}, c=0) &= \int p_{\text{model 2.}}(\mathbf{z}|\mathbf{x}, c=0) p_{A_1}(\mathbf{x}|R\mathbf{z})\, d\mathbf{z} \\ &= \int p_{\text{model 1.}}(\mathbf{z}|\mathbf{x}, c=0) p_{A_1}(\mathbf{x}|R\mathbf{z})\, d\mathbf{z}. \end{aligned} \tag{60}$$

Since $A_1 R \neq A_1$, we see that there exist $\mathbf{x}$ for which $\hat{p}_{\text{model 1.}}(\mathbf{x}_{c=1}|\mathbf{x}, c=0) \neq \hat{p}_{\text{model 2.}}(\mathbf{x}_{c=1}|\mathbf{x}, c=0)$. Thus, counterfactual predictions under the two models are different. □

One way to understand Proposition 5 is that the $N(0, I)$ prior has a high degree of symmetry, which means there is a high degree of indeterminacy in $\mathbf{z}$. The relevance of the rotational symmetry of the $N(0, I)$ distribution to VAE models has previously been discussed by Mathieu et al. (2019); Rolinek et al. (2019), here we focus on the CVAE and the consequences for counterfactual inference.

This intuition also indicates why Proposition 5 is not the end of the story. In our practical applications in Sec. 6, we find that the latent space distribution $q(\mathbf{z})$ is not, in fact, an isotropic Gaussian. Note that, in this case, the example in the Proposition breaks down. If we consider model 2. with decoder $A_1 R$ and $\mathbf{z} \sim q(\mathbf{z})$, then the predictive distribution $p_{\text{model 2.}}(\mathbf{x}|c=1)$ will be different from $p_{\text{model 1.}}(\mathbf{x}|c=1)$. If instead we consider a different latent distribution $R^{-1} \circ q(\mathbf{z})$ which arises when we apply R^{-1} to $q(\mathbf{z})$, then $p_{\text{model 2.}}(\mathbf{x}|c=1)$ will be correct *but* we will no longer have $\mathbf{z} \perp\!\!\!\perp c$.

Experimentally, we also find that counterfactual inference is stable between different training seeds, and that it accords extremely well with counterfactuals that are estimated using held-out cell type information. This indicates that there is a mismatch between the basic theory and application. Our approach to theoretically analyse CoMP is therefore inspired by our practical observation that $q(\mathbf{z})$ is rarely equal to an isotropic Gaussian in practice. To make the theory more realistic, we relax the assumption that, in the true data generating distribution, the latent distribution is $N(0, I)$.

E.1. Counterfactual identifiability for CoMP and other CVAE methods

We now assume that the true data generating distribution has $\mathbf{z} \sim r(\mathbf{z})$, a non-Gaussian distribution. The assumption of non-Gaussianity permits us to make the connection to the theory of independent component analysis (ICA) (Hyvarinen et al., 2001), in which the non-Gaussianity assumption is of critical importance. ICA has successfully been applied to causal discovery (Shimizu et al., 2006) and inference (Moneta et al., 2013). To simplify the exposition, we initially focus on a generative model with a noiseless linear decoder as follows

$$\mathbf{z} \sim r(\mathbf{z}) \qquad \mathbf{x} = A_c \mathbf{z} \tag{61}$$

and we further make the ICA assumption that

$$\mathbf{z} = B\mathbf{s} \tag{62}$$

where the components s_i of $\mathbf{s}$ are independent and non-Gaussian of unit variance.

We begin by translating the standard linear ICA theory into our setting. The standard ICA setting applies to data from a single condition. For condition $c = 0$, we have $\mathbf{x} = A_0 B\mathbf{s}$. The theory of ICA tells us that $\mathbf{s}$ and $A_0 B$ are likelihood identifiable, up to permutation and negation.

Theorem 6 (Comon (1994))**.** *Assume $\mathbf{x} = D\mathbf{s}$ where s_i are mutually independent and non-Gaussian with variance 1. Then D is likelihood identifiable up to the following*

- *we can apply a* permutation *to the columns of D by right multiplication by a permutation matrix P,*
- *we can* negate *columns of D by right multiplication by a matrix N with diagonals entries either 1 or -1 and 0 elsewhere.*

It is clear why these two indeterminacies exist: given $\mathbf{x} = D\mathbf{s}$ we also have $\mathbf{x} = DPP^{-1}\mathbf{s}$ and $\mathbf{x} = DNN^{-1}\mathbf{s}$, and $P^{-1}\mathbf{s}$ and $N^{-1}\mathbf{s}$ satisfy all the conditions of the theorem.

Our existing assumptions are almost enough to guarantee counterfactual identifiability. To gain intuition, suppose that $B = I$ and that A_0, A_1 are fully identifiable (setting aside the indeterminacies for a moment). Then we would have

$$\mathbf{x}_{c=1}|(\mathbf{x}, c = 0) = A_1 A_0^{-1}\mathbf{x}. \tag{63}$$

Although A_0, A_1 are not fully identifiable, we cannot apply arbitrary permutations and negations to A_0 and A_1 separately and still have the condition $\mathbf{z} \perp\!\!\!\perp c$ hold. Suppose we make the substitutions

$$A_0 \mapsto A_0 P_0 \qquad A_1 \mapsto A_1 P_1 \tag{64}$$

then the implied distributions for $\mathbf{z}$ must also be changed as

$$r(\mathbf{z}|c = 0) \mapsto P_0^{-1} \circ r(\mathbf{z}|c = 0) \qquad r(\mathbf{z}|c = 1) \mapsto P_1^{-1} \circ r(\mathbf{z}|c = 1). \tag{65}$$

If $P_0 = P_1$, then the distributions remain equal, but the counterfactual predictions are also unchanged since

$$(A_1 P_1)(A_0 P_0)^{-1} = A_1 P_1 P_0^{-1} A_0^{-1} = A_1 A_0^{-1}. \tag{66}$$

On the other hand, if $P_0 \neq P_1$ then applying different permutations would lead to two different distributions for $r(\mathbf{z}|c = 0)$ and $r(\mathbf{z}|c = 1)$, violating $\mathbf{z} \perp\!\!\!\perp c$. There is one more case to be eliminated. That is the case where P_0 and P_1 are different, but due to a symmetry of the distribution $r(\mathbf{z})$, the condition $\mathbf{z} \perp\!\!\!\perp c$ still holds. The following assumption excludes this possibility and formalises our assumptions for $r(\mathbf{z})$.

Assumption 7. *Assume that $\mathbf{z} \sim r(\mathbf{z})$ can be expressed as $\mathbf{z} = B\mathbf{s}$ where the components s_i of $\mathbf{s}$ are independent and non-Gaussian of unit variance. Furthermore, assume that for every permutation matrix P and negation matrix N such that PN is not the identity we have*

$$PN\mathbf{s} \overset{d}{\neq} \mathbf{s}. \tag{67}$$

This assumption is enough to prove the following theorem.

Theorem 2. *Suppose the true data generating distribution has $\mathbf{z} \sim r(\mathbf{z})$ and linear decoders for each condition. Assume $r(\mathbf{z})$ is non-Gaussian and that Assumption 7 holds. Then counterfactuals are identifiable from unpaired data.*

Proof. We have the model $\mathbf{z} \sim r(\mathbf{z})$ and $\mathbf{x} = A_c\mathbf{z}$. The true counterfactual predictions are

$$\mathbf{x}_{c=1}|(\mathbf{x}, c = 0) = A_1 A_0^{-1}\mathbf{x}, \tag{68}$$

hence it is enough to show that $A_1 A_0^{-1}$ is likelihood identifiable. Since $\mathbf{x}|c = A_c\mathbf{z} = A_c B\mathbf{s} = D_c\mathbf{s}$ and

$$D_1 D_0^{-1} = (A_1 B)(A_0 B)^{-1} = A_1 A_0^{-1} \tag{69}$$

it is enough to show that $D_1 D_0^{-1}$ is likelihood identifiable. By Theorem 6, D_c are identifiable from unpaired data up to permutation and negation. Suppose we are able to identify $D_c P_c N_c$. (Note: permutation and negation matrices together form a group, so $P_c N_c$ is a general composition of permutations and negations.) Since $\mathbf{s} \perp\!\!\!\perp c$ in the identified model, we must have

$$(P_0 N_0)^{-1}\mathbf{s} \overset{d}{=} (P_1 N_1)^{-1}\mathbf{s} \tag{70}$$

which further implies

$$(P_1 N_1)(P_0 N_0)^{-1}\mathbf{s} \overset{d}{=} \mathbf{s}. \tag{71}$$

It is possible to express $(P_1 N_1)(P_0 N_0)^{-1}$ as a product PN of a permutation and a negation. Applying Assumption 7, we must have

$$(P_1 N_1)(P_0 N_0)^{-1} = I. \tag{72}$$

Hence

$$D_1 P_1 N_1 (D_0 P_0 N_0)^{-1} = D_1 (P_1 N_1)(P_0 N_0)^{-1} D_0^{-1} = D_1 I D_0^{-1} = D_1 D_0^{-1}. \tag{73}$$

Thus $A_1 A_0^{-1}$ is identifiable. □

It is unnecessary to compute the independent components An important aspect of the Theorem is that it is *not* necessary to compute the independent components $\mathbf{s}$ and the matrices D_c. For the problem of counterfactual identifiability, it is sufficient that such matrices exist. For computation, we can rely on A_0 and A_1.

Extensions of the proof Our proof directly extends to the case for more than two conditions. It is possible to adapt the proof by considering the existing theory for Noisy ICA (Hyvarinen, 1999) and nonlinear ICA (Hyvarinen et al., 2001). The latter has been explored in the context of VAE models by Khemakhem et al. (2020), and presents interesting challenges compared to the linear theory.

Theorem 2 is a key theorem because it provides circumstances under which a CVAE type model can identify counterfactuals. The requirement $\mathbf{z} \perp\!\!\!\perp c$ can be seen as a significant weakening of the assumption $\mathbf{z} \sim N(0, I)$, whilst Assumption 7 introduces necessary constraints on $\mathbf{z}$ for identifiability to hold.

E.2. Consistency of CoMP under prior misspecification

We have seen that CVAE-type models, including CoMP, can identify counterfactuals, but we now assume that $\mathbf{z}$ be of some unknown non-Gaussian distribution rather than an isotropic Gaussian. This is at odds with the CoMP training objective, which uses a Gaussian prior. Rather than altering the objective and learning a non-Gaussian prior (which is one valid approach), we instead rely on our experience, which shows that the CoMP training objective does not lead to $q_\phi(\mathbf{z})$ being a Gaussian.

We will show that, under certain conditions, training with the unaltered CoMP objective when the true data generating distribution has $\mathbf{z} \sim r(\mathbf{z})$ allows us to find the correct decoder network, and further the marginal distribution of the encoder $q_\phi(\mathbf{z})$ matches $r(\mathbf{z})$.

