# A Appendix

## A.1 Representation alignment in InfoNCE with cosine similarity

Pointwise mutual information (PMI) is a measurement of association that compares the probability of two events $x$ and $x'$ happening jointly with their probability of happening independently, defined as:

$$\text{PMI}(x, x') = \log \frac{p(x, x')}{p(x)p(x')} = \log \frac{p(x'|x)}{p(x')} \tag{4}$$

PMI values reflect, in log scale, the likelihood of observing $x'$ having observed $x$ relative to otherwise. In the case of synthetic augmentation, $p(x'|x) \gg p(x')$ if $x'$ is an augmentation of $x$, and $p(x'|x) = 0$ otherwise, hence PMI$(x, x')$ is a small positive value reflective of the number of augmentations, e.g. 5, or unboundedly negative.

The InfoNCE(17) objective is optimised when representations $z, z'$ of samples $x, x'$ satisfy $sim(z, z') = PMI(x, x') + c(x)$, where $sim(\cdot)$ is the similarity function, e.g. cosine similarity (sim$(z, z') = \frac{z^T z}{||z||_2||z'||_2}$), and $c$ is a scalar that can vary with $x$. Us of the bounded popular cosine similarity function restricts the ability for the optimality condition to be reached, instead the optimization of this *restricted* InfoNCE objective leads to representations of similar data being aligned ($z = z'$) and representations of dissimilar data being maximally dispersed.

## A.2 Relationship between Representations and PMI

When considering why representations learned by InfoNCE are useful, which intuitively pertains to the *information* they capture, the fact that the loss function is optimised when representations satisfy a relationship to pointwise mutual *information* seems highly relevant (§2). Even more so, since an analogous relationship underpins properties of word2vec learned word embeddings (§2). However, several further observations undermine this natural line of thought:

(i) Closer approximations of mutual information do not appear to improve representations (21);

(ii) As discussed in §3.1, employing **cosine similarity** sim(x,x')$=\frac{z^T z'}{|z||z'|} \in [-1, 1]$ often leads to better downstream performance than using *unbounded* similarity functions, e.g. dot product, even though PMI values can fall far outside the bounded range [-1,1]; and

(iii) Several recent self-supervised methods take a different contrastive approach, with the aim of circumventing negative sampling, showing no clear relationship to PMI and yet perform well (1).

## A.3 Objective derivation

Let $\mathbf{x} = \{x^1, ..., x^j\}$, with $j \leq N$, be a set of $N$ samples generated through augmentations, as described in section A.4. Let $\theta = \{\theta_x, \theta_z, \pi\}$ and $\phi = \{\phi_z, \phi_y\}$ be parameters of the model and approximate posterior, respectively. We derive the Evidence Lower Bound (ELBO) used as the SimVAE optimization objective and described in section 3.2 as:

$$
\begin{aligned}
\min_{\theta} D_{\mathrm{KL}}[\, p(\mathbf{x}) \,\|\, p_\theta(\mathbf{x}) \,] &= \max_{\theta} \mathop{\mathbb{E}}_{\mathbf{x}}\Big[\log p_\theta(\mathbf{x})\Big] \\
&= \max_{\theta,\phi} \mathop{\mathbb{E}}_{\mathbf{x}}\Big[\int_{\mathbf{z}} \sum_{y} q_\phi(y, \mathbf{z}|\mathbf{x}) \log p_\theta(\mathbf{x})\Big] \\
&= \max_{\theta,\phi} \mathop{\mathbb{E}}_{\mathbf{x}}\Big[\int_{\mathbf{z}} \sum_{y} q_\phi(y, \mathbf{z}|\mathbf{x}) \log p_\theta(\mathbf{x}) \tfrac{q_\phi(y,\mathbf{z}|\mathbf{x})}{q_\phi(y,\mathbf{z}|\mathbf{x})}\Big] \\
&= \max_{\theta,\phi} \mathop{\mathbb{E}}_{\mathbf{x}}\Big[\int_{\mathbf{z}} \sum_{y} q_\phi(y, \mathbf{z}|\mathbf{x}) \log \tfrac{p_{\theta_x}(\mathbf{x}|\mathbf{z}) p_{\theta_z}(\mathbf{z}|y) p_\pi(y)}{p_\theta(y,\mathbf{z}|\mathbf{x})} \tfrac{q_\phi(y,\mathbf{z}|\mathbf{x})}{q_\phi(y,\mathbf{z}|\mathbf{x})}\Big] \\
&= \max_{\theta,\phi} \mathop{\mathbb{E}}_{\mathbf{x}}\Big[\int_{\mathbf{z}} \sum_{y} q_\phi(y, \mathbf{z}|\mathbf{x}) \log \tfrac{p_{\theta_x}(\mathbf{x}|\mathbf{z}) p_{\theta_z}(\mathbf{z}|y) p_\pi(y)}{q_\phi(y,\mathbf{z}|\mathbf{x})}\Big] + D_{\mathrm{KL}}[\, q_\phi(y, \mathbf{z}|\mathbf{x}) \,\|\, p_\theta(y, \mathbf{z}|\mathbf{x}) \,] \\
&\geq \max_{\theta,\phi} \mathop{\mathbb{E}}_{\mathbf{x}}\Big[\int_{\mathbf{z}} \sum_{y} q_\phi(y, \mathbf{z}|\mathbf{x}) \log \tfrac{p_{\theta_x}(\mathbf{x}|\mathbf{z}) p_{\theta_z}(\mathbf{z}|y) p_\pi(y)}{q_\phi(y,\mathbf{z}|\mathbf{x})}\Big] \\
&= \max_{\theta,\phi} \mathop{\mathbb{E}}_{\mathbf{x}}\Big[\int_{\mathbf{z}} \sum_{y} q_{\phi_z}(\mathbf{z}|\mathbf{x}) p_{\phi_y}(y|\mathbf{z}) \log \tfrac{p_{\theta_x}(\mathbf{x}|\mathbf{z}) p_{\theta_z}(\mathbf{z}|y) p_\pi(y)}{q_{\phi_z}(\mathbf{z}|\mathbf{x}) q_{\phi_y}(y|\mathbf{z})}\Big] \\
&= \max_{\theta,\phi} \mathop{\mathbb{E}}_{\mathbf{x}}\Big[\int_{\mathbf{z}} q_{\phi_z}(\mathbf{z}|\mathbf{x}) \big\{\log \tfrac{p_{\theta_x}(\mathbf{x}|\mathbf{z})}{q_{\phi_z}(\mathbf{z}|\mathbf{x})} + \sum_{y} q_{\phi_y}(y|\mathbf{z}) \log \tfrac{p_{\theta_z}(\mathbf{z}|y) p_\pi(y)}{q_{\phi_y}(y|\mathbf{z})}\big\}\Big] \\
&= \max_{\theta,\phi} \mathop{\mathbb{E}}_{\mathbf{x}} \underbrace{\int_{\mathbf{z}} q_{\phi_z}(\mathbf{z}|\mathbf{x}) \log p_{\theta_x}(\mathbf{x}|\mathbf{z})}_{-\mathrm{recon}(\mathbf{x})} \underbrace{- \int_{\mathbf{z}} q_{\phi_z}(\mathbf{z}|\mathbf{x}) \log q_{\phi_z}(\mathbf{z}|\mathbf{x})}_{H_{q_\phi(\mathbf{z}|\mathbf{x})}} \\
&\qquad\qquad + \int_{\mathbf{z}} q_{\phi_z}(\mathbf{z}|\mathbf{x}) \sum_{y} p_{\pi,\theta_z}(y|\mathbf{z}) \log p_{\theta_z}(\mathbf{z}|y) p_\pi(y)
\end{aligned}
$$

where recon$(\cdot)$ refers to the *reconstruction loss*, $H$ to the entropy and $D_{\mathrm{KL}}$ to the KL-divergence. In the last step, we use $\max_{\phi_y} q_{\phi_y}(y|\mathbf{z}) = p_{\pi,\theta_z}(y|\mathbf{z}) \doteq \frac{p_{\theta_z}(\mathbf{z}|y) p_\pi(y)}{\sum_{y'} p_{\theta_z}(\mathbf{z}|y') p_\pi(y')}$ using Bayes' rule since $y$ is assumed to be discrete in this case. In the setting with $N = 2$ related samples, $\mathbf{x} = \{x, x'\}$, the SimVAE objective can be formulated as:

$$
\begin{aligned}
\min_{\theta} D_{\mathrm{KL}}[\, p(\mathbf{x}) \,\|\, p_\theta(\mathbf{x}) \,] \geq \max_{\theta,\phi} \mathop{\mathbb{E}}_{\mathbf{x}} & \underbrace{\int_{z} q_\phi(z|x) \log p_{\theta_x}(x|z)}_{-\mathrm{recon}(x)} + \underbrace{\int_{z'} q_\phi(z'|x') \log p_{\theta_x}(x'|z')}_{-\mathrm{recon}(x')} \\
& \underbrace{- \int_{z} q_\phi(z|x) \log q_\phi(z|x)}_{H_{q_\phi(z|x)}} \underbrace{- \int_{z'} q_\phi(z'|x') \log q_\phi(z'|x')}_{H_{q_\phi(z'|x')}} \\
& + \int_{\mathbf{z}} q_\phi(\mathbf{z}|\mathbf{x}) \sum_{y} p_{\pi,\theta_z}(y|\mathbf{z}) \log p_{\theta_z}(\mathbf{z}|y) p_\pi(y)
\end{aligned}
$$

Algorithm 1 provides an overview of the main computational steps required for the training of the SimVAE evidence lower bound detailed above.

**Algorithm 1** SimVAE

**Require:** data $\{\mathbf{x}_k\}_{k=1}^{M}$; batch size $N$; data dimension $D$; augmentation set $\mathcal{T}$; latent dimension $L$; number of augmentations $A$; encoder network $f_{\boldsymbol{\phi}}$; decoder network $g_{\boldsymbol{\theta}}$; prior variance $\{\sigma_l^*\}_{l=1}^{L}$

**for** randomly sampled mini-batch $\{\mathbf{x}_k\}_{k=1}^{N}$ **do**

# augment mini-batch

$\{t_a\}_{a=1}^{A} \sim \mathcal{T}$;

$\{\mathbf{x}_k^a\}_{a=1}^{A} = \{t_a(\mathbf{x}_k)\}_{a=1}^{A}$;

# forward pass : $\mathbf{z} \sim p(\mathbf{z}|\mathbf{x}), \tilde{\mathbf{x}} \sim p(\mathbf{x}|\mathbf{z})$

$\{(\boldsymbol{\mu}_k^a, \boldsymbol{\Sigma}_k^a) = f_{\boldsymbol{\phi}}(\mathbf{x}_k^a)\}_{a=1}^{A}$;

$\{\mathbf{z}_k^a \sim \mathcal{N}(\boldsymbol{\mu}_k^a, \boldsymbol{\Sigma}_k^a)\}_{a=1}^{A}$;

$\{\tilde{\mathbf{x}}_k^a = g_{\boldsymbol{\theta}}(\mathbf{z}_k^a)\}_{a=1}^{A}$;

# compute & minimize loss terms

$\mathcal{L}_{\text{rec}}^k = \frac{1}{\boldsymbol{\sigma} N D} \sum_{a=1}^{A} \sum_{d=1}^{D} (x_{k,d}^a - \tilde{x}_{k,d}^a)^2$

$\mathcal{L}_{\text{H}}^k = L \log(2\pi e) + \frac{1}{2} \sum_{a=1}^{A} \log(|\boldsymbol{\Sigma}_k^a|)$

$\boldsymbol{\mu}_k^* = \frac{1}{A} \sum_{a=1}^{A} \mathbf{z}_k^a$

$\mathcal{L}_{\text{prior}}^k = N + AL \log(\sqrt{2\pi}) + A \sum_{l=1}^{L} \log(\sigma_l^*) + \sum_{a=1}^{A} \sum_{l=1}^{L} \frac{1}{2\sigma_l^*} (z_{k,l}^a - \mu_{k,l}^*)^2$

$\min(\mathcal{L} = \frac{1}{N} \sum_{k=1}^{N} \mathcal{L}_{\text{rec}}^k + \mathcal{L}_{\text{H}}^k + \mathcal{L}_{\text{prior}}^k)$ w.r.t $\boldsymbol{\phi}, \boldsymbol{\theta}$ by SGD;

