CONDITIONAL OUT-OF-SAMPLE GENERATION FOR UNPAIRED DATA USING TRVAE

Anonymous authors
Paper under double-blind review

ABSTRACT

While generative models have shown great success in generating high-dimensional samples conditional on low-dimensional descriptors (learning e.g. stroke thickness in MNIST, hair color in CelebA, or speaker identity in Wavenet), their generation out-of-sample poses fundamental problems. The conditional variational autoencoder (CVAE) as a simple conditional generative model does not explicitly relate conditions during training and, hence, has no incentive of learning a compact joint distribution across conditions. We overcome this limitation by matching their distributions using maximum mean discrepancy (MMD) in the decoder layer that follows the bottleneck. This introduces a strong regularization both for reconstructing samples within the same condition and for transforming samples across conditions, resulting in much improved generalization. We refer to the architecture as *transformer* VAE (trVAE). Benchmarking trVAE on high-dimensional image and tabular data, we demonstrate higher robustness and higher accuracy than existing approaches. In particular, we show qualitatively improved predictions for cellular perturbation response to treatment and disease based on high-dimensional single-cell gene expression data, by tackling previously problematic minority classes and multiple conditions. For generic tasks, we improve Pearson correlations of high-dimensional estimated means and variances with their ground truths from 0.89 to 0.97 and 0.75 to 0.87, respectively.

1 INTRODUCTION

The task of generating high-dimensional samples x conditional on a latent random vector z and a categorical variable s has established solutions (Mirza & Osindero, 2014; Ren et al., 2016). The situation becomes more complicated if the support of z is divided into different domains d with different semantic meanings: say $d \in \{\text{men}, \text{women}\}$ and one is interested in out-of-sample generation of samples x in a domain and condition (d, s) that is not part of the training data. If one predicts how a given black-haired man would look with blonde hair, which we refer to as *transforming* $x_{\text{men, black-hair}} \mapsto x_{\text{men, blonde-hair}}$, this becomes an out-of-sample problem if the training data does not have instances of blonde-haired men, but merely of blonde- and black-haired woman and blacked haired men. In an application with higher relevance, there is strong interest in how untreated ($s = 0$) humans ($d = 0$) respond to drug treatment ($s = 1$) based on training data from in vitro ($d = 1$) and mice ($d = 2$) experiments. Hence, the target domain of interest ($d = 0$) does not offer training data for $s = 1$, but only for $s = 0$.

In the present paper, we suggest to address the challenge of transforming out-of-sample by regularizing the joint distribution across the categorical variable s using maximum mean discrepancy (MMD) in the framework of a conditional variational autoencoder (CVAE) (Sohn et al., 2015). This produces a more compact representation of a distribution that displays high variance in the vanilla CVAE, which incentivizes learning of features across s and results in more accurate out-of-sample prediction. MMD has proven successful in a variety of tasks. In particular, matching distributions with MMD in variational autoencoders (Kingma & Welling, 2013) has been put forward for unsupervised domain adaptation (Louizos et al., 2015) or for learning statistically independent latent dimensions (Lopez et al., 2018b). In supervised domain adaptation approaches, MMD-based regularization has been shown to be a viable strategy of learning label-predictive features with domain-specific information removed (Long et al., 2015; Tzeng et al., 2014).

In further related work, the out-of-sample transformation problem was addressed via hard-coded latent space vector arithmetics (Lotfollahi et al., 2019) and histogram matching (Amodio et al., 2018). The approach of the present paper, however, introduce a data-driven end-to-end approach, which does not involve hard-coded elements and generalizes to more than one condition.

2 BACKGROUND

2.1 VARIATIONAL AUTOENCODER

In representation learning, one aims to map a vector x to a representation z for which a given downstream task can be performed more efficiently. Hierarchical Bayesian models (Gelman & Hill, 2006) yield probabilistic representations in the form of sufficient statistics for the model's posterior distribution.

Let $\{X, S\}$ denote the set of observed random variables and Z the set of latent variables (Z^i denotes component i). Then Bayesian inference aims to maximize the likelihood:

$$p_\theta(X \mid S) = \int p_\theta(X \mid Z, S) p_\theta(Z \mid S) dZ. \tag{1}$$

Because the integral is in general intractable, variational inference finds a distribution $q_\phi(Z \mid X, S)$ that minimizes a lower bound on the data — the evidence lower bound (ELBO):

$$\log p_\theta(X \mid S) \geq \mathbb{E}_{q_\phi(Z|X,S)}[\log p_\theta(X \mid Z, S)] - D_{\mathrm{KL}}(q_\phi(Z|X,S)||p(Z|S)) \tag{2}$$

In the case of a variational auto-encoder (VAE), the variational distribution is parametrized by a neural network, both the generative model and the variational approximation have conditional distributions parametrized with neural networks. The difference between the data likelihood and the ELBO is the variational gap:

$$D_{\mathrm{KL}}(q_\phi(Z \mid X, S)||p_\theta(Z \mid X, S)). \tag{3}$$

The original AEVB framework is described in the seminal paper (Kingma & Welling, 2013) for the case $Z = \{z\}, X = \{x\}, S = \emptyset$. The representation z is optimized to "explain" the data x. The variational distribution can be used to meet different needs: $q_\phi(y \mid x)$ is a classifier for a class label y and $q_\phi(z \mid x)$ summarizes the data. When using VAE, the empirical data distribution $p_{\text{data}}(X, S)$ is transformed to the representation $\hat{q}_\phi(Z) = \mathbb{E}_{p_{\text{data}}(X,S)} q_\phi(Z \mid X, S)$.

The case in which $S \neq \emptyset$ is referred to as the conditional variational autoencoder (CVAE) (Sohn et al., 2015), and a straight-forward extension of the original framework.

2.2 MAXIMUM-MEAN DISCREPANCY

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. Let $\mathcal{X}$ (resp. $\mathcal{X}'$) be a separable metric space. Let $x : \Omega \to \mathcal{X}$ (resp. $x' : \Omega \to \mathcal{X}'$) be a random variable. Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (resp. $k : \mathcal{X}' \times \mathcal{X}' \to \mathbb{R}$) be a continuous, bounded, positive semi-definite kernel. Let $\mathcal{H}$ be the corresponding reproducing kernel Hilbert space (RKHS) and $\phi : \Omega \to \mathcal{H}$ the corresponding feature mapping.