Theorem 3. *Define $p_{r,\theta}(\mathbf{x}|c) = \mathbb{E}_{r(\mathbf{z})}[p_\theta(\mathbf{x}|\mathbf{z}, c)]$. There exists a constant K_1 such that, if $\mathrm{KL}[r(\mathbf{z})\|p(\mathbf{z})] \leq K_1$ and if the encoder network is sufficiently flexible, then maximising the CoMP objective with infinite data generated under the misspecified model with $\mathbf{z} \sim r(\mathbf{z})$ leads to a θ_∞ that is a maximum point of $\mathbb{E}_{\mathbf{x},c}[\log p_{r,\theta}(\mathbf{x}|c)]$.*

Proof. We begin by writing out the CoMP training objective with a data batch of size n

$$\mathcal{L}_n^{\text{CoMP}} = \frac{1}{n}\sum_{i=1}^{n}\left[\log \frac{p_\theta(\mathbf{x}_i|\mathbf{z}_i, c_i)p(\mathbf{z}_i)}{q_\phi(\mathbf{z}_i|\mathbf{x}_i, c_i)} - \gamma \log\left(\frac{\frac{1}{|I_{c_i}|}\sum_{j\in I_{c_i}} q_\phi(\mathbf{z}_i|\mathbf{x}_j, c_i)}{\frac{1}{|I_{\neg c_i}|}\sum_{j\in I_{\neg c_i}} q_\phi(\mathbf{z}_i|\mathbf{x}_j, c_j)}\right)\right]. \tag{74}$$

Suppose that the true data generating distribution is

$$\mathbf{z} \sim r(\mathbf{z}) \qquad \mathbf{x}|c \sim p_{\theta^*}(\mathbf{x}|\mathbf{z}, c). \tag{75}$$

We define

$$p_{r,\theta}(\mathbf{x}|c) = \mathbb{E}_{r(\mathbf{z})}[p_\theta(\mathbf{x}|\mathbf{z}, c)] \tag{76}$$

and

$$r_\theta(\mathbf{z}|\mathbf{x}, c) = \frac{r(\mathbf{z})p_\theta(\mathbf{x}|\mathbf{z}, c)}{p_{r,\theta}(\mathbf{x}|c)}. \tag{77}$$

Then we can rewrite the ELBO part of the objective as

$$\frac{1}{n}\sum_{i=1}^{n}\log\frac{p_\theta(\mathbf{x}_i|\mathbf{z}_i, c_i)p(\mathbf{z}_i)}{q_\phi(\mathbf{z}_i|\mathbf{x}_i, c_i)} = \frac{1}{n}\sum_{i=1}^{n}\log\frac{p_\theta(\mathbf{x}_i|\mathbf{z}_i, c_i)p(\mathbf{z}_i)r(\mathbf{z}_i)p_{r,\theta}(\mathbf{x}_i|c_i)}{q_\phi(\mathbf{z}_i|\mathbf{x}_i, c_i)r(\mathbf{z}_i)p_{r,\theta}(\mathbf{x}_i|c_i)} \tag{78}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\log p_{r,\theta}(\mathbf{x}_i|c_i) - \log\frac{q_\phi(\mathbf{z}_i|\mathbf{x}_i, c_i)p_{r,\theta}(\mathbf{x}_i, c_i)}{r(\mathbf{z}_i)p_\theta(\mathbf{x}_i|\mathbf{z}_i, c_i)} - \log\frac{r(\mathbf{z}_i)}{p(\mathbf{z}_i)} \tag{79}$$

take the expectation over $\mathbf{z}_i \sim q_\phi(\mathbf{z}|\mathbf{x}_i, c_i)$

$$= \frac{1}{n}\sum_{i=1}^{n}\log p_{r,\theta}(\mathbf{x}_i|c_i) - \mathrm{KL}[q_\phi(\mathbf{z}_i|\mathbf{x}_i, c_i)\|r_\theta(\mathbf{z}_i|\mathbf{x}_i, c_i)] - \mathbb{E}_{q_\phi(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log\frac{r(\mathbf{z}_i)}{p(\mathbf{z}_i)}\right]. \tag{80}$$

Consider the augmented objective in which we introduce a coefficient ζ on the final term, and we reintroduce the CoMP misalignment penalty

$$\begin{aligned}\mathbb{E}[\mathcal{L}_n^{\text{CoMP}}(\theta,\phi)] =& \frac{1}{n}\sum_{i=1}^n \log p_{r,\theta}(\mathbf{x}_i|c_i) - \mathrm{KL}[q_\phi(\mathbf{z}_i|\mathbf{x}_i,c_i)\|r_\theta(\mathbf{z}_i|\mathbf{x}_i,c_i)]\\ &- \zeta\frac{1}{n}\sum_{i=1}^n \mathbb{E}_{q_\phi(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log\frac{r(\mathbf{z}_i)}{p(\mathbf{z}_i)}\right] \\ &- \gamma\frac{1}{n}\sum_{i=1}^n \mathbb{E}_{q_\phi(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log\left(\frac{\frac{1}{|I_{c_i}|}\sum_{j\in I_{c_i}} q_\phi(\mathbf{z}_i|\mathbf{x}_j,c_i)}{\frac{1}{|I_{\neg c_i}|}\sum_{j\in I_{\neg c_i}} q_\phi(\mathbf{z}_i|\mathbf{x}_j,c_j)}\right)\right].\end{aligned} \tag{81}$$

Treating ζ, γ as Lagrange multipliers, this is equivalent to the following constrainted optimisation problem

$$\max_{\theta,\phi} \frac{1}{n}\sum_{i=1}^n \log p_{r,\theta}(\mathbf{x}_i|c_i) - \mathrm{KL}[q_\phi(\mathbf{z}_i|\mathbf{x}_i,c_i)\|r_\theta(\mathbf{z}_i|\mathbf{x}_i,c_i)] \tag{82}$$

$$\text{subject to } \frac{1}{n}\sum_{i=1}^n \mathbb{E}_{q_\phi(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log\frac{r(\mathbf{z}_i)}{p(\mathbf{z}_i)}\right] \le K_\zeta \text{ and} \tag{83}$$

$$\frac{1}{n}\sum_{i=1}^n \mathbb{E}_{q_\phi(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log\left(\frac{\frac{1}{|I_{c_i}|}\sum_{j\in I_{c_i}} q_\phi(\mathbf{z}_i|\mathbf{x}_j,c_i)}{\frac{1}{|I_{\neg c_i}|}\sum_{j\in I_{\neg c_i}} q_\phi(\mathbf{z}_i|\mathbf{x}_j,c_j)}\right)\right] \le L_\gamma. \tag{84}$$

Neglecting the constraints for a moment, if we maximise the main objective with respect to ϕ with a sufficiently expressive encoder, we will recover $q_\phi(\mathbf{z}|\mathbf{x},c) = r_\theta(\mathbf{z}|\mathbf{x},c)$ in the limit $n\to\infty$. We are then left with the maximum likelihood problem

$$\max_\theta \frac{1}{n}\sum_{i=1}^n \log p_{r,\theta}(\mathbf{x}_i|c_i), \tag{85}$$

thus θ will recover a likelihood-maximising decoder under the *misspecified* prior $r(\mathbf{z})$.

This solution will be valid if the constraints are satisfied at that solution. Note that in the limit $n\to\infty$, Theorem 1 shows that the CoMP misalignment penalty becomes

$$\sum_{c\in\mathcal{C}} p(c)\,\mathrm{KL}\left[q_\phi(\mathbf{z}|c)||q_\phi(\mathbf{z}|\neg\, c)\right]. \tag{86}$$

Since $q_\phi(\mathbf{z}|\mathbf{x},c) = r_\theta(\mathbf{z}|\mathbf{x},c)$, we have $q_\phi(\mathbf{z}|c) = r(\mathbf{z})$ for every c, hence the misalignment penalty is equal to 0 and the constraint is satisfied. In this limit, we also have

$$\frac{1}{n}\sum_{i=1}^n \mathbb{E}_{q_\phi(\mathbf{z}_i|\mathbf{x}_i,c_i)}\left[\log\frac{r(\mathbf{z}_i)}{p(\mathbf{z}_i)}\right] \to \mathrm{KL}[r(\mathbf{z})\|p(\mathbf{z})]. \tag{87}$$

Therefore, returning to the original problem with $\zeta = 1$, provided that $\mathrm{KL}[r(\mathbf{z})\|p(\mathbf{z})] \le K_1$, then the constraints are satisfied under unconstrained optimisation in (82), meaning that θ tends to a maximum likelihood solution of $\mathbb{E}_{\mathbf{x},c}[\log p_{r,\theta}(\mathbf{x}|c)]$. □

The result of Theorem 3 is directly connected to our previous discussion about ICA, since ICA is a solution to the maximum likelihood problem (Hyvarinen et al., 2001). However, the consistency theorem is much more general, since it applies to a much wider range of models, and does not assume a linear decoder. Theorem 3 also justifies our distinction in Theorem 2 between $\mathbf{z}$ and $\mathbf{s}$. In trying to satisfy the condition $\mathrm{KL}[r(\mathbf{z})\|p(\mathbf{z})] \le K_1$, we are allowed to apply any linear transformation to $\mathbf{z}$. If any linear transformation of $\mathbf{z}$ satisfies $\mathrm{KL}[r(\mathbf{z})\|p(\mathbf{z})] \le K_1$, then consistency holds true.

F. Experimental details

The code to reproduce our experiments is available at `https://github.com/BenevolentAI/CoMP`.

F.1. Dataset details and data processing

Tumour / Cell Line This dataset, as used in the experiments in *Cellinger* (Warren et al., 2021)[4], consists of bulk expression profiles for tumours ($n = 12,236$) and cancer cell-lines ($n = 1,249$) across 39 different cancer types. The tumour samples are taken from The Cancer Genome Atlas (TCGA) (Weinstein et al., 2013)[5] and Therapeutically Applicable Research To Generate Effective Treatments (TARGET) (Gerhard et al., 2018)[6] and were compiled by the Treehouse Childhood Cancer Initiative at the UC Santa Cruz Genomics Institute (Goldman et al., 2020)[7]. The cell lines are from the Cancer Cell Line Encyclopedia (CCLE) (Ghandi et al., 2019)[8]. The condition variable is the tumour / cell line label. The expression data is restricted to the intersecting subset of 16,612 protein-coding genes and are TPM $\log_2$-transformed values.

In our experiments, as is common practice in omics data analysis (e.g. Warren et al. (2021)), we pre-process the data by filtering out low-variance genes. Here, we select the 8,000 highest variance genes across cell-lines and tumours separately and take the union to give a final feature set of 9,468 genes.

For our calculation of m-kBET$_{k,\alpha}$ and $\tilde{s}_{k,c}$ metrics we only include cancer types with at least 400 samples (i.e. $4 \times k$ for our choice of $k = 100$) to ensure that the metric retains the ability to evaluate *local* mixing. 15 cancer types pass this threshold, representing 82% of all samples.

Stimulated / untreated single-cell PBMCs This dataset consists of single-cell expression profiles of 14,053 genes for peripheral blood mononuclear cells (PBMCs), various immune cell types pooled from eight lupus patient samples. 7,217 of the cells were stimulated with interferon (IFN)-β while 6,359 were left untreated (control) (Kang et al., 2018). This dataset has been used in Lotfollahi et al. (2019b) and Lotfollahi et al. (2019a) previously. We obtained an annotated and pre-filtered dataset from Lotfollahi et al. (2019a)[9] [10] , which includes metadata on immune cell type labels along the condition label; stimulated or control.

The file was read into scanpy (Wolf et al., 2018) and pre-processed using `sc.pp.normalize_total(data, inplace=True)`, which normalises the data such that each cell has a total count equal to the median total count across all cells. The normalised counts were then $\log(x + 1)$ transformed using the scanpy function, `sc.pp.log1p(data)`. We selected the top 2,000 most variable genes using the scanpy function, `sc.pp.highly_variable_genes(data, flavor="seurat", n_top_genes=2000)`.

We obtained the top 50 differentially expressed (DE) genes between stimulated and control cells for each cell type by subsetting the data for each cell type and using scanpy's function `sc.tl.rank_genes_groups(cell_type_data, groupby="stim", n_genes=50, method="wilcoxon")`, which ranks genes based on a Wilcoxon rank-sum test. For each cell type, we separated the top 50 DE genes into those that were up-regulated and down-regulated by IFN-β stimulation.

Single-cell RNA-seq data integration This dataset consists of single-cell RNA count measurements for 33,694 genes in 21,463 PBMCs that were processed using different library preparation protocols; 3-prime V1 (4,809 cells), 3-prime V2 (8,380 cells) and 5-prime (7,697 cells). This dataset has been used in Korsunsky et al. (2019) previously.

We obtained the data in binary format (RDS) and associated cell sub-type metadata from Korsunsky et al. (2019)[11]. We selected the cells that were processed using 3-prime V2 and 5-prime library preparation methods and filtered out mitochondrial genes and those that had zero counts across all cells in the two library preparation methods. This resulted in single-cell expression data for 16,077 cells and 23,338 genes, after filtering.