**end for**

**return** $\boldsymbol{\phi}, \boldsymbol{\theta}$;

## A.4 Experimental Details

### A.4.1 Datasets

**FashionMNIST** The FashionMNIST dataset (24) is a collection of 60'000 training and 10'000 test images depicting Zalando clothing items (i.e., t-shirts, trousers, pullovers, dresses, coats, sandals, shirts, sneakers, bags and ankle boots). Images were kept to their original 28x28 pixel resolution. The 10-class clothing type classification task was used for evaluation.

**CIFAR10** The CIFAR10 dataset (14) offers a compact dataset of 60,000 (50,000 training and 10,000 testing images) small, colorful images distributed across ten categories including objects like airplanes, cats, and ships, with various lighting conditions. Images were kept to their original 32x32 pixel resolution.

**Celeb-A** The Celeb-A dataset (15) comprises a vast collection of celebrity facial images. It encompasses a diverse set of 183'000 high-resolution images (i.e., 163'000 training and 20'000 test images), each depicting a distinct individual. The dataset showcases a wide range of facial attributes and poses and provides binary labels for 40 facial attributes including hair & skin color, presence or absence of attributes such as eyeglasses and facial hair. Each image was cropped and resized to a 64x64 pixel resolution. Attributes referring to hair color were aggregated into a 5-class attribute (i.e., bald, brown hair, blond hair, gray hair, black hair). Images with missing or ambiguous hair color information were discarded at evaluation.

All datasets were sourced from Pytorch's dataset collection.

### A.4.2 Data augmentation strategy

Taking inspiration from SimCLR's (3) augmentation strategy which highlights the importance of random image cropping and color jitter on downstream performance, our augmentation strategy includes random image cropping, random image flipping and random color jitter. The color augmentations are only applied to the non gray-scale datasets (i.e., CIFAR10 (14) & Celeb-A dataset (15)). Due to the varying complexity of the datasets we explored, hyperparameters such as the cropping strength were adapted to each dataset to ensure that semantically meaningful features remained after augmentation. The augmentation strategy hyperparameters used for each dataset are detailed in table 3.

| **Dataset** | **Crop** | | **Vertical Flip** | **Color Jitter** | | |
|---|---|---|---|---|---|---|
| | scale | ratio | prob. | b-s-c | hue | prob. |
| MNIST | 0.4 | [0.75,1.3] | 0.5 | - | - | - |
| Fashion | 0.4 | [0.75,1.3] | 0.5 | - | - | - |
| CIFAR10 | 0.6 | [0.75,1.3] | 0.5 | 0.8 | 0.2 | 0.8 |
| Celeb-A | 0.6 | [0.75,1.3] | 0.5 | 0.8 | 0.2 | 0.8 |

Table 3: Data augmentation strategy for each dataset: (from left to right) cropping scale, cropping ratio, probability of vertical and horizontal flipping, brightness-saturation-contrast jitter strength, hue jitter strength, probability of color jitter

### A.4.3 Training Implementation Details

This section contains all details regarding the architectural and optimization design choices used to train SimVAE and all baselines. Method-specific hyperparameters are also reported below.

**Datasets and Evaluation Metrics** We evaluated SimVAE on three benchmark datasets including two with natural images: FashionMNIST (24), Celeb-A (15) and CIFAR10 (14). We augment images following the SimCLR (3) protocol which includes cropping and flipping as well as color jitter for natural images. We evaluate representations' utility for downstream classification tasks using a linear probe, a non-linear MLP probe, and k-nearest neighbors (kNN) (4) trained on the pre-trained frozen representations using image labels (3; 2). Additionally, we conducted a fully unsupervised evaluation by fitting a Gaussian mixture model (GMM) to the frozen features for which the number of clusters was set to its ground-truth value. Downstream performance is measured in terms of classification accuracy (CA). A model's generative quality was evaluated using the Fréchet Inception Distance (FID) (9), reconstruction error as well as the Normalized Mutual Information (NMI) and Adjusted Rank Index (ARI) clustering scores (see appendix A.5).