Consider the kernel-based estimate of *distance* between two distributions p and q over the random variables X and X'. Such a distance, defined via the canonical distance between their $\mathcal{H}$-embeddings, is called the maximum mean discrepancy (Gretton et al., 2012) and denoted $\mathrm{MMD}(p, q)$, with an explicit expression:

$$\ell_{\mathrm{MMD}}(X, X') = \frac{1}{n_0^2} \sum_{n,m} k(x_n, x_m) + \frac{1}{n_1^2} \sum_{n,m} k(x'_n, x'_m) - \frac{2}{n_0 n_1} \sum_{n,m} k(x_n, x'_m), \tag{4}$$

where the sums run over the number of samples n_0 and n_1 for x and x', respectively. Asymptotically, for a universal kernel such as the Gaussian kernel $k(x, x') = e^{-\gamma \|x - x'\|^2}$, $\ell_{\mathrm{MMD}}(X, X')$ is 0 if and only if $p \equiv q$. For the implementation, we use multi-scale RBF kernels defined as:

$$k(x, x') = \sum_{i=1}^{l} k(x, x', \gamma_i) \tag{5}$$

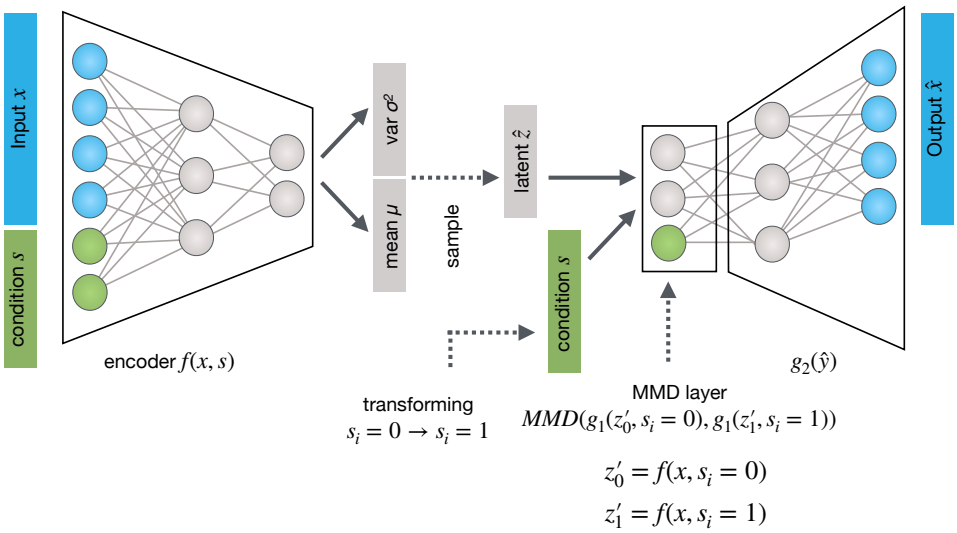

Figure 1: The transformer VAE (trVAE) is an MMD-regularized CVAE. It receives randomized batches of data (x) and condition (s) as input during training, stratified for approximately equal proportions of s. In contrast to a standard CVAE, we regularize the effect of s on the representation obtained after the first-layer $g_1(\hat{z}, s)$ of the decoder g. During prediction time, we transform batches of the source condition $x_{s=0}$ to the target condition $x_{s=1}$ by encoding $\hat{z}_0 = f(x_0, s = 0)$ and decoding $g(\hat{z}_0, s = 1)$.

where $k(x, x', \gamma_i) = e^{-\gamma_i \|x - x'\|^2}$ and γ_i is a hyper-parameter.

Addressing the domain adaptation problem, the "Variational Fair Autoencoder" (VFAE) (Louizos et al., 2015) uses MMD to match latent distributions $q_\phi(z|s = 0)$ and $q_\phi(z|s = 1)$ — where s denotes a domain — by adapting the standard VAE cost function $\mathcal{L}_{\text{VAE}}$ according to

$$\mathcal{L}_{\text{VFAE}}(\phi, \theta; X, X', S, S') = \mathcal{L}_{\text{VAE}}(\phi, \theta; X, S) + \mathcal{L}_{\text{VAE}}(\phi, \theta; X', S') - \beta \ell_{\text{MMD}}(Z_{s=0}, Z'_{s'=1}), \tag{6}$$

where X and X' are two high-dimensional observations with their respective conditions S and S'.

In contrast to GANs (Goodfellow et al., 2014), whose training procedure is notoriously hard due to the minmax optimization problem, training models using MMD or Wasserstein distance metrics is comparatively simple (Li et al., 2015; Arjovsky et al., 2017; Dziugaite et al., 2015a) as only a direct minimization of a straight forward loss is involved during the training. It has been shown that MMD based GANs have some advantages over Wasserstein GANs resulting in a simpler and faster-training algorithm with matching performance (Bińkowski et al., 2018). This motivated us to choose MMD as a metric for regularizing distribution matching.

3 Defining the transformer VAE

Let us adapt the following notation for the transformation within a standard CVAE. High-dimensional observations x and a scalar or low-dimensional condition s are transformed using f (encoder) and g (decoder), which are parametrized by weight-sharing neural networks, and give rise to predictors $\hat{z}, \hat{y}$ and $\hat{x}$:

$$\hat{z} = f(x, s) \tag{7a}$$
$$\hat{y} = g_1(\hat{z}, s) \tag{7b}$$
$$\hat{x} = g_2(\hat{y}) \tag{7c}$$

where we distinguished the first (g_1) and the remaining layers (g_2) of the decoder $g = g_2 \circ g_1$ (Fig. 1).