Using R version 3.6.3 and Seurat 3.0.2, we normalised via `NormalizeData(x, normalization.method = "LogNormalize", scale.factor = 10000)` and identified the most variable genes using

[4] www.nature.com/articles/s41467-020-20294-x#data-availability

[5] www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga

[6] ocg.cancer.gov/programs/target

[7] https://treehousegenomics.soe.ucsc.edu/public-data/previous-compendia.html#tumor_v10_polyA

[8] portals.broadinstitute.org/ccle

[9] https://github.com/theislab/trVAE_reproducibility

[10] https://drive.google.com/drive/folders/1n1SLbXha4OH7j7zZ0zZAxrj_-2kczgl8, filename: kang_count.h5ad

[11] https://github.com/immunogenomics/harmony2019/tree/master/data/figure4

`FindVariableFeatures(x, selection.method = "vst", nfeatures = 2000)` for each library preparation protocol independently. Then, we applied Seurat's `SelectIntegrationFeatures` function to select the top 2000 genes that were repeatedly variable across library preparation protocols.

We applied Seurat's canonical correlation analysis (CCA) method (Stuart et al., 2019) to integrate the data from the two library protocols (using Seurat's `FindIntegrationAnchors` and `IntegrateData` functions) and extracted the resulting embeddings. Using Harmony version 0.1.0 (Korsunsky et al., 2019) we integrated the data from the two library protocols using the `RunHarmony` function and extracted the resulting embeddings.

UCI Adult Income This dataset is derived from the 1994 United States census bureau and contains information relating to education, marriage status, ethnicity, self-reported gender of census participants and a binary high / low income label ($50,000 threshold). Data was downloaded from the UCI Machine Learning Repository (Dua & Graff, 2017).

F.2. Evaluation metrics

Let $\{(\mathbf{x}_i\, c_i, d_i)\}_{i=1}^N$ be the dataset N samples with $c_i \in \{0,1\}$ the binary condition variable and $d_i \in \{d^{(m)}\}_{m=1}^M$ an additional discrete random variable of interest not used in training.

We start with some housekeeping definitions of sample index subsets of the full dataset $[1, N]$. Let $N_{i,k}$ be the index set of the k nearest-neighbours of z_i. Let I_c be the index set of samples has $c_j = c$, and J_d for samples that has $d_j = d$.

From Büttner et al. (2019), kBET$_{k,\alpha}$ is the proportion of rejected null hypotheses from the set of separate χ^2 independence tests, with significance threshold α, on the k nearest-neighbours of every sample. If we let kBET$^d_{k,\alpha}$ be the metric calculated on the filtered sub-population with index set J_d, then we define a mean kBET metric as

$$\text{m-kBET}_{k,\alpha} := \frac{1}{M}\sum_{m=1}^{M} \text{kBET}^{d^{(m)}}_{k,\alpha}. \tag{88}$$

We also consider local Silhouette Coefficients (Rousseeuw, 1987)

$$s_{k,c} := \frac{1}{|I_c|}\sum_{i\in I_c}\frac{b_{i,k}-a_{i,k}}{\max(a_{i,k},b_{i,k})}, \quad s_k := \frac{1}{|I_c \cup I_{\neg c}|}\sum_{i\in I_c\cup I_{\neg c}}\frac{b_{i,k}-a_{i,k}}{\max(a_{i,k},b_{i,k})}, \tag{89}$$

where $a_{i,k}$ and $b_{i,k}$ are the mean Euclidean distances between $\mathbf{z}_i$ and all other sample points in the k nearest-neighbour set that are of the same and different condition variable respectively; i.e.

$$a_{i,k} \equiv \frac{1}{|N_{i,k}\cap I_{c_i}|}\sum_{j\in N_{i,k}\cap I_{c_i}} \|\mathbf{z}_i - \mathbf{z}_j\|, \qquad b_{i,k} \equiv \frac{1}{|N_{i,k}\cap I_{\neg c_i}|}\sum_{j\in N_{i,k}\cap I_{\neg c_i}} \|\mathbf{z}_i - \mathbf{z}_j\|. \tag{90}$$

Similar to the mean kBET metric, we can also define a mean Silhouette Coefficient $\tilde{s}_{k,c}$ as follows. We first define

$$s^d_{k,c} := \frac{1}{|I_c \cap J_d|}\sum_{i\in I_c\cap J_d}\frac{b_{i,k,d}-a_{i,k,d}}{\max(a_{i,k,d},b_{i,k,d})}, \tag{91}$$

with

$$\begin{aligned} a_{i,k,d} &\equiv \frac{1}{|N_{i,k}\cap I_{c_i}\cap J_d|}\sum_{j\in N_{i,k}\cap I_{c_i}\cap J_d} \|\mathbf{z}_i - \mathbf{z}_j\|, \\ b_{i,k,d} &\equiv \frac{1}{|N_{i,k}\cap I_{\neg c_i}\cap J_d|}\sum_{j\in N_{i,k}\cap I_{\neg c_i}\cap J_d} \|\mathbf{z}_i - \mathbf{z}_j\|. \end{aligned} \tag{92}$$

Then the mean local Silhouette Coefficient is

$$\tilde{s}_{k,c} := \frac{1}{M}\sum_{m=1}^{M} s^{d^{(m)}}_{k,c}, \tag{93}$$

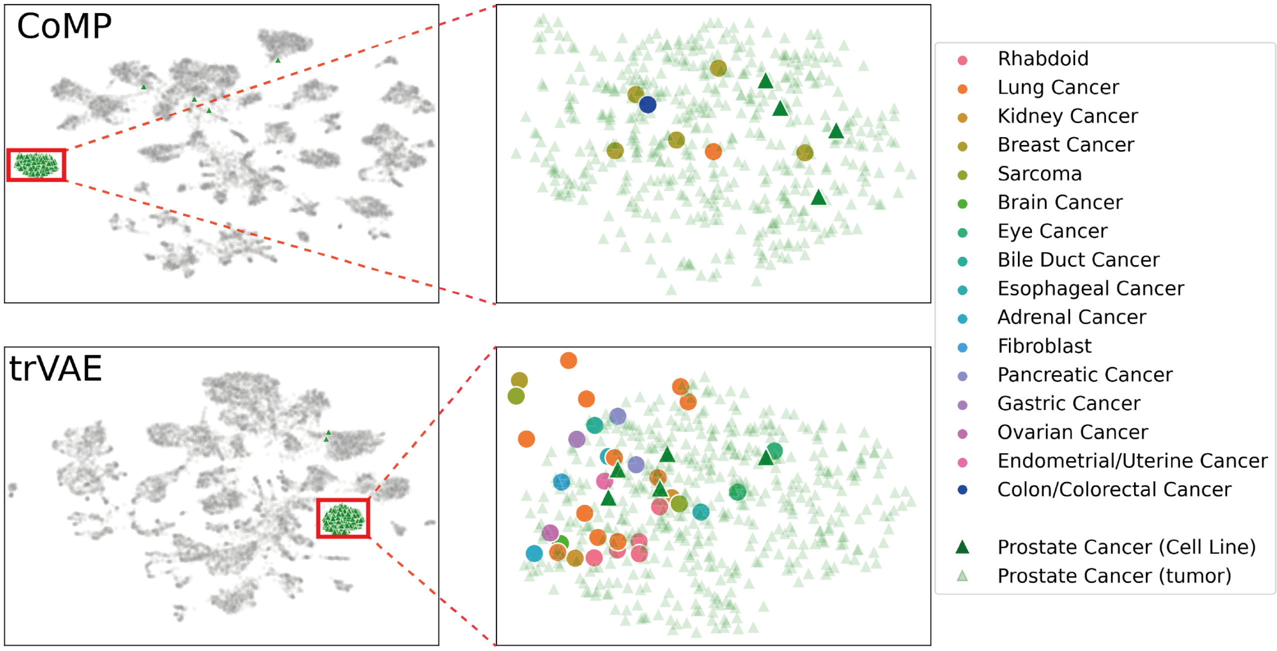

Figure 5. 2D UMAP projection of the CoMP and trVAE posterior means of $\mathbf{z}_i$ from Tumour / Cell Line data and the detailed Prostate Cancer tumour sample clusters.

with $\tilde{s}_k$ defined analogously to s_k in (89). A well-mixed representation that keeps samples with identical d_i together will have low values of m-kBET$_{k,\alpha}$ and $\tilde{s}_{k,c}$ close to zero. Higher values near 1 would indicate either an undesirable dependency between $\mathbf{z}$ and c in the form of identifiable clusters around values of c, a censoring process that fails to preserve the clustering with respect to d, or a combination of both.

F.3. Tumour / Cell Line representations for individual cancer types

As the cancer type labels are not used in training, there is the possibility that cell lines of one cancer type will cluster around tumours of a different type. Here we illustrate this risk by examining the subset of Prostate Cancer latent representations inferred by CoMP and trVAE, where the majority of tumours and cell lines for this cancer type can be found in a single group. As shown in Figure 5, trVAE has cell-lines from other cancer types erroneously placed within the prostate cancer cluster; CoMP, on the other hand, maintains a relatively high level of specificity with fewer non-prostate cancer cell lines present. On average across all cancer types, this favourable behaviour of CoMP is reflected in the low $\tilde{s}$ and m-kBET scores.

F.4. Condition mixing metrics for stimulated / untreated single-cell PBMCs

In this section we present additional results of our experiments on the stimulated / untreated single-cell PBMCs scRNA-seq data evaluating the condition mixing capabilities of CoMP. We focus on the two mixing metrics – s_k and kBET$_{k,\alpha}$ – and report both the mean values and their standard errors over 10 random model initialisations. We have the following three sets of experiments:

Benchmarking In Tables 4 and 5 we benchmark CoMP against the four other VAE models and show that CoMP outperforms the other models by significant margins on both metrics.

Cell type level evaluation In Table 6 we evaluate the mixing at a cell type level, where the strong mixing capabilities of CoMP is seen consistently across cell types. In particular, we highlight the good mixing of the CD14 Mono cell type by CoMP relative to the other penalised models.

Table 4. kBET metrics for stimulated / untreated single-cell PBMCs expression dataset with $k = 100$ and $\alpha = 0.1$. Here, kBET and m-kBET refer to the mean kBET and mean m-kBET across 10 random seeds for each model, respectively. SEM represents the standard error of the mean.

Model	kBET$_{k,\alpha}$	kBET$_{k,\alpha}$ ± SEM	m-kBET$_{k,\alpha}$	m-kBET$_{k,\alpha}$ ± SEM
VAE	0.9788	(0.9754, 0.9821)	0.9443	(0.9351, 0.9535)
CVAE	0.9056	(0.8973, 0.9139)	0.8202	(0.8060, 0.8344)
VFAE	0.4753	(0.4660, 0.4847)	0.4067	(0.3942, 0.4192)
trVAE	0.5082	(0.4946, 0.5218)	0.3819	(0.3683, 0.3955)
CoMP	0.1211	(0.0845, 0.1577)	0.0681	(0.0388, 0.0975)

Table 5. Silhouette Coefficient metrics for stimulated / untreated single-cell PBMCs expression dataset with $k = 100$. Here, s and $\tilde{s}$ refer to the mean s and mean $\tilde{s}$ across 10 random seeds for each model, respectively. SEM represents the standard error of the mean.

Model	s_k	s_k ± SEM	$\tilde{s}_k$	$\tilde{s}_k$ ± SEM
VAE	0.6354	(0.6303, 0.6404)	0.5249	(0.5172, 0.5326)
CVAE	0.4872	(0.4805, 0.4939)	0.3856	(0.3802, 0.3910)
VFAE	0.0501	(0.0457, 0.0544)	0.0793	(0.0731, 0.0855)
trVAE	0.0651	(0.0596, 0.0705)	0.0605	(0.0574, 0.0636)
CoMP	-0.0026	(-0.0032, -0.0020)	-0.0013	(-0.0027, 0.0001)

CoMP penalty scale γ In Tables 7 to 9 we explore the effect of varying the CoMP penalty scale γ at both the population and cell type levels. Here we see that the optimum value is ~ 1.