While z formally depends on s, it is commonly empirically observed $Z \perp\!\!\!\perp S$, that is, the representation z is disentangled from the condition information s. By contrast, the original representation typically

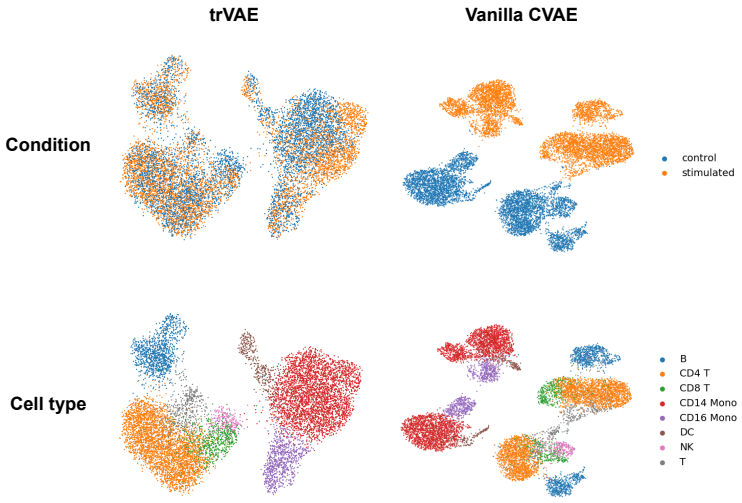

Figure 2: Comparison of representations for MMD-layer in trVAE and the corresponding layer in the vanilla CVAE using UMAP (McInnes et al., 2018). The MMD regularization incentivizes the model to learn condition-invariant features resulting in a more compact representation. The figure shows the qualitative effect for the "PBMC data" introduced in experiments section 4.3. Both representations show the same number of samples.

strongly covaries with S: $X \not\perp\!\!\!\perp S$. The observation can be explained by admitting that an efficient z-representation, suitable for minimizing reconstruction and regularization losses, should be as free as possible from information about s. Information about s is directly and explicitly available to the decoder (equation 7b), and hence, there is an incentive to optimize the parameters of f to *only* explain the variation in x that is *not* explained by s. Experiments below demonstrate that indeed, MMD regularization on the *bottleneck layer* z does not improve performance.

However, even if z is completely free of variation from s, the y representation has a strong s component, $Y \not\perp\!\!\!\perp S$, which leads to a separation of $y_{s=1}$ and $y_{s=0}$ into different regions of their support $\mathcal{Y}$. In the standard CVAE, without any regularization of this y representation, a highly varying, non-compact distribution emerges across different values of s (Fig. 2). To compactify the distribution so that it displays only subtle, controlled differences, we impose MMD (equation 4) in the first layer of the decoder (Fig. 1). We assume that modeling y in the same region of the support of $\mathcal{Y}$ across s forces learning common features across s where possible. The more of these common features are learned, the more accurately the transformation task will performed, and the higher are chances of successful out-of-sample generation. Using one of the benchmark datasets introduced, below, we qualitatively illustrate the effect (Fig. 2).

During training time, all samples are passed to the model with their corresponding condition labels (x_s, s). At prediction time, we pass $(x_{s=0}, s = 0)$ to the encoder f to obtain the latent representation $\hat{z}_{s=0}$. In the decoder g, we pass $(\hat{z}_{s=0}, s = 1)$ and through that, let the model transform data to $\hat{x}_{s=1}$.

The cost function of trVAE derives directly from the standard CVAE cost function, as introduced in the backgrounds section,

$$\mathcal{L}_{\text{CVAE}}(\phi, \theta; X, S, \alpha, \eta) = \eta \mathbb{E}_{q_\theta(Z|X,S)} \log(p_\phi(X|Z, S)) - \alpha D_{\text{KL}}(q_\theta(Z|X, S)||p_\phi(Z|X, S)). \quad (8)$$

Consistent with the above, let $\hat{y}_{s=0} = g_1(f(x, s = 0), s = 0)$ and $\hat{y}_{s=1} = g_1(f(x', s = 1), s = 1)$. Through duplicating the cost function for X' and adding an MMD term, the loss of trVAE becomes:

$$\begin{aligned}\mathcal{L}_{\text{trVAE}}(\phi, \theta; X, X', S, S', \alpha, \eta, \beta) = \mathcal{L}_{\text{CVAE}}(\phi, \theta; X, S, \alpha, \eta) \\ + \mathcal{L}_{\text{CVAE}}(\phi, \theta; X', S', \alpha, \eta) \\ - \beta \ell_{\text{MMD}}(\hat{Y}_{s=0}, \hat{Y}_{s'=1}). \quad (9)\end{aligned}$$

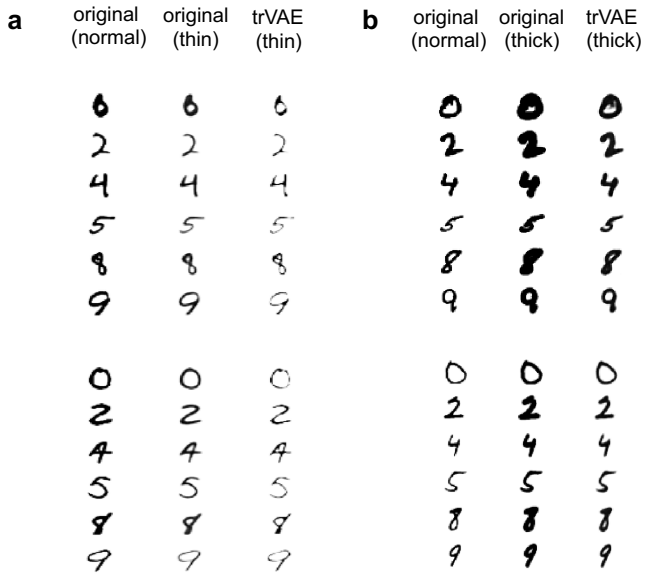

Figure 3: Out-of-sample style transfer for Morpho-MNIST dataset containing normal, thin and thick digits. trVAE successfully transforms normal digits to thin **(a)** and thick (**(b)** for digits not seen during training (out-of-sample).

4 EXPERIMENTS

We demonstrate the advantages of an MMD-regularized first layer of the decoder by benchmarking versus a variety of existing methods and alternatives:

- Vanilla CVAE (Sohn et al., 2015)
- CVAE with MMD on bottleneck (MMD-CVAE), similar to VFAE (Louizos et al., 2015)
- MMD-regularized autoencoder (Dziugaite et al., 2015b; Amodio et al., 2019)
- CycleGAN (Zhu et al., 2017)
- scGen, a VAE combined with vector arithmetics (Lotfollahi et al., 2019)
- scVI, a CVAE with a negative binomial output distribution (Lopez et al., 2018a)

First, we demonstrate trVAE's basic out-of-sample style transfer capacity on two established image datasets, on a qualitative level. We then address quantitative comparisons of challenging benchmarks with clear ground truth, predicting the effects of biological perturbation based on high-dimensional structured data. We used convolutional layers for imaging examples in section 4.1 and fully connected layers for single-cell gene expression datasets in sections 4.2 and 4.3. The optimal hyper-parameters for each application were chosen by using a parameter gird-search for each model. The detailed hyper-parameters for different models are reported in tables 1-9 in appendix A.

4.1 MNIST AND CELEBA STYLE TRANSFORMATION

Here, we use Morpho-MNIST (Castro et al., 2018), which contains 60,000 images each of "normal" and "transformed" digits, which are drawn with a thinner and thicker stroke. For training, we used all normal-stroke data. Hence, the training data covers all domains ($d \in \{0, 1, 2, \ldots, 9\}$) in the normal stroke condition ($s = 0$). In the transformed conditions (thin and thick strokes, $s \in \{1, 2\}$), we only kept domains $d \in \{1, 3, 6, 7\}$.

We train a convolutional trVAE in which we first encode the stroke width via two fully-connected layers with 128 and 784 features, respectively. Next, we reshape the 784-dimensional into 28*28*1 images and add them as another channel in the image. Such trained trVAE faithfully transforms digits of normal stroke to digits of thin and thicker stroke to the out-of-sample domains (Fig. 3)

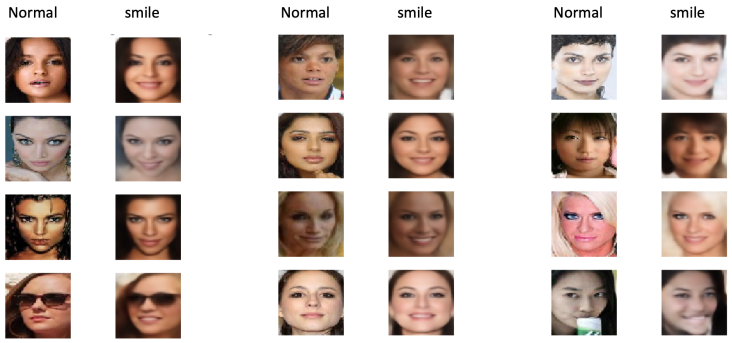

Figure 4: CelebA dataset with images in two conditions: celebrities without a smile and with a smile on their face. trVAE successfully adds a smile on faces of women without a smile despite these samples completely lacking from the training data (out-of-sample). The training data only comprises non-smiling women and smiling and non-smiling men.

Next, we apply trVAE to CelebA (Liu et al., 2015), which contains 202,599 images of celebrity faces with 40 binary attributes for each image. We focus on the task of learning a transformation that turns a non-smiling face into a smiling face. We kept the smiling (s) and gender (d) attributes and trained the model with images from both smiling and non-smiling men but only with non-smiling women.

In this case, we trained a deep convolutional trVAE with a U-Net-like architecture (Ronneberger et al., 2015). We encoded the binary condition labels as in the Morpho-MNIST example and fed them as an additional channel in the input.

Predicting out-of-sample, trVAE successfully transforms non-smiling faces of women to smiling faces while preserving most aspects of the original image (Fig. 4). In addition to showing the model's capacity to handle more complex data, this example demonstrates the flexibility of the the model adapting to well-known architectures like U-Net in the field.

4.2 Infection response

Accurately modeling cell response to perturbations is a key question in computational biology. Recently, neural network models have been proposed for out-of-sample predictions of high-dimensional tabular data that quantifies gene expression of single-cells (Lotfollahi et al., 2019; Amodio et al., 2018). However, these models are not trained on the task relying instead on hard-coded transformations and cannot handle more than two conditions.

We evaluate trVAE on a single-cell gene expression dataset that characterizes the gut (Haber et al., 2017) after Salmonella or Heligmosomoides polygyrus (H. poly) infections, respectively. For this, we closely follow the benchmark as introduced in (Lotfollahi et al., 2019). The dataset contains eight different cell types in four conditions: control or healthy cells (n=3,240), H.Poly infection a after three days (H.Poly.Day3, n=2,121), H.poly infection after 10 days (H.Poly.Day10, n=2,711) and salmonella infection (n=1,770) (Fig. 5a). The normalized gene expression data has 1,000 dimensions corresponding to 1,000 genes. Since three of the benchmark models are only able to handle two conditions, we only included the control and H.Poly.Day10 conditions for model comparisons. In this setting, we hold out Tuft infected cells for training and validation, as these consitute the hardest case for out-of-sample generalization (least shared features, few training data).

Figure 5b-c shows trVAE accurately predicts the mean and variance for high-dimensional gene expression in Tuft cells. We compared the distribution of *Defa24*, the gene with the highest change after H.poly infection in Tuft cells, which shows trVAE provides better estimates for mean and variance compared to other models. Moreover, trVAE outperforms other models also when quantifying the correlation of the predicted 1,000 dimensional x with its ground truth (Fig. 5e). In particular, we note that the MMD regularization on the *bottleneck layer* of the CVAE does not improve performance, as argued above.