F.5. Counterfactual prediction of stimulated / untreated single-cell PBMCs expression dataset (IFN-β stimulation)

In this section we present the full results on the counterfactual prediction of single-cell PBMC expression data under IFN-β stimulation. In Tables 10 and 11, we present the mean and standard error of the Pearson correlation coefficient and MSE metrics respectively for CoMP and the other four VAE models. We present our results for each cell type separately. As is consistent with the summary presented in Figures 3 and 6, we see that CoMP produces highly accurate counterfactual reconstructions. Indeed, this can be seen in the scatter plots showing the mean expression of (actual) stimulated cells against the mean of counterfactually stimulated control cells (Figure 7). Here, we see that the other VAE models tend to underestimate the expression of genes that are up-regulated by IFN-β stimulation and overestimate the expression of genes that are down-regulated. However, this is not as evident with CoMP.

Similar to the mixing metrics, we evaluate the effect of varying the penalty scale γ. As we see in Tables 12 and 13, the optimal value is ≈ 1.

Table 6. Cell type specific kBET and Silhouette Coefficient metrics for the stimulated / untreated single-cell PBMCs expression dataset summarised for 10 random seeds for each model. Metrics represent the mean value across the 10 random seeds for each model. Here, $k = 100$ and $\alpha = 0.1$.

Cell type	Model	kBET$_{k,\alpha}$	kBET$_{k,\alpha}$ ± SEM	s_k	s_k ± SEM
B	VAE	0.9724	(0.9683, 0.9765)	0.5375	(0.5225, 0.5526)
B	CVAE	0.9016	(0.8964, 0.9068)	0.2884	(0.2783, 0.2985)
B	VFAE	0.3892	(0.3357, 0.4426)	0.0263	(0.0205, 0.0320)
B	trVAE	0.2697	(0.2243, 0.3151)	0.0102	(0.0083, 0.0121)
B	CoMP	0.0110	(0.0002, 0.0217)	-0.0040	(-0.0050, -0.0030)
CD14 Mono	VAE	1.0000	(1.0000, 1.0000)	0.9388	(0.9356, 0.9420)
CD14 Mono	CVAE	1.0000	(1.0000, 1.0000)	0.9373	(0.9317, 0.9428)
CD14 Mono	VFAE	0.8192	(0.8084, 0.8300)	0.0548	(0.0486, 0.0610)
CD14 Mono	trVAE	0.9360	(0.9213, 0.9508)	0.1579	(0.1414, 0.1743)
CD14 Mono	CoMP	0.1709	(0.0817, 0.2601)	0.0003	(-0.0015, 0.0022)
CD16 Mono	VAE	1.0000	(1.0000, 1.0000)	0.7462	(0.7168, 0.7756)
CD16 Mono	CVAE	1.0000	(1.0000, 1.0000)	0.7455	(0.7204, 0.7706)
CD16 Mono	VFAE	0.8796	(0.8572, 0.9020)	0.2328	(0.1860, 0.2796)
CD16 Mono	trVAE	0.8059	(0.7856, 0.8263)	0.0535	(0.0493, 0.0576)
CD16 Mono	CoMP	0.0947	(0.0184, 0.1711)	0.0005	(-0.0011, 0.0021)
CD4 T	VAE	0.9964	(0.9960, 0.9968)	0.5069	(0.4932, 0.5206)
CD4 T	CVAE	0.9104	(0.9028, 0.9179)	0.2103	(0.1956, 0.2250)
CD4 T	VFAE	0.1460	(0.1358, 0.1562)	0.0035	(0.0030, 0.0039)
CD4 T	trVAE	0.2236	(0.1985, 0.2487)	0.0042	(0.0036, 0.0047)
CD4 T	CoMP	0.0538	(0.0442, 0.0634)	-0.0022	(-0.0028, -0.0016)
CD8 T	VAE	0.9041	(0.8633, 0.9448)	0.2828	(0.2714, 0.2942)
CD8 T	CVAE	0.4805	(0.4080, 0.5529)	0.0653	(0.0579, 0.0728)
CD8 T	VFAE	0.0397	(0.0307, 0.0486)	0.0094	(0.0083, 0.0106)
CD8 T	trVAE	0.0317	(0.0224, 0.0409)	0.0071	(0.0052, 0.0090)
CD8 T	CoMP	0.0634	(0.0437, 0.0830)	-0.0000	(-0.0008, 0.0008)
DC	VAE	1.0000	(1.0000, 1.0000)	0.6834	(0.6678, 0.6991)
DC	CVAE	1.0000	(1.0000, 1.0000)	0.6723	(0.6598, 0.6847)
DC	VFAE	0.7095	(0.6715, 0.7476)	0.2901	(0.2733, 0.3069)
DC	trVAE	0.6286	(0.5972, 0.6600)	0.2339	(0.2225, 0.2453)
DC	CoMP	0.0784	(0.0329, 0.1239)	0.0034	(-0.0027, 0.0096)
NK	VAE	0.9645	(0.9525, 0.9764)	0.2609	(0.2358, 0.2860)
NK	CVAE	0.8798	(0.8682, 0.8914)	0.1095	(0.0975, 0.1215)
NK	VFAE	0.1548	(0.1258, 0.1838)	0.0093	(0.0072, 0.0114)
NK	trVAE	0.1013	(0.0514, 0.1512)	0.0113	(0.0067, 0.0159)
NK	CoMP	0.0721	(0.0488, 0.0953)	-0.0025	(-0.0035, -0.0015)
T	VAE	0.7172	(0.6377, 0.7967)	0.2423	(0.2135, 0.2711)
T	CVAE	0.3891	(0.3315, 0.4466)	0.0561	(0.0459, 0.0663)
T	VFAE	0.1155	(0.0934, 0.1376)	0.0082	(0.0060, 0.0104)
T	trVAE	0.0585	(0.0354, 0.0815)	0.0061	(0.0047, 0.0075)
T	CoMP	0.0009	(0.0003, 0.0016)	-0.0062	(-0.0074, -0.0050)

Table 7. Effect of varying γ for the CoMP model on the kBET metrics for the stimulated / untreated single-cell PBMCs expression dataset. Here, kBET and m-kBET refer to the mean kBET and mean m-kBET across 10 random seeds for each value of γ, respectively. SEM represents the standard error of the mean. Here, $k = 100$ and $\alpha = 0.1$.

Model	γ	kBET$_{k,\alpha}$	kBET$_{k,\alpha}$ ± SEM	m-kBET$_{k,\alpha}$	m-kBET$_{k,\alpha}$ ± SEM
CoMP	0.25	0.2703	(0.2482, 0.2924)	0.1276	(0.1085, 0.1467)
CoMP	0.50	0.1741	(0.1438, 0.2045)	0.0763	(0.0543, 0.0983)
CoMP	1.00	0.1211	(0.0845, 0.1577)	0.0681	(0.0388, 0.0975)
CoMP	5.00	0.4426	(0.3889, 0.4963)	0.4419	(0.3827, 0.5011)
CoMP	10.00	0.5311	(0.4456, 0.6167)	0.5118	(0.4135, 0.6100)
CoMP	15.00	0.4288	(0.3511, 0.5065)	0.4614	(0.3738, 0.5489)
CoMP	20.00	0.5880	(0.5101, 0.6660)	0.6383	(0.5547, 0.7219)

Table 8. Effect of varying γ for the CoMP model on the Silhouette Coefficient metrics for the stimulated / untreated single-cell PBMCs expression dataset. Here, s and $\tilde{s}$ refer to the mean s and mean $\tilde{s}$ across 10 random seeds for each value of γ, respectively. SEM represents the standard error of the mean. Here, $k = 100$.

Model	γ	s_k	s_k ± SEM	$\tilde{s}_k$	$\tilde{s}_k$ ± SEM
CoMP	0.25	-0.0024	(-0.0028, -0.0021)	-0.0006	(-0.0018, 0.0006)
CoMP	0.50	-0.0029	(-0.0030, -0.0028)	-0.0023	(-0.0031, -0.0015)
CoMP	1.00	-0.0026	(-0.0032, -0.0020)	-0.0013	(-0.0027, 0.0001)
CoMP	5.00	0.0043	(0.0026, 0.0059)	0.0209	(0.0145, 0.0274)
CoMP	10.00	0.0028	(0.0016, 0.0039)	0.0523	(0.0300, 0.0746)
CoMP	15.00	0.0046	(0.0025, 0.0067)	0.0750	(0.0484, 0.1016)
CoMP	20.00	0.0061	(0.0038, 0.0083)	0.1319	(0.1053, 0.1584)

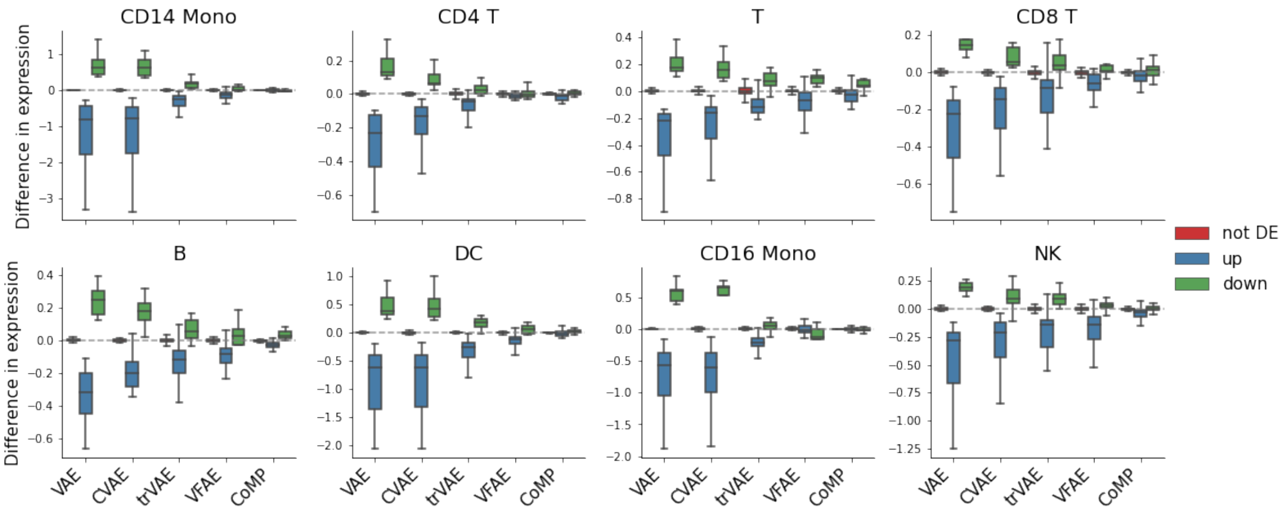

Figure 6. The difference in gene expression values for 1950 non-differentially expressed genes (red) and the top 50 differentially expressed genes (up-regulated: blue, down-regulated: green) between IFN-β stimulated cells and counterfactually stimulated control cells for each cell type. The difference in expression for a gene is the gene mean expression across stimulated cells of a cell type minus the mean reconstructed gene expression for counterfactually stimulated control cells of the same cell type.

Table 9. Effect of varying γ for cell type specific kBET and s metrics for the stimulated / untreated single-cell PBMCs expression dataset. Metrics represent the mean value across the 10 random seeds. Here, $k = 100$ and $\alpha = 0.1$.

Cell type	Model	γ	kBET$_{k,\alpha}$	kBET$_{k,\alpha}$ ± SEM	s_k	s_k ± SEM
B	CoMP	0.25	0.0061	(0.0037, 0.0086)	-0.0046	(-0.0053, -0.0038)
B	CoMP	0.50	0.0030	(0.0004, 0.0056)	-0.0033	(-0.0039, -0.0028)
B	CoMP	1.00	0.0110	(0.0002, 0.0217)	-0.0040	(-0.0050, -0.0030)
B	CoMP	5.00	0.4860	(0.3881, 0.5840)	0.0094	(0.0073, 0.0115)
B	CoMP	10.00	0.5321	(0.4226, 0.6415)	0.0195	(0.0118, 0.0273)
B	CoMP	15.00	0.5229	(0.4331, 0.6127)	0.0167	(0.0109, 0.0224)
B	CoMP	20.00	0.6135	(0.5308, 0.6963)	0.0729	(0.0384, 0.1075)
CD14 Mono	CoMP	0.25	0.6868	(0.6336, 0.7400)	0.0060	(-0.0001, 0.0121)
CD14 Mono	CoMP	0.50	0.3840	(0.3001, 0.4680)	-0.0007	(-0.0017, 0.0003)
CD14 Mono	CoMP	1.00	0.1709	(0.0817, 0.2601)	0.0003	(-0.0015, 0.0022)
CD14 Mono	CoMP	5.00	0.4530	(0.3295, 0.5765)	0.0254	(0.0092, 0.0416)
CD14 Mono	CoMP	10.00	0.7073	(0.6114, 0.8031)	0.0745	(0.0419, 0.1072)
CD14 Mono	CoMP	15.00	0.6991	(0.5946, 0.8036)	0.1240	(0.0795, 0.1685)
CD14 Mono	CoMP	20.00	0.7889	(0.6943, 0.8834)	0.1429	(0.1012, 0.1846)
CD16 Mono	CoMP	0.25	0.0963	(0.0382, 0.1543)	0.0031	(0.0018, 0.0045)
CD16 Mono	CoMP	0.50	0.0589	(0.0216, 0.0962)	0.0003	(-0.0008, 0.0013)
CD16 Mono	CoMP	1.00	0.0947	(0.0184, 0.1711)	0.0005	(-0.0011, 0.0021)
CD16 Mono	CoMP	5.00	0.5897	(0.4934, 0.6859)	0.0421	(0.0248, 0.0593)
CD16 Mono	CoMP	10.00	0.6113	(0.4660, 0.7566)	0.1741	(0.0946, 0.2537)
CD16 Mono	CoMP	15.00	0.5950	(0.4552, 0.7348)	0.2778	(0.1705, 0.3850)
CD16 Mono	CoMP	20.00	0.8420	(0.7381, 0.9460)	0.4705	(0.3782, 0.5628)
CD4 T	CoMP	0.25	0.0642	(0.0543, 0.0742)	-0.0027	(-0.0032, -0.0022)
CD4 T	CoMP	0.50	0.0401	(0.0344, 0.0458)	-0.0027	(-0.0031, -0.0023)
CD4 T	CoMP	1.00	0.0538	(0.0442, 0.0634)	-0.0022	(-0.0028, -0.0016)
CD4 T	CoMP	5.00	0.3631	(0.2966, 0.4296)	0.0036	(0.0024, 0.0048)
CD4 T	CoMP	10.00	0.4058	(0.3020, 0.5096)	0.0075	(0.0039, 0.0111)
CD4 T	CoMP	15.00	0.3287	(0.2466, 0.4108)	0.0080	(0.0043, 0.0117)
CD4 T	CoMP	20.00	0.5127	(0.4140, 0.6115)	0.0561	(0.0192, 0.0929)
CD8 T	CoMP	0.25	0.0081	(0.0051, 0.0111)	-0.0015	(-0.0026, -0.0004)
CD8 T	CoMP	0.50	0.0216	(0.0129, 0.0304)	-0.0021	(-0.0032, -0.0010)
CD8 T	CoMP	1.00	0.0634	(0.0437, 0.0830)	-0.0000	(-0.0008, 0.0008)
CD8 T	CoMP	5.00	0.4287	(0.3508, 0.5067)	0.0266	(0.0167, 0.0365)
CD8 T	CoMP	10.00	0.4289	(0.3251, 0.5326)	0.0115	(0.0073, 0.0156)
CD8 T	CoMP	15.00	0.2733	(0.1927, 0.3540)	0.0071	(0.0051, 0.0090)
CD8 T	CoMP	20.00	0.4913	(0.3731, 0.6095)	0.0495	(0.0127, 0.0863)
DC	CoMP	0.25	0.1379	(0.0972, 0.1786)	0.0056	(0.0026, 0.0085)
DC	CoMP	0.50	0.0739	(0.0394, 0.1085)	0.0009	(-0.0029, 0.0047)
DC	CoMP	1.00	0.0784	(0.0329, 0.1239)	0.0034	(-0.0027, 0.0096)
DC	CoMP	5.00	0.2339	(0.1546, 0.3132)	0.0461	(0.0230, 0.0693)
DC	CoMP	10.00	0.4642	(0.3143, 0.6141)	0.1064	(0.0433, 0.1695)
DC	CoMP	15.00	0.6008	(0.4650, 0.7367)	0.1502	(0.0832, 0.2172)
DC	CoMP	20.00	0.7962	(0.6907, 0.9017)	0.1718	(0.1117, 0.2319)
NK	CoMP	0.25	0.0158	(0.0075, 0.0241)	-0.0043	(-0.0049, -0.0036)
NK	CoMP	0.50	0.0233	(0.0045, 0.0420)	-0.0048	(-0.0056, -0.0041)
NK	CoMP	1.00	0.0721	(0.0488, 0.0953)	-0.0025	(-0.0035, -0.0015)
NK	CoMP	5.00	0.6378	(0.5259, 0.7497)	0.0058	(0.0018, 0.0097)
NK	CoMP	10.00	0.5422	(0.4403, 0.6440)	0.0135	(0.0094, 0.0177)
NK	CoMP	15.00	0.4105	(0.3194, 0.5016)	0.0134	(0.0083, 0.0185)
NK	CoMP	20.00	0.5637	(0.4606, 0.6667)	0.0372	(0.0152, 0.0592)
T	CoMP	0.25	0.0052	(0.0025, 0.0079)	-0.0066	(-0.0076, -0.0057)
T	CoMP	0.50	0.0054	(0.0021, 0.0086)	-0.0059	(-0.0066, -0.0052)
T	CoMP	1.00	0.0009	(0.0003, 0.0016)	-0.0062	(-0.0074, -0.0050)
T	CoMP	5.00	0.3430	(0.2440, 0.4420)	0.0087	(0.0041, 0.0134)
T	CoMP	10.00	0.4024	(0.3032, 0.5015)	0.0113	(0.0067, 0.0159)
T	CoMP	15.00	0.2605	(0.1846, 0.3364)	0.0030	(0.0005, 0.0055)
T	CoMP	20.00	0.4979	(0.3883, 0.6076)	0.0542	(0.0224, 0.0860)

Table 10. Counterfactual reconstruction by cell type: Pearson correlation coefficient metrics for all genes (r_{all}) and the top 50 DE genes (r_{DE}). Metrics represent the mean across 10 random seeds for each model. SEM represents standard error of the mean.

Cell type	Model	r_{all}	$r_{\text{all}} \pm$ SEM	r_{DE}	$r_{\text{DE}} \pm$ SEM
B	VAE	0.8854	(0.8850, 0.8857)	0.8170	(0.8165, 0.8175)
B	CVAE	0.9499	(0.9481, 0.9516)	0.9153	(0.9125, 0.9181)
B	VFAE	0.9908	(0.9901, 0.9915)	0.9880	(0.9866, 0.9893)
B	trVAE	0.9877	(0.9868, 0.9886)	0.9833	(0.9817, 0.9849)
B	CoMP	0.9986	(0.9984, 0.9988)	0.9985	(0.9982, 0.9987)
CD14 Mono	VAE	0.7488	(0.7485, 0.7491)	0.4896	(0.4891, 0.4900)
CD14 Mono	CVAE	0.7529	(0.7520, 0.7538)	0.4958	(0.4938, 0.4977)
CD14 Mono	VFAE	0.9954	(0.9951, 0.9958)	0.9928	(0.9921, 0.9935)
CD14 Mono	trVAE	0.9830	(0.9804, 0.9856)	0.9650	(0.9586, 0.9714)
CD14 Mono	CoMP	0.9954	(0.9915, 0.9992)	0.9920	(0.9848, 0.9993)
CD16 Mono	VAE	0.8304	(0.8301, 0.8307)	0.7135	(0.7131, 0.7140)
CD16 Mono	CVAE	0.8351	(0.8341, 0.8360)	0.7223	(0.7203, 0.7243)
CD16 Mono	VFAE	0.9912	(0.9909, 0.9915)	0.9910	(0.9904, 0.9916)
CD16 Mono	trVAE	0.9881	(0.9873, 0.9889)	0.9821	(0.9802, 0.9839)
CD16 Mono	CoMP	0.9990	(0.9985, 0.9994)	0.9989	(0.9986, 0.9993)
CD4 T	VAE	0.8975	(0.8971, 0.8978)	0.8366	(0.8360, 0.8372)
CD4 T	CVAE	0.9697	(0.9682, 0.9712)	0.9514	(0.9492, 0.9537)
CD4 T	VFAE	0.9977	(0.9975, 0.9979)	0.9983	(0.9982, 0.9985)
CD4 T	trVAE	0.9915	(0.9908, 0.9922)	0.9905	(0.9893, 0.9918)
CD4 T	CoMP	0.9990	(0.9990, 0.9991)	0.9988	(0.9987, 0.9989)
CD8 T	VAE	0.9108	(0.9104, 0.9112)	0.8719	(0.8713, 0.8724)
CD8 T	CVAE	0.9726	(0.9715, 0.9736)	0.9613	(0.9598, 0.9628)
CD8 T	VFAE	0.9923	(0.9920, 0.9927)	0.9935	(0.9931, 0.9939)
CD8 T	trVAE	0.9828	(0.9810, 0.9846)	0.9808	(0.9781, 0.9836)
CD8 T	CoMP	0.9927	(0.9917, 0.9937)	0.9950	(0.9945, 0.9955)
DC	VAE	0.8156	(0.8153, 0.8159)	0.5809	(0.5802, 0.5816)
DC	CVAE	0.8213	(0.8205, 0.8221)	0.5943	(0.5925, 0.5961)
DC	VFAE	0.9885	(0.9879, 0.9892)	0.9894	(0.9887, 0.9901)
DC	trVAE	0.9743	(0.9702, 0.9783)	0.9502	(0.9396, 0.9608)
DC	CoMP	0.9946	(0.9925, 0.9966)	0.9931	(0.9899, 0.9962)
NK	VAE	0.8918	(0.8910, 0.8926)	0.8304	(0.8292, 0.8316)
NK	CVAE	0.9539	(0.9520, 0.9558)	0.9269	(0.9237, 0.9301)
NK	VFAE	0.9870	(0.9865, 0.9874)	0.9864	(0.9855, 0.9873)
NK	trVAE	0.9393	(0.9259, 0.9526)	0.9290	(0.9113, 0.9466)
NK	CoMP	0.9917	(0.9904, 0.9929)	0.9899	(0.9881, 0.9917)
T	VAE	0.8848	(0.8843, 0.8853)	0.7469	(0.7457, 0.7480)
T	CVAE	0.9516	(0.9500, 0.9533)	0.8960	(0.8926, 0.8994)
T	VFAE	0.9849	(0.9841, 0.9856)	0.9763	(0.9750, 0.9777)
T	trVAE	0.9567	(0.9498, 0.9637)	0.9368	(0.9294, 0.9443)
T	CoMP	0.9941	(0.9936, 0.9946)	0.9934	(0.9928, 0.9940)

Table 11. Counterfactual reconstruction by cell type: Mean squared error metrics for all genes (MSE_{all}) and the top 50 DE genes (MSE_{DE}). Metrics represent the mean across 10 random seeds for each model. SEM represents standard error of the mean.