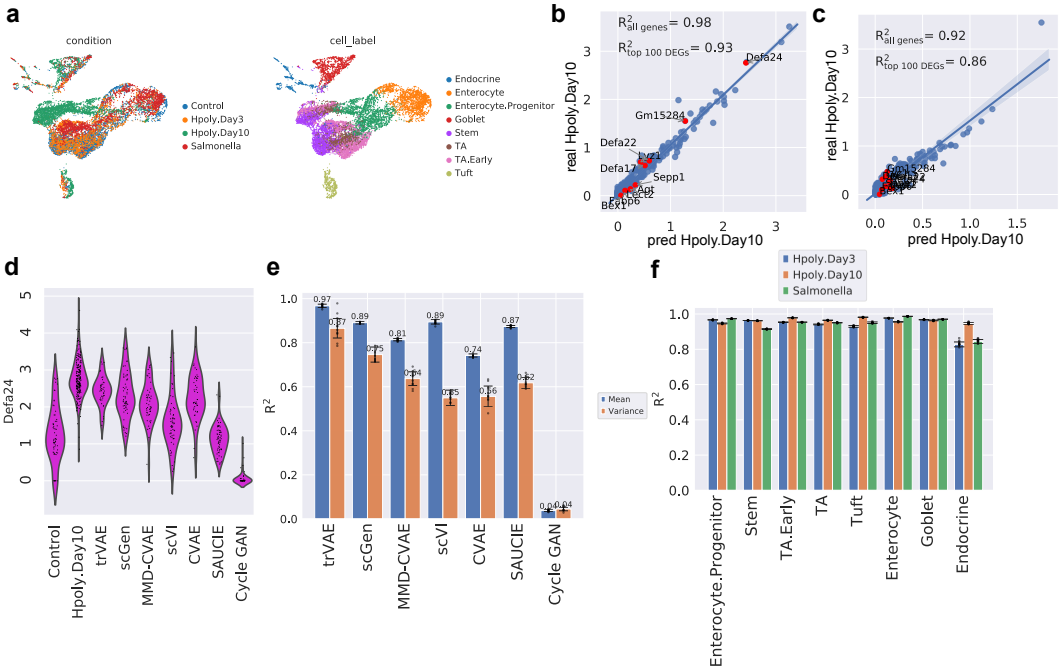

Figure 5: **(a)** UMAP visualization of conditions and cell type for gut cells. **(b-c)** Mean and variance expression of 1,000 genes comparing trVAE-predicted and real infected Tuft cells together with the top 10 differentiall-expressed genes highlighted in red (R^2 denotes Pearson correlation between ground truth and predicted values). **(d)** Distribution of *Defa24*: the top response gene to H.poly.Day10 infection between control, predicted and real stimulated cells for different models. Vertical axis: expression distribution for *Defa24*. Horizontal axis: control, real and predicted distribution by different models. **(e)** Comparison of Pearson's R^2 values for mean and variance gene expression between real and predicted cells for different models. Center values show the mean of R^2 values estimated using $n = 100$ random subsamples for the prediction of each model and error bars depict standard deviation. **(f)** Comparison of R^2 values for mean gene expression between real and predicted cells by trVAE for the eight different cell types and three conditions. Center values show the mean of R^2 values estimated using $n = 100$ random subsamples for each cell type and error bars depict standard deviation.

In order to show our model is able to handle multiple conditions, we performed another experiment with all three conditions included. We trained trVAE holding out each of the eight cells types in all perturbed conditions. Figure 5f shows trVAE can accurately predict all cell types in each perturbed condition, in contrast to existing models.

4.3 Stimulation response

Similar to modeling infection response as above, we benchmark on another single-cell gene expression dataset consisting of 7,217 IFN-β stimulated and 6,359 control peripheral blood mononuclear cells (PBMCs) from eight different human Lupus patients (Kang et al., 2018). The stimulation with IFN-β induces dramatic changes in the transcriptional profiles of immune cells, which causes big shifts between control and stimulated cells (Fig. 6a). We studied the out-of-sample prediction of natural killer (NK) cells held out during the training of the model.

trVAE accurately predicts mean (Fig. 6b) and variance (Fig. 6c) for all genes in the held out NK cells. In particular, genes strongly responding to IFN-β (highlighted in red in Fig. 6b-c) are well captured. An effect of applying IFN-β is an increase in ISG15 for NK cells, which the model never sees during training. trVAE predicts this change by increasing the expression of ISG15 as observed in real NK cells (Fig. 6d). A cycle GAN and an MMD-regularized auto-encoder (SAUCIE) and other models yield less accurate results than our model. Comparing the correlation of predicted mean and variance of gene expression for all dimensions of the data, we find trVAE performs best (Fig. 6e).

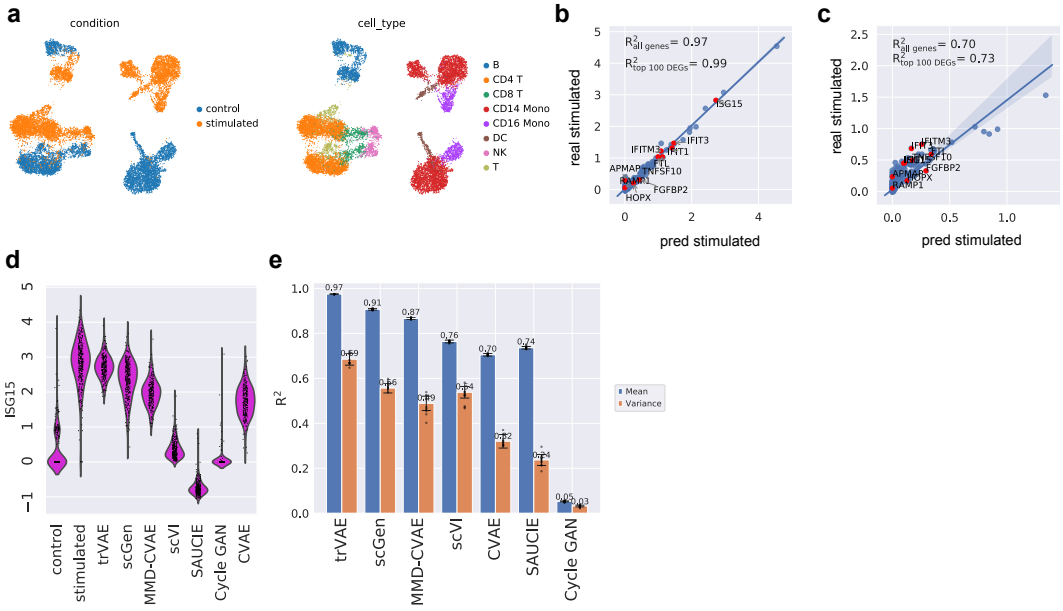

Figure 6: **(a)** UMAP visualization of peripheral blood mononuclear cells (PBMCs). **(b-c)** Mean and variance per 2,000 dimensions between trVAE-predicted and real natural killer cells (NK) together with the top 10 differentially-expressed genes highlighted in red. **(d)** Distribution of *ISG15*: the most strongly changing gene after IFN-β perturbation between control, real and predicted stimulated cells for different models. Vertical axis: expression distribution for *ISG15*. Horizontal axis: control, real and predicted distribution by different models. **(e)** Comparison of R^2 values for mean and variance gene expression between real and predicted cells for different models. Center values show the mean of R^2 values estimated using $n = 100$ random subsamples for the prediction of each model and error bars depict standard deviation.