Cell type	Model	MSE_{all}	$\text{MSE}_{\text{all}} \pm \text{SEM}$	MSE_{DE}	$\text{MSE}_{\text{DE}} \pm \text{SEM}$
B	VAE	0.0085	(0.0085, 0.0085)	0.3230	(0.3221, 0.3239)
B	CVAE	0.0038	(0.0037, 0.0039)	0.1398	(0.1347, 0.1448)
B	VFAE	0.0008	(0.0007, 0.0008)	0.0199	(0.0178, 0.0220)
B	trVAE	0.0010	(0.0009, 0.0010)	0.0276	(0.0250, 0.0301)
B	CoMP	0.0001	(0.0001, 0.0001)	0.0024	(0.0020, 0.0028)
CD14 Mono	VAE	0.0483	(0.0483, 0.0484)	1.7942	(1.7923, 1.7962)
CD14 Mono	CVAE	0.0476	(0.0475, 0.0478)	1.7624	(1.7563, 1.7684)
CD14 Mono	VFAE	0.0014	(0.0013, 0.0015)	0.0343	(0.0314, 0.0371)
CD14 Mono	trVAE	0.0044	(0.0038, 0.0051)	0.1422	(0.1190, 0.1654)
CD14 Mono	CoMP	0.0011	(0.0002, 0.0019)	0.0245	(0.0023, 0.0468)
CD16 Mono	VAE	0.0301	(0.0301, 0.0302)	1.1255	(1.1234, 1.1276)
CD16 Mono	CVAE	0.0294	(0.0293, 0.0295)	1.0933	(1.0878, 1.0989)
CD16 Mono	VFAE	0.0017	(0.0017, 0.0018)	0.0223	(0.0204, 0.0242)
CD16 Mono	trVAE	0.0029	(0.0027, 0.0031)	0.0861	(0.0790, 0.0932)
CD16 Mono	CoMP	0.0002	(0.0001, 0.0003)	0.0031	(0.0017, 0.0044)
CD4 T	VAE	0.0060	(0.0059, 0.0060)	0.2274	(0.2266, 0.2282)
CD4 T	CVAE	0.0018	(0.0017, 0.0019)	0.0677	(0.0644, 0.0709)
CD4 T	VFAE	0.0001	(0.0001, 0.0002)	0.0021	(0.0018, 0.0023)
CD4 T	trVAE	0.0005	(0.0005, 0.0006)	0.0126	(0.0107, 0.0144)
CD4 T	CoMP	0.0001	(0.0001, 0.0001)	0.0015	(0.0014, 0.0016)
CD8 T	VAE	0.0058	(0.0058, 0.0058)	0.2187	(0.2178, 0.2196)
CD8 T	CVAE	0.0019	(0.0019, 0.0020)	0.0684	(0.0659, 0.0709)
CD8 T	VFAE	0.0005	(0.0005, 0.0006)	0.0098	(0.0093, 0.0103)
CD8 T	trVAE	0.0012	(0.0011, 0.0013)	0.0332	(0.0284, 0.0381)
CD8 T	CoMP	0.0005	(0.0004, 0.0006)	0.0074	(0.0066, 0.0082)
DC	VAE	0.0332	(0.0331, 0.0332)	1.2308	(1.2292, 1.2324)
DC	CVAE	0.0322	(0.0321, 0.0324)	1.1887	(1.1834, 1.1939)
DC	VFAE	0.0024	(0.0023, 0.0025)	0.0303	(0.0287, 0.0318)
DC	trVAE	0.0056	(0.0048, 0.0064)	0.1758	(0.1436, 0.2081)
DC	CoMP	0.0011	(0.0007, 0.0016)	0.0161	(0.0083, 0.0239)
NK	VAE	0.0091	(0.0091, 0.0092)	0.3395	(0.3370, 0.3420)
NK	CVAE	0.0043	(0.0041, 0.0044)	0.1535	(0.1477, 0.1593)
NK	VFAE	0.0014	(0.0013, 0.0014)	0.0345	(0.0327, 0.0362)
NK	trVAE	0.0053	(0.0042, 0.0064)	0.1455	(0.1100, 0.1810)
NK	CoMP	0.0008	(0.0007, 0.0010)	0.0204	(0.0169, 0.0238)
T	VAE	0.0077	(0.0077, 0.0077)	0.2799	(0.2786, 0.2811)
T	CVAE	0.0033	(0.0032, 0.0034)	0.1126	(0.1088, 0.1164)
T	VFAE	0.0011	(0.0010, 0.0011)	0.0237	(0.0223, 0.0252)
T	trVAE	0.0030	(0.0025, 0.0034)	0.0674	(0.0583, 0.0764)
T	CoMP	0.0004	(0.0004, 0.0005)	0.0066	(0.0060, 0.0073)

Table 12. Effect of γ on counterfactual data reconstruction: Mean Pearson correlation coefficient for all and DE genes across 10 random seeds.

Cell type	Model	γ	r_{all}	$r_{\text{all}} \pm$ SEM	r_{DE}	$r_{\text{DE}} \pm$ SEM
B	CoMP	0.25	0.9987	(0.9986, 0.9988)	0.9986	(0.9984, 0.9987)
B	CoMP	0.50	0.9987	(0.9985, 0.9988)	0.9985	(0.9983, 0.9988)
B	CoMP	1.00	0.9986	(0.9984, 0.9988)	0.9985	(0.9982, 0.9987)
B	CoMP	5.00	0.9577	(0.9432, 0.9722)	0.9623	(0.9510, 0.9736)
B	CoMP	10.00	0.9233	(0.9023, 0.9443)	0.9397	(0.9239, 0.9555)
B	CoMP	15.00	0.9106	(0.8925, 0.9287)	0.9299	(0.9163, 0.9435)
B	CoMP	20.00	0.8974	(0.8841, 0.9107)	0.9080	(0.8966, 0.9195)
CD14 Mono	CoMP	0.25	0.9906	(0.9866, 0.9945)	0.9828	(0.9746, 0.9909)
CD14 Mono	CoMP	0.50	0.9948	(0.9917, 0.9980)	0.9910	(0.9844, 0.9977)
CD14 Mono	CoMP	1.00	0.9954	(0.9915, 0.9992)	0.9920	(0.9848, 0.9993)
CD14 Mono	CoMP	5.00	0.9892	(0.9831, 0.9954)	0.9823	(0.9723, 0.9922)
CD14 Mono	CoMP	10.00	0.9536	(0.9316, 0.9757)	0.9439	(0.9217, 0.9662)
CD14 Mono	CoMP	15.00	0.9224	(0.8933, 0.9515)	0.9174	(0.8886, 0.9463)
CD14 Mono	CoMP	20.00	0.9034	(0.8737, 0.9332)	0.8966	(0.8673, 0.9258)
CD16 Mono	CoMP	0.25	0.9983	(0.9977, 0.9989)	0.9982	(0.9976, 0.9988)
CD16 Mono	CoMP	0.50	0.9989	(0.9985, 0.9992)	0.9988	(0.9985, 0.9991)
CD16 Mono	CoMP	1.00	0.9990	(0.9985, 0.9994)	0.9989	(0.9986, 0.9993)
CD16 Mono	CoMP	5.00	0.9847	(0.9805, 0.9890)	0.9801	(0.9753, 0.9850)
CD16 Mono	CoMP	10.00	0.9420	(0.9168, 0.9672)	0.9503	(0.9313, 0.9693)
CD16 Mono	CoMP	15.00	0.9017	(0.8702, 0.9332)	0.9222	(0.8999, 0.9444)
CD16 Mono	CoMP	20.00	0.8563	(0.8289, 0.8838)	0.8850	(0.8668, 0.9031)
CD4 T	CoMP	0.25	0.9989	(0.9989, 0.9990)	0.9987	(0.9986, 0.9988)
CD4 T	CoMP	0.50	0.9991	(0.9990, 0.9991)	0.9989	(0.9988, 0.9990)
CD4 T	CoMP	1.00	0.9990	(0.9990, 0.9991)	0.9988	(0.9987, 0.9989)
CD4 T	CoMP	5.00	0.9925	(0.9899, 0.9951)	0.9901	(0.9863, 0.9939)
CD4 T	CoMP	10.00	0.9948	(0.9933, 0.9962)	0.9970	(0.9954, 0.9985)
CD4 T	CoMP	15.00	0.9944	(0.9931, 0.9958)	0.9979	(0.9975, 0.9983)
CD4 T	CoMP	20.00	0.9782	(0.9689, 0.9875)	0.9738	(0.9584, 0.9892)
CD8 T	CoMP	0.25	0.9963	(0.9961, 0.9964)	0.9965	(0.9964, 0.9966)
CD8 T	CoMP	0.50	0.9955	(0.9951, 0.9960)	0.9962	(0.9959, 0.9965)
CD8 T	CoMP	1.00	0.9927	(0.9917, 0.9937)	0.9950	(0.9945, 0.9955)
CD8 T	CoMP	5.00	0.9666	(0.9626, 0.9705)	0.9790	(0.9765, 0.9814)
CD8 T	CoMP	10.00	0.9559	(0.9499, 0.9620)	0.9757	(0.9727, 0.9787)
CD8 T	CoMP	15.00	0.9528	(0.9468, 0.9589)	0.9745	(0.9715, 0.9774)
CD8 T	CoMP	20.00	0.9455	(0.9397, 0.9512)	0.9605	(0.9516, 0.9694)
DC	CoMP	0.25	0.9959	(0.9955, 0.9962)	0.9945	(0.9942, 0.9949)
DC	CoMP	0.50	0.9966	(0.9963, 0.9970)	0.9956	(0.9949, 0.9962)
DC	CoMP	1.00	0.9946	(0.9925, 0.9966)	0.9931	(0.9899, 0.9962)
DC	CoMP	5.00	0.9694	(0.9576, 0.9811)	0.9671	(0.9528, 0.9814)
DC	CoMP	10.00	0.9219	(0.8966, 0.9472)	0.9265	(0.9031, 0.9499)
DC	CoMP	15.00	0.8867	(0.8549, 0.9184)	0.8955	(0.8686, 0.9224)
DC	CoMP	20.00	0.8547	(0.8288, 0.8806)	0.8676	(0.8428, 0.8924)
NK	CoMP	0.25	0.9962	(0.9959, 0.9964)	0.9955	(0.9951, 0.9959)
NK	CoMP	0.50	0.9949	(0.9942, 0.9957)	0.9938	(0.9927, 0.9950)
NK	CoMP	1.00	0.9917	(0.9904, 0.9929)	0.9899	(0.9881, 0.9917)
NK	CoMP	5.00	0.9567	(0.9491, 0.9643)	0.9399	(0.9296, 0.9501)
NK	CoMP	10.00	0.8916	(0.8654, 0.9178)	0.8670	(0.8361, 0.8979)
NK	CoMP	15.00	0.8780	(0.8501, 0.9060)	0.8518	(0.8187, 0.8850)
NK	CoMP	20.00	0.8767	(0.8517, 0.9018)	0.8477	(0.8189, 0.8765)
T	CoMP	0.25	0.9951	(0.9947, 0.9954)	0.9945	(0.9940, 0.9950)
T	CoMP	0.50	0.9950	(0.9947, 0.9952)	0.9945	(0.9940, 0.9949)
T	CoMP	1.00	0.9941	(0.9936, 0.9946)	0.9934	(0.9928, 0.9940)
T	CoMP	5.00	0.9682	(0.9584, 0.9779)	0.9690	(0.9627, 0.9753)
T	CoMP	10.00	0.9462	(0.9336, 0.9588)	0.9595	(0.9519, 0.9672)
T	CoMP	15.00	0.9402	(0.9277, 0.9527)	0.9569	(0.9495, 0.9642)
T	CoMP	20.00	0.9239	(0.9136, 0.9342)	0.9254	(0.9092, 0.9416)

Table 13. Effect of γ on counterfactual data reconstruction: Mean of the mean squared error (MSE) for all and DE genes across 10 random seeds.