5 Discussion

By arguing that the vanilla CVAE yields representations in the first layer following the bottleneck that vary strongly across categorical conditions, we introduced an MMD regularization that forces these representations to be similar across conditions. The resulting model (trVAE) outperforms existing modeling approaches on benchmark and real-world data sets.

Within the bottleneck layer, CVAEs already display a well-controlled behavior, and regularization does not improve performance. Further regularization at later layers might be beneficial but is numerically costly and unstable as representations become high-dimensional. However, we have not yet systematically investigated this and leave it for future studies.

Further future work will concern the application of trVAE on larger and more data, focusing on interaction effects among conditions. For this, an important application is the study of drug interaction effects, as previously noted by Amodio et al. (2018). Future conceptual investigations concern establishing connections to causal-inference-inspired models such as CEVAE (Louizos et al., 2017): faithful modeling of an interventional distribution might possibly be re-framed as successful perturbation effect prediction across domains.

References

Matthew Amodio, David van Dijk, Ruth Montgomery, Guy Wolf, and Smita Krishnaswamy. Out-of-sample extrapolation with neuron editing. *arXiv:1805.12198*, 2018.

Matthew Amodio, David van Dijk, Krishnan Srinivasan, William S Chen, Hussein Mohsen, Kevin R. Moon, Allison Campbell, Yujiao Zhao, Xiaomei Wang, Manjunatha Venkataswamy, Anita Desai, V. Ravi, Priti Kumar, Ruth Montgomery, Guy Wolf, and Smita Krishnaswamy. Exploring single-cell data with deep multitasking neural networks. *bioRxiv*, 2019. doi: 10.1101/237065.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

Mikołaj Bińkowski, Dougal J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv:1801.01401*, 2018.

Daniel C. Castro, Jeremy Tan, Bernhard Kainz, Ender Konukoglu, and Ben Glocker. Morpho-MNIST: Quantitative assessment and diagnostics for representation learning. 2018.

Gintare Karolina Dziugaite, Daniel M. Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, UAI'15, pp. 258–267, Arlington, Virginia, United States, 2015a. AUAI Press.

Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. *arXiv:1505.03906*, 2015b.

Andrew Gelman and Jennifer Hill. *Data analysis using regression and multilevel/hierarchical models*. Cambridge university press, 2006.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012.

Adam L. Haber, Moshe Biton, Noga Rogel, Rebecca H. Herbst, Karthik Shekhar, Christopher Smillie, Grace Burgin, Toni M. Delorey, Michael R. Howitt, Yarden Katz, Itay Tirosh, Semir Beyaz, Danielle Dionne, Mei Zhang, Raktima Raychowdhury, Wendy S. Garrett, Orit Rozenblatt-Rosen, Hai Ning Shi, Omer Yilmaz, Ramnik J. Xavier, and Aviv Regev. A single-cell survey of the small intestinal epithelium. *Nature*, 551:333, 2017.

Hyun Min Kang, Meena Subramaniam, Sasha Targ, Michelle Nguyen, Lenka Maliskova, Elizabeth McCarthy, Eunice Wan, Simon Wong, Lauren Byrnes, Cristina M Lanata, et al. Multiplexed droplet single-cell rna-sequencing using natural genetic variation. *Nature biotechnology*, 36(1):89, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013.

Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, pp. 1718–1727, 2015.

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

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. Learning transferable features with deep adaptation networks. *arXiv:1502.02791*, 2015.

Romain Lopez, Jeffrey Regier, Michael B. Cole, Michael I. Jordan, and Nir Yosef. Deep generative modeling for single-cell transcriptomics. *Nature Methods*, 15(12):1053–1058, 2018a.

Romain Lopez, Jeffrey Regier, Michael I Jordan, and Nir Yosef. Information constraints on auto-encoding variational bayes. In *Advances in Neural Information Processing Systems*, pp. 6114–6125, 2018b.

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

Christos Louizos, Kevin Swersky, Yujia Li, Max Welling, and Richard Zemel. The variational fair autoencoder. *arXiv:1511.00830*, 2015.

Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. Causal effect inference with deep latent-variable models. In *Advances in Neural Information Processing Systems*, pp. 6446–6456, 2017.

L. McInnes, J. Healy, and J. Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv:1802.03426*, 2018.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv:1411.1784*, 2014.

Yong Ren, Jun Zhu, Jialian Li, and Yucen Luo. Conditional generative moment-matching networks. In *Advances in Neural Information Processing Systems*, pp. 2928–2936, 2016.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241, 2015.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems 28*, pp. 3483–3491. 2015.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv:1412.3474*, 2014.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

A HYPER-PARAMETERS

Table 1: Convolutional trVAE detailed architecture used for Morpho-MNIST dataset.

Name	Operation	NoF/Kernel Dim.	Dropout	BN	Activation	Input
input	-	(28, 28, 1)	×	×	-	-
conditions	-	2	×	×	-	-
FC-1	FC	128	×	×	Leaky ReLU	conditions
FC-2	FC	784	0.2	√	Leaky ReLU	FC-1
FC-2_resh	Reshape	(28, 28, 1)	×	×	×	FC-2
Conv2D_1	Conv2D	(4, 4, 64, 2)	×	×	Leaky ReLU	[FC-2_resh, input]
Conv2D_2	Conv2D	(4, 4, 64, 64)	×	×	Leaky ReLU	Conv2D_1
FC-3	FC	128	×	√	Leaky ReLU	Flatten(Conv2D_2)
mean	FC	50	×	×	Linear	FC-3
var	FC	50	×	×	Linear	FC-3
z	FC	50	×	×	Linear	[mean, var]
FC-4	FC	128	×	×	Leaky ReLU	conditions
FC-5	FC	784	0.2	√	Leaky ReLU	FC-4
FC-5_resh	Reshape	(28, 28, 1)	×	×	×	FC-5
MMD	FC	128	×	√	Leaky ReLU	[z, FC-5_resh]
FC-6	FC	256	×	×	Leaky ReLU	MMD
FC-7_resh	Reshape	(2, 2, 64)	×	×	×	FC-6
Conv_transp_1	Conv2D Transpose	(4, 4, 128, 64)	×	×	Leaky ReLU	FC-7_resh
Conv_transp_2	Conv2D Transpose	(4, 4, 64, 64)	×	×	Leaky ReLU	UpSampling2D(Conv_tra
Conv_transp_3	Conv2D Transpose	(4, 4, 64, 64)	×	×	Leaky ReLU	Conv_transp_2
Conv_transp_4	Conv2D Transpose	(4, 4, 2, 64)	×	×	Leaky ReLU	UpSampling2D(Conv_tra
output	Conv2D Transpose	(4, 4, 1, 2)	×	×	ReLU	UpSampling2D(Conv_tra
Optimizer	Adam					
Learning Rate	0.001					
Leaky ReLU slope	0.2					
Batch Size	1024					
# of Epochs	5000					
α	0.001					
β	1000					

Table 2: U-Net trVAE detailed architecture used for CelebA dataset.