Cell type	Model	γ	MSE_{all}	$\text{MSE}_{\text{all}} \pm \text{SEM}$	MSE_{DE}	$\text{MSE}_{\text{DE}} \pm \text{SEM}$
B	CoMP	0.25	0.0001	(0.0001, 0.0001)	0.0023	(0.0021, 0.0025)
B	CoMP	0.50	0.0001	(0.0001, 0.0001)	0.0023	(0.0019, 0.0026)
B	CoMP	1.00	0.0001	(0.0001, 0.0001)	0.0024	(0.0020, 0.0028)
B	CoMP	5.00	0.0032	(0.0022, 0.0043)	0.0672	(0.0479, 0.0864)
B	CoMP	10.00	0.0056	(0.0041, 0.0071)	0.1046	(0.0768, 0.1325)
B	CoMP	15.00	0.0065	(0.0053, 0.0078)	0.1201	(0.0960, 0.1442)
B	CoMP	20.00	0.0077	(0.0067, 0.0086)	0.1666	(0.1442, 0.1891)
CD14 Mono	CoMP	0.25	0.0023	(0.0014, 0.0032)	0.0576	(0.0309, 0.0843)
CD14 Mono	CoMP	0.50	0.0012	(0.0005, 0.0020)	0.0276	(0.0068, 0.0483)
CD14 Mono	CoMP	1.00	0.0011	(0.0002, 0.0019)	0.0245	(0.0023, 0.0468)
CD14 Mono	CoMP	5.00	0.0034	(0.0013, 0.0056)	0.0935	(0.0330, 0.1541)
CD14 Mono	CoMP	10.00	0.0123	(0.0066, 0.0180)	0.3157	(0.1724, 0.4591)
CD14 Mono	CoMP	15.00	0.0196	(0.0124, 0.0268)	0.4825	(0.3056, 0.6594)
CD14 Mono	CoMP	20.00	0.0245	(0.0173, 0.0316)	0.6022	(0.4302, 0.7742)
CD16 Mono	CoMP	0.25	0.0003	(0.0002, 0.0005)	0.0054	(0.0034, 0.0074)
CD16 Mono	CoMP	0.50	0.0002	(0.0001, 0.0003)	0.0036	(0.0024, 0.0048)
CD16 Mono	CoMP	1.00	0.0002	(0.0001, 0.0003)	0.0031	(0.0017, 0.0044)
CD16 Mono	CoMP	5.00	0.0040	(0.0024, 0.0056)	0.0876	(0.0451, 0.1300)
CD16 Mono	CoMP	10.00	0.0120	(0.0070, 0.0170)	0.2751	(0.1522, 0.3979)
CD16 Mono	CoMP	15.00	0.0194	(0.0134, 0.0254)	0.4529	(0.3058, 0.6001)
CD16 Mono	CoMP	20.00	0.0284	(0.0233, 0.0335)	0.6688	(0.5421, 0.7955)
CD4 T	CoMP	0.25	0.0001	(0.0001, 0.0001)	0.0015	(0.0014, 0.0017)
CD4 T	CoMP	0.50	0.0001	(0.0001, 0.0001)	0.0013	(0.0013, 0.0014)
CD4 T	CoMP	1.00	0.0001	(0.0001, 0.0001)	0.0015	(0.0014, 0.0016)
CD4 T	CoMP	5.00	0.0005	(0.0003, 0.0006)	0.0133	(0.0081, 0.0185)
CD4 T	CoMP	10.00	0.0003	(0.0002, 0.0004)	0.0040	(0.0019, 0.0062)
CD4 T	CoMP	15.00	0.0003	(0.0003, 0.0004)	0.0027	(0.0022, 0.0032)
CD4 T	CoMP	20.00	0.0013	(0.0008, 0.0019)	0.0381	(0.0155, 0.0607)
CD8 T	CoMP	0.25	0.0003	(0.0003, 0.0003)	0.0049	(0.0046, 0.0051)
CD8 T	CoMP	0.50	0.0003	(0.0003, 0.0003)	0.0052	(0.0048, 0.0056)
CD8 T	CoMP	1.00	0.0005	(0.0004, 0.0006)	0.0074	(0.0066, 0.0082)
CD8 T	CoMP	5.00	0.0023	(0.0020, 0.0025)	0.0329	(0.0290, 0.0369)
CD8 T	CoMP	10.00	0.0030	(0.0026, 0.0033)	0.0366	(0.0322, 0.0411)
CD8 T	CoMP	15.00	0.0032	(0.0028, 0.0035)	0.0387	(0.0344, 0.0431)
CD8 T	CoMP	20.00	0.0038	(0.0034, 0.0042)	0.0717	(0.0512, 0.0921)
DC	CoMP	0.25	0.0009	(0.0008, 0.0009)	0.0135	(0.0122, 0.0149)
DC	CoMP	0.50	0.0007	(0.0006, 0.0008)	0.0102	(0.0082, 0.0123)
DC	CoMP	1.00	0.0011	(0.0007, 0.0016)	0.0161	(0.0083, 0.0239)
DC	CoMP	5.00	0.0075	(0.0044, 0.0106)	0.1189	(0.0580, 0.1798)
DC	CoMP	10.00	0.0165	(0.0113, 0.0217)	0.2549	(0.1607, 0.3490)
DC	CoMP	15.00	0.0228	(0.0169, 0.0288)	0.3663	(0.2603, 0.4723)
DC	CoMP	20.00	0.0299	(0.0250, 0.0347)	0.4809	(0.3844, 0.5773)
NK	CoMP	0.25	0.0004	(0.0004, 0.0004)	0.0089	(0.0080, 0.0097)
NK	CoMP	0.50	0.0005	(0.0004, 0.0006)	0.0122	(0.0099, 0.0146)
NK	CoMP	1.00	0.0008	(0.0007, 0.0010)	0.0204	(0.0169, 0.0238)
NK	CoMP	5.00	0.0041	(0.0035, 0.0048)	0.1169	(0.0988, 0.1350)
NK	CoMP	10.00	0.0090	(0.0069, 0.0110)	0.2419	(0.1869, 0.2969)
NK	CoMP	15.00	0.0100	(0.0078, 0.0122)	0.2687	(0.2101, 0.3273)
NK	CoMP	20.00	0.0103	(0.0084, 0.0122)	0.2874	(0.2376, 0.3371)
T	CoMP	0.25	0.0004	(0.0003, 0.0004)	0.0053	(0.0048, 0.0058)
T	CoMP	0.50	0.0004	(0.0003, 0.0004)	0.0053	(0.0049, 0.0058)
T	CoMP	1.00	0.0004	(0.0004, 0.0005)	0.0066	(0.0060, 0.0073)
T	CoMP	5.00	0.0022	(0.0016, 0.0029)	0.0409	(0.0311, 0.0507)
T	CoMP	10.00	0.0037	(0.0029, 0.0046)	0.0579	(0.0454, 0.0703)
T	CoMP	15.00	0.0041	(0.0033, 0.0050)	0.0624	(0.0501, 0.0747)
T	CoMP	20.00	0.0053	(0.0046, 0.0060)	0.1005	(0.0829, 0.1181)

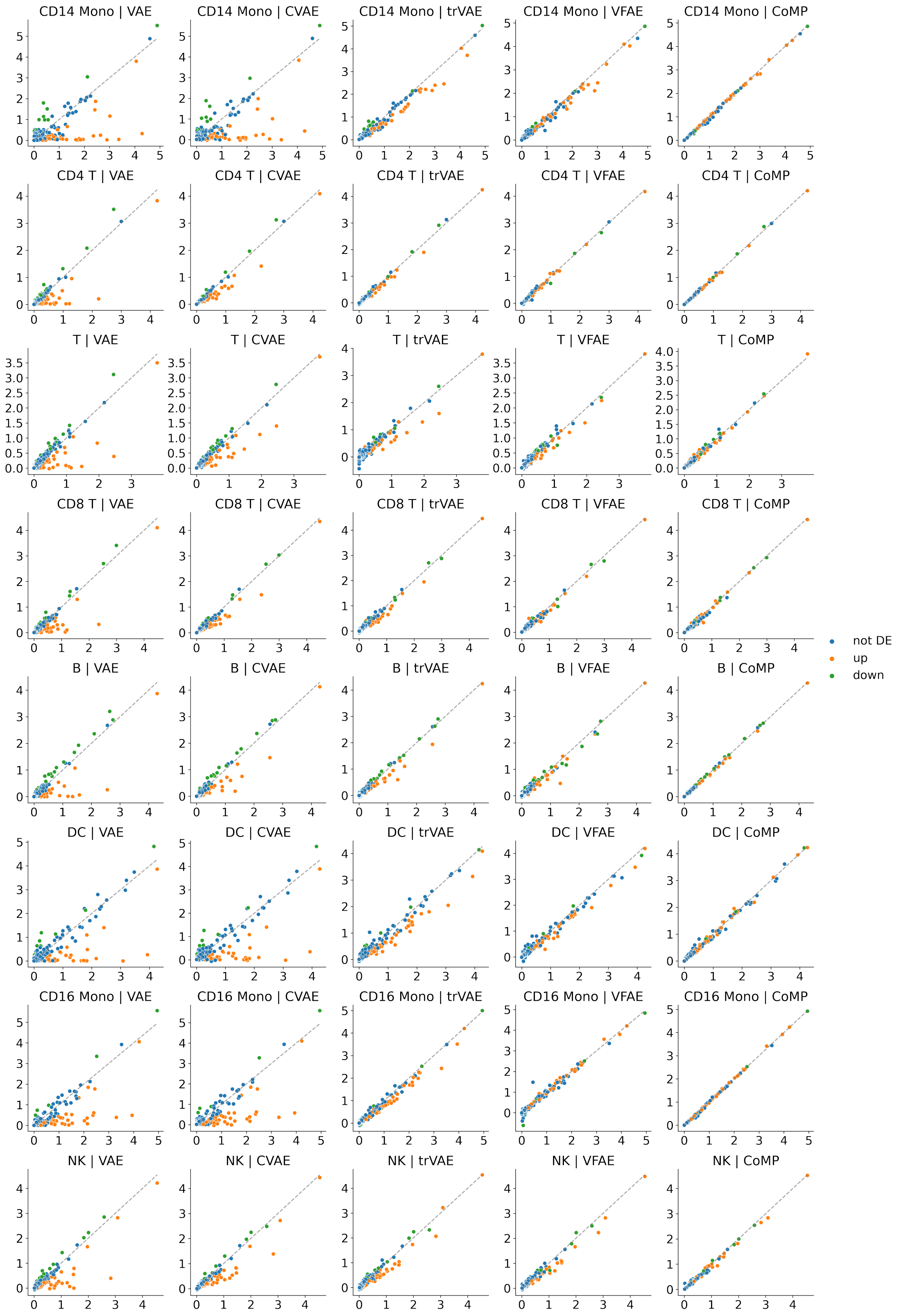

Figure 7. Mean gene expression of actual stimulated cells against the mean gene expression of counterfactually stimulated control cells for each cell type and model.

Table 14. scRNA-seq data integration experiment results, with $k = 100$, $c =$ Protocol, and $\alpha = 0.05$. $s_{k,c}$ and $\tilde{s}_{k,c}$ are the two Silhouette Coefficient variants (see Section 6). The top scores are in **bold**. For CoMP, results represent the mean across 19 random initialisations ± standard error of the mean. $\tilde{s}$ and m-kBET represent mean metrics computed over considered cell sub-types (see F.6 for details of considered cell sub-types).

	s	kBET	$\tilde{s}$	m-kBET
Seurat CCA	0.0176	0.436	0.022	0.356
Harmony	0.0158	0.318	0.013	0.245
CoMP	$\mathbf{0.0007 \pm 0.002}$	$\mathbf{0.171 \pm 0.003}$	$\mathbf{0.0029 \pm 0.0004}$	$\mathbf{0.132 \pm 0.002}$

F.6. Batch mixing metrics for scRNA-seq data integration

In this section we present additional details of our experiments for the scRNA-seq data integration experiment, evaluating the mixing of cells from two batches (different library preparation protocols) in the latent space for CoMP over 19 random seeds, compared with Seurat (CCA) (Stuart et al., 2019) and Harmony (Korsunsky et al., 2019). The following cell sub-types were considered in this analysis: naive B cells, memory B cells, CD14 monocytes, CD16 monocytes, naive CD4+ T cells, effector CD8+ T cells, naive CD8+ T cells, memory CD8+ T cells, regulatory T cells, activated dendritic cells, plasmacytoid dendritic cells, and natural killer cells. Megakaryocytes and Hematopoietic stem cells were excluded from the analysis due to the very low numbers of cells within the dataset, which made computing the metrics for these cell sub-types infeasible. Note that cell sub-types were predefined and provided in the associated metadata from Korsunsky et al. (2019). In Table 14, we present the results of CoMP across a number of training seeds, showing that it has reliable performance against baselines with minimal standard error.

F.7. Implementation details and hyperparameters

The encoder and decoders are parameterised by multi-layer fully-connected networks. Following the trVAE implementation (Lotfollahi et al., 2019a), we implement a multi-scale Gaussian kernel for both trVAE and VFAE benchmark models, except on the Adult Income dataset where a single scale kernel was used to match the original implementation. The details of the model architectures and hyperparameters used in CoMP, VFAE and trVAE across three sets of experiments are given in Tables 15–24. The networks in all experiments were trained with a 90/10 train/validation split, metrics were calculated on the entire dataset.