Name	Operation	NoF/Kernel Dim.	Dropout	BN	Activation	Input
input	-	(64, 64, 3)	×	×	-	-
conditions	-	2	×	×	-	-
FC-1	FC	128	×	×	ReLU	conditions
FC-2	FC	1024	0.2	√	ReLU	FC-1
FC-2_reshaped	Reshape	(64, 64, 1)	×	×	×	FC-2
Conv_1	Conv2D	(3, 3, 64, 4)	×	×	ReLU	[FC-2_reshaped, input]
Conv_2	Conv2D	(3, 3, 64, 64)	×	×	ReLU	Conv_1
Pool_1	Pooling2D	×	×	×	×	Conv_2
Conv_3	Conv2D	(3, 3, 128, 64)	×	×	ReLU	Pool_1
Conv_4	Conv2D	(3, 3, 128, 128)	×	×	ReLU	Conv_3
Pool_2	Pooling2D	×	×	×	×	Conv_4
Conv_5	Conv2D	(3, 3, 256, 128)	×	×	ReLU	Pool_2
Conv_6	Conv2D	(3, 3, 256, 256)	×	×	ReLU	Conv_5
Conv_7	Conv2D	(3, 3, 256, 256)	×	×	ReLU	Conv_6
Pool_3	Pooling2D	×	×	×	×	Conv_7
Conv_8	Conv2D	(3, 3, 512, 256)	×	×	ReLU	Pool_3
Conv_9	Conv2D	(3, 3, 512, 512)	×	×	ReLU	Conv_8
Conv_10	Conv2D	(3, 3, 512, 512)	×	×	ReLU	Conv_9
Pool_4	Pooling2D	×	×	×	×	Conv_10
Conv_11	Conv2D	(3, 3, 512, 256)	×	×	ReLU	Pool_4
Conv_12	Conv2D	(3, 3, 512, 512)	×	×	ReLU	Conv_11
Conv_13	Conv2D	(3, 3, 512, 512)	×	×	ReLU	Conv_12
Pool_4	Pooling2D	×	×	×	×	Conv_13
flat	Flatten	×	×	×	×	Pool_4
FC-3	FC	1024	×	×	ReLU	flat
FC-4	FC	256	0.2	×	ReLU	FC-3
mean	FC	60	×	×	Linear	FC-4
var	FC	60	×	×	Linear	FC-4
z-sample	FC	60	×	×	Linear	[mean, var]
FC-5	FC	128	×	×	ReLU	conditions
MMD	FC	256	×	√	ReLU	[z-sample, FC-5]
FC-6	FC	1024	×	×	ReLU	MMD
FC-7	FC	4096	×	×	ReLU	FC-6
FC-7_reshaped	Reshape	×	×	×	FC-7	
Conv_transp_1	Conv2D Transpose	(3, 3, 512, 512)	×	×	ReLU	FC-7_reshaped
Conv_transp_2	Conv2D Transpose	(3, 3, 512, 512)	×	×	ReLU	Conv_transp_1
Conv_transp_3	Conv2D Transpose	(3, 3, 512, 512)	×	×	ReLU	Conv_transp_2
up_sample_1	UpSampling2D	×	×	×	×	Conv_transp_3
Conv_transp_4	Conv2D Transpose	(3, 3, 512, 512)	×	×	ReLU	up_sample_1
Conv_transp_5	Conv2D Transpose	(3, 3, 512, 512)	×	×	ReLU	Conv_transp_4
Conv_transp_6	Conv2D Transpose	(3, 3, 512, 512)	×	×	ReLU	Conv_transp_5
up_sample_2	UpSampling2D	×	×	×	×	Conv_transp_6
Conv_transp_7	Conv2D Transpose	(3, 3, 128, 256)	×	×	ReLU	up_sample_2
Conv_transp_8	Conv2D Transpose	(3, 3, 128, 128)	×	×	ReLU	Conv_transp_7
up_sample_3	UpSampling2D	×	×	×	×	Conv_transp_8
Conv_transp_9	Conv2D Transpose	(3, 3, 64, 128)	×	×	ReLU	up_sample_3
Conv_transp_10	Conv2D Transpose	(3, 3, 64, 64)	×	×	ReLU	Conv_transp_9
output	Conv2D Transpose	(1, 1, 3, 64)	×	×	ReLU	Conv_transp_10
Optimizer	Adam					
Learning Rate	0.001					
Leaky ReLU slope	0.2					
Batch Size	1024					
# of Epochs	5000					
α	0.001					
β	1000					

Table 3: trVAE detailed architecture. We used the same architecture for all the examples in the paper. The input_dim parameter for each dataset is: IFN-β (2,000), H.poly (1,000).

Name	Operation	NoF/Kernel Dim.	Dropout	BN	Activation	Input
input	-	input_dim	$\times$	$\times$	-	-
conditions	-	n_conditions	$\times$	$\times$	-	-
FC-1	FC	800	0.2	$\surd$	Leaky ReLU	[input, conditions]
FC-2	FC	800	0.2	$\surd$	Leaky ReLU	FC-1
FC-3	FC	128	0.2	$\surd$	Leaky ReLU	FC-2
mean	FC	50	$\times$	$\times$	Linear	FC-3
var	FC	50	$\times$	$\times$	Linear	FC-3
z-sample	FC	50	$\times$	$\times$	Linear	[mean, var]
MMD	FC	128	0.2	$\surd$	Leaky ReLU	[z-sample, conditions]
FC-4	FC	800	0.2	$\surd$	Leaky ReLU	MMD
FC-5	FC	800	0.2	$\surd$	Leaky ReLU	FC-3
output	FC	input_dim	$\times$	$\times$	ReLU	FC-4
Optimizer	Adam					
Learning Rate	0.001					
Leaky ReLU slope	0.2					
Batch Size	512					
# of Epochs	5000					
α	0.00001					
β	100					
η	100					

Table 4: scGen detailed architecture.