We use the means of the posteriors as the encoded representation $\mathbf{z}$ for each sample. For the Adult Income experiments this differs from Louizos et al. (2015), where the representations are sampled from the posterior before classification. We found the noise from sampling would mask the inclusion of predictive information about gender in the encoded means from the VFAE, as can be seen in the difference in accuracy between [VFAE-s] and [VFAE-m] in Table 3.

F.8. Model training resources

Experiments were performed on NVIDIA Tesla V100 GPUs. Each training run of CoMP for a single hyperparameter configuration on the Tumour / Cell Line dataset (our largest dataset) on a single GPU takes 2–3 hours. Running times for the other models are broadly similar.

F.9. CO_2 emissions

Experiments were conducted using private infrastructure, which has an estimated carbon efficiency of 0.188 $kgCO_2eq/kWh$. An estimated cumulative 1900 hours of computation were performed on hardware of type Tesla V100. Total emissions are estimated to be 107 $kgCO_2eq$. Estimations were conducted using the Machine Learning Impact calculator presented in Lacoste et al. (2019).

Table 15. CoMP architecture and hyperparameters for the Tumour / Cell Line dataset.

Layer	Output Dim	Inputs	Notes
Input	9468		
Conditions	2		
Encoder			
FC_1	512	[Input, Conditions]	BatchNorm1D, LeakyReLU
FC_2	512	FC_1	BatchNorm1D, LeakyReLU
FC_3	512	FC_2	BatchNorm1D, LeakyReLU
Z_mean	16	FC_3	
Z	16	[Z_mean, 0.1]	Normal()
Decoder			
FC_1	512	Z	BatchNorm1D, LeakyReLU
FC-2	512	FC_1	BatchNorm1D, LeakyReLU
FC-3	512	FC-2	BatchNorm1D, LeakyReLU
$\hat{X}$_mean	9468	FC-3	
$\hat{X}$_scale	1	FC-3	
$\hat{X}$	9468	[$\hat{X}$_mean, $\hat{X}$_scale]	Normal()
Penalty			
CoMP penalty		[Z, Conditions]	
Optimiser	Adam		
Learning rate	1e-4		
Batch size	5500		
Epochs	4000		
β	1e-7		
γ	0.5		
LeakyReLU slope	0.01		

Table 16. VFAE architecture and hyperparameters for the Tumour / Cell Line dataset.

Layer	Output Dim	Inputs	Notes
Input	9468		
Conditions	2		
Encoder			
FC_1	512	[Input, Conditions]	BatchNorm1D, LeakyReLU
FC_2	512	FC_1	BatchNorm1D, LeakyReLU
FC_3	512	FC_2	BatchNorm1D, LeakyReLU
Z_mean	16	FC_3	
Z_scale	16	FC_3	
Z	16	[Z_mean, Z_scale]	Normal()
Decoder			
FC_1	512	Z	BatchNorm1D, LeakyReLU
FC-2	512	FC_1	BatchNorm1D, LeakyReLU
FC-3	512	FC-2	BatchNorm1D, LeakyReLU
X̂_mean	9468	FC-3	
X̂_scale	1	FC-3	
X̂	9468	[X̂_mean, X̂_scale]	Normal()
Penalty			
MMD		[FC1, Conditions]	Multi-scale RBF kernel
Optimiser	Adam		
Learning rate	1e-03		
Batch size	5550		
Epochs	4000		
β	1e-7		
γ	4		
LeakyReLU slope	0.01		

Table 17. trVAE architecture and hyperparameters for the Tumour / Cell Line dataset.

Layer	Output Dim	Inputs	Notes
Input	9468		
Conditions	2		
Encoder			
FC_1	512	[Input, Conditions]	BatchNorm1D, LeakyReLU
FC_2	512	FC_1	BatchNorm1D, LeakyReLU
FC_3	512	FC_2	BatchNorm1D, LeakyReLU
Z_mean	16	FC_3	
Z_scale	16	FC_3	
Z	16	[Z_mean, Z_scale]	Normal()
Decoder			
FC_1	512	Z	BatchNorm1D, LeakyReLU
FC-2	512	FC_1	BatchNorm1D, LeakyReLU
FC-3	512	FC-2	BatchNorm1D, LeakyReLU
$\hat{X}$_mean	9468	FC-3	
$\hat{X}$_scale	1	FC-3	
$\hat{X}$	9468	[$\hat{X}$_mean, $\hat{X}$_scale]	Normal()
Penalty			
MMD		[FC1, Conditions]	Multi-scale RBF kernel
Optimiser	Adam		
Learning rate	3e-4		
Batch size	5550		
Epochs	4000		
β	1e-7		
γ	10		
LeakyReLU slope	0.01		

Table 18. CoMP architecture and hyperparameters for the stimulated / untreated single-cell PBMCs dataset.

Layer	Output Dim	Inputs	Notes
Input	2000		
Conditions	2		
Encoder			
FC_1	512	[Input, Conditions]	BatchNorm1D, LeakyReLU
FC_2	512	FC_1	BatchNorm1D, LeakyReLU
FC_3	512	FC_2	BatchNorm1D, LeakyReLU
Z_mean	40	FC_3	
Z	40	[Z_mean, 0.1]	Normal()
Decoder			
FC_1	512	Z	BatchNorm1D, LeakyReLU
FC-2	512	FC_1	BatchNorm1D, LeakyReLU
FC-3	512	FC-2	BatchNorm1D, LeakyReLU
X̂_mean	2000	FC-3	
X̂_scale	1	FC-3	
X̂	2000	[X̂_mean, X̂_scale]	Normal()
Penalty			
CoMP penalty		[Z, Conditions]	
Optimiser	Adam		
Learning rate	1e-06		
Batch size	512		
Epochs	10000		
β	1e-7		
γ	1		
LeakyReLU slope	0.01		

Table 19. VFAE architecture and hyperparameters for the stimulated / untreated single-cell PBMCs dataset.

Layer	Output Dim	Inputs	Notes
Input	2000		
Conditions	2		
Encoder			
FC_1	512	[Input, Conditions]	BatchNorm1D, LeakyReLU
FC_2	512	FC_1	BatchNorm1D, LeakyReLU
FC_3	512	FC_2	BatchNorm1D, LeakyReLU
Z_mean	40	FC_3	
Z	40	[Z_mean, 0.1]	Normal()
Decoder			
FC_1	512	Z	BatchNorm1D, LeakyReLU
FC-2	512	FC_1	BatchNorm1D, LeakyReLU
FC-3	512	FC-2	BatchNorm1D, LeakyReLU
$\hat{X}$_mean	2000	FC-3	
$\hat{X}$_scale	1	FC-3	
$\hat{X}$	2000	[$\hat{X}$_mean, $\hat{X}$_scale]	Normal()
Penalty			
MMD		[FC1, Conditions]	Multi-scale RBF kernel
Optimiser	Adam		
Learning rate	1e-4		
Batch size	512		
Epochs	10000		
β	1e-7		
γ	1		
LeakyReLU slope	0.01		

Table 20. trVAE architecture and hyperparameters for the single-cell PBMC dataset.

Layer	Output Dim	Inputs	Notes
Input	2000		
Conditions	2		
Encoder			
FC_1	512	[Input, Conditions]	BatchNorm1D, LeakyReLU
FC_2	512	FC_1	BatchNorm1D, LeakyReLU
FC_3	512	FC_2	BatchNorm1D, LeakyReLU
Z_mean	40	FC_3	
Z	40	[Z_mean, 0.1]	Normal()
Decoder			
FC_1	512	Z	BatchNorm1D, LeakyReLU
FC-2	512	FC_1	BatchNorm1D, LeakyReLU
FC-3	512	FC-2	BatchNorm1D, LeakyReLU
$\hat{X}$_mean	2000	FC-3	
$\hat{X}$_scale	1	FC-3	
$\hat{X}$	2000	[$\hat{X}$_mean, $\hat{X}$_scale]	Normal()
Penalty			
MMD		[FC1, Conditions]	Multi-scale RBF kernel
Optimiser	Adam		
Learning rate	5e-4		
Batch size	512		
Epochs	6000		
β	1e-7		
γ	10		
LeakyReLU slope	0.01		

Table 21. CoMP architecture and hyperparameters for the scRNA-seq data integration dataset.

Layer	Output Dim	Inputs	Notes
Input	2000		
Conditions	2		
Encoder			
FC_1	512	[Input, Conditions]	BatchNorm1D, LeakyReLU
FC_2	512	FC_1	BatchNorm1D, LeakyReLU
FC_3	512	FC_2	BatchNorm1D, LeakyReLU
Z_mean	40	FC_3	
Z	40	[Z_mean, 0.1]	Normal()
Decoder			
FC_1	512	Z	BatchNorm1D, LeakyReLU
FC-2	512	FC_1	BatchNorm1D, LeakyReLU
FC-3	512	FC-2	BatchNorm1D, LeakyReLU
$\hat{X}$_mean	2000	FC-3	
$\hat{X}$_scale	1	FC-3	
$\hat{X}$	2000	[$\hat{X}$_mean, $\hat{X}$_scale]	Normal()
Penalty			
CoMP penalty		[Z, Conditions]	
Optimiser	Adam		
Learning rate	5e-06		
Batch size	512		
Epochs	350		
β	1		
γ	0.75		
LeakyReLU slope	0.01		

Table 22. CoMP architecture and hyperparameters for the UCI Adult Income dataset.

Layer	Output Dim	Inputs	Notes
Input	82		
Conditions	2		
Encoder			
FC_1	64	[Input, Conditions]	BatchNorm1D, LeakyReLU
FC_2	64	FC_1	BatchNorm1D, LeakyReLU
Z_mean	16	FC_2	
Z	16	[Z_mean, 0.1]	Normal()
Decoder			
FC_1	64	Z	BatchNorm1D, LeakyReLU
FC-2	64	FC_1	BatchNorm1D, LeakyReLU
$\hat{X}$_mean	82	FC-2	
$\hat{X}$_scale	1	FC-2	
$\hat{X}$	82	[$\hat{X}$_mean, $\hat{X}$_scale]	Normal()
Penalty			
CoMP penalty		[Z, Conditions]	
Optimiser	Adam		
Learning rate	1e-04		
Batch size	4096		
Epochs	10000		
β	1		
γ	0.5		
LeakyReLU slope	0.01		

Table 23. VFAE architecture and hyperparameters for the UCI Adult Income dataset.

Layer	Output Dim	Inputs	Notes
Input	82		
Conditions	2		
Encoder			
FC_1	64	[Input, Conditions]	BatchNorm1D, LeakyReLU
Z_mean	16	FC_1	
Z_scale	16	FC_1	
Z	16	[Z_mean, Z_scale]	Normal()
Decoder			
FC_1	64	Z	BatchNorm1D, LeakyReLU
$\hat{X}$_mean	82	FC_1	
$\hat{X}$_scale	1	FC_1	
$\hat{X}$	82	[$\hat{X}$_mean, $\hat{X}$_scale]	Normal()
Penalty			
MMD		[Z, Conditions]	
Optimiser	Adam		
Learning rate	1e-04		
Batch size	512		
Epochs	10000		
β	1		
γ	1000		
RBF scale	2		
LeakyReLU slope	0.01		

Table 24. trVAE architecture and hyperparameters for the UCI Adult Income dataset.

Layer	Output Dim	Inputs	Notes
Input	82		
Conditions	2		
Encoder			
FC_1	32	[Input, Conditions]	BatchNorm1D, LeakyReLU
FC_2	32	FC_1	BatchNorm1D, LeakyReLU
Z_mean	8	FC_2	
Z	8	[Z_mean, 0.1]	Normal()
Decoder			
FC_1	32	Z	BatchNorm1D, LeakyReLU
FC-2	32	FC_1	BatchNorm1D, LeakyReLU
$\hat{X}$_mean	82	FC-2	
$\hat{X}$_scale	1	FC-2	
$\hat{X}$	82	[$\hat{X}$_mean, $\hat{X}$_scale]	Normal()
Penalty			
MMD		[FC1, Conditions]	Multi-scale RBF kernel
Optimiser	Adam		
Learning rate	1e-04		
Batch size	4096		
Epochs	10000		
β	0.001		
γ	10		
LeakyReLU slope	0.01		