Name	Operation	NoF/Kernel Dim.	Dropout	BN	Activation	Input
input	-	input_dim	$\times$	$\times$	-	-
FC-1	FC	800	0.2	$\surd$	Leaky ReLU	input
FC-2	FC	800	0.2	$\surd$	Leaky ReLU	FC-1
FC-3	FC	128	0.2	$\surd$	Leaky ReLU	FC-2
mean	FC	100	$\times$	$\times$	Linear	FC-3
var	FC	100	$\times$	$\times$	Linear	FC-3
z	FC	100	$\times$	$\times$	Linear	[mean, var]
MMD	FC	128	0.2	$\surd$	Leaky ReLU	z
FC-4	FC	800	0.2	$\surd$	Leaky ReLU	MMD
FC-5	FC	800	0.2	$\surd$	Leaky ReLU	FC-3
output	FC	input_dim	$\times$	$\times$	ReLU	FC-4
Optimizer	Adam					
Learning Rate	0.001					
Leaky ReLU slope	0.2					
Batch Size	32					
# of Epochs	300					
α	0.00050					
β	100					
η	100					

Table 5: CVAE detailed architecture.

Name	Operation	NoF/Kernel Dim.	Dropout	BN	Activation	Input
input	-	input_dim	×	×	-	-
conditions	-	1	×	×	-	-
FC-1	FC	800	0.2	√	Leaky ReLU	[input, conditions]
FC-2	FC	800	0.2	√	Leaky ReLU	FC-1
FC-3	FC	128	0.2	√	Leaky ReLU	FC-2
mean	FC	50	×	×	Linear	FC-3
var	FC	50	×	×	Linear	FC-3
z-sample	FC	50	×	×	Linear	[mean, var]
MMD	FC	128	0.2	√	Leaky ReLU	[z-sample, conditions]
FC-4	FC	800	0.2	√	Leaky ReLU	MMD
FC-5	FC	800	0.2	√	Leaky ReLU	FC-3
output	FC	input_dim	×	×	ReLU	FC-4
Optimizer	Adam					
Learning Rate	0.001					
Leaky ReLU slope	0.2					
Batch Size	512					
# of Epochs	300					
α	0.001					

Table 6: MMD-CVAE detailed architecture.

Name	Operation	NoF/Kernel Dim.	Dropout	BN	Activation	Input
input	-	input_dim	×	×	-	-
conditions	-	1	×	×	-	-
FC-1	FC	800	0.2	√	Leaky ReLU	[input, conditions]
FC-2	FC	800	0.2	√	Leaky ReLU	FC-1
FC-3	FC	128	0.2	√	Leaky ReLU	FC-2
mean	FC	50	×	×	Linear	FC-3
var	FC	50	×	×	Linear	FC-3
z-sample	FC	50	×	×	Linear	[mean, var]
MMD	FC	128	0.2	√	Leaky ReLU	[z-sample, conditions]
FC-4	FC	800	0.2	√	Leaky ReLU	MMD
FC-5	FC	800	0.2	√	Leaky ReLU	FC-3
output	FC	input_dim	×	×	ReLU	FC-4
Optimizer	Adam					
Learning Rate	0.001					
Leaky ReLU slope	0.2					
Batch Size	512					
# of Epochs	500					
α	0.001					
β	1					

Table 7: Style transfer GAN detailed architecture.

Name	Operation	NoF/Kernel Dim.	Dropout	BN	Activation	Input
input	-	input_dim	×	×	-	-
FC-1	FC	700	0.5	√	Leaky ReLU	input
FC-2	FC	100	0.5	√	Leaky ReLU	FC-1
FC-3	FC	50	0.5	√	Leaky ReLU	FC-2
FC-4	FC	100	0.5	√	Leaky ReLU	FC-3
FC-5	FC	700	0.5	√	Leaky ReLU	FC-4
generator_out	FC	6,998	×	√	ReLU	FC-5
FC-6	FC	700	0.5	√	Leaky ReLU	generator_out
FC-7	FC	100	0.5	√	Leaky ReLU	FC-6
discriminator_out	FC	1	×	×	Sigmoid	FC-7
Generator Optimizer	Adam					
Discriminator Optimizer	Adam					
Optimizer	Adam					
Learning Rate	0.001					
Leaky ReLU slope	0.2					
# of Epochs	1000					

Table 8: scVI detailed architecture.

Name	Operation	NoF/Kernel Dim.	Dropout	BN	Activation	Input
input	-	input_dim	×	×	-	-
conditions	-	1	×	×	-	-
FC-1	FC	128	0.2	√	ReLU	input
mean	FC	10	×	×	Linear	FC-1
var	FC	10	×	×	Linear	FC-1
z	FC	10	×	×	Linear	[mean, var]
FC-2	FC	128	0.2	√	ReLU	[z, conditions]
output	FC	input_dim	×	×	ReLU	FC-2
Optimizer	Adam					
Learning Rate	0.001					
Batch Size	128					
# of Epochs	1000					
α	0.001					

Table 9: SAUCIE detailed architecture.

Name	Operation	NoF/Kernel Dim.	Dropout	BN	Activation	Input
input	-	input_dim	×	×	-	-
conditions	-	1	×	×	-	-
FC-1	FC	512	×	√	Leaky ReLU	[input, conditions]
FC-2	FC	256	×	×	Leaky ReLU	FC-1
FC-3	FC	128	×	×	Leaky ReLU	FC-2
FC-4	FC	20	×	×	Leaky ReLU	FC-3
FC-5	FC	128	×	×	Leaky ReLU	FC-4
FC-6	FC	256	×	×	Leaky ReLU	FC-5
FC-7	FC	512	×	×	Leaky ReLU	FC-6
output	FC	input_dim	×	×	ReLU	FC-4
Optimizer	Adam					
Learning Rate	0.001					
Leaky ReLU slope	0.2					
Batch Size	256					
# of Epochs	1000					