**Baselines methods** We compare SimVAE to other VAE-based models including the vanilla VAE (13), $\beta$-VAE (10) and CR-VAE (19), as well as to state-of-the-art self-supervised discriminative methods including SimCLR (3), VicREG (1), and MoCo (8). As a lower bound, we also provide results obtained for randomly initialized embeddings. To ensure fair comparison, the augmentation strategy, representation dimensionality, batch size, and encoder-decoder architectures were kept invariant across methods. To enable a qualitative comparison of representations, decoder networks were trained for each discriminative baseline on top of frozen representations using the reconstruction error. See appendices A.4.3 and A.4.4 for further details on training baselines and decoder models.

**Hyperparameters** We use MLP and Resnet18 (7) network architectures for simple and natural image datasets respectively. We fix the dimension of representations $z$ to 10 for FashionMNIST, and to 64 for Celeb-A and CIFAR10 datasets. For all generative approaches, we adopt Gaussian posteriors, priors, and likelihoods, employing diagonal covariance matrices as in (13). We fix covariances of the prior and likelihood distributions and perform a hyper-parameter search. SimVAE conveniently allows for the simultaneous incorporation of sets of related observations. After tuning, we fix the number of augmentations to 6 (see Figure 4 for an ablation). For baselines, all sensitive hyperparameters were tuned independently for each dataset and method.

**Network Architectures** The encoder network architectures used for SimCLR, MoCo, VicReg, and VAE-based approaches including SimVAE for simple (i.e., FashionMNIST ) and complex datasets (i.e., CIFAR10, Celeb-A) are detailed in table 4a, table 5a respectively. Generative models which include all VAE-based methods also require decoder networks for which the architectures are detailed in table 4b and table 5b. The encoder and decoder architecture networks are kept constant across methods including the latent dimensionality to ensure a fair comparison across methods.

**Optimisation & Hyper-parameter tuning** All methods were trained using an Adam optimizer until training loss convergence. A learning rate tuning was performed for each method independently

| Layer Name | Output Size | Block Parameters |
|---|---|---|
| fc1 | 500 | 784x500 fc, relu |
| fc2 | 500 | 500x500 fc, relu |
| fc3 | 2000 | 500x2000 fc, relu |
| fc4 | 10 | 2000x10 fc |

(a) Encoder

| Layer Name | Output Size | Block Parameters |
|---|---|---|
| fc1 | 2000 | 10x2000 fc, relu |
| fc2 | 500 | 2000x500 fc, relu |
| fc3 | 500 | 500x500 fc, relu |
| fc4 | 784 | 500x784 fc |

(b) Decoder

Table 4: Multi-layer perceptron network architectures used for FashionMNIST training

| Layer Name | Output | Block Parameters |
|---|---|---|
| conv1 | 32x32 | 4x4, 16, stride 1<br>batchnorm, relu<br>3x3 maxpool, stride 2 |
| conv2_x | 32x32 | 3x3, 32, stride 1<br>3x3, 32, stride 1 |
| conv3_x | 16x16 | 3x3, 64, stride 2<br>3x3, 64, stride 1 |
| conv4_x | 8x8 | 3x3, 128, stride 2<br>3x3, 128, stride 1 |
| conv5_x | 4x4 | 3x3, 256, stride 2<br>3x3, 256, stride 1 |
| fc | 64 | 4096x64 fc |

(a) Encoder

| Layer Name | Output | Block Parameters |
|---|---|---|
| fc | 256x4x4 | 64x4096 fc |
| conv1_x | 8x8 | 3x3, 128, stride 2<br>3x3, 128, stride 1 |
| conv2_x | 16x16 | 3x3, 64, stride 2<br>3x3, 64, stride 1 |
| conv3_x | 32x32 | 3x3, 32, stride 2<br>3x3, 32, stride 1 |
| conv4_x | 64x64 | 3x3, 16, stride 2<br>3x3, 16, stride 1 |
| conv5 | 64x64 | 5x5, 3, stride 1 |

(b) Decoder

Table 5: Resnet18 network architectures used for CIFAR10 & Celeb-A training

across the range $1e^{-3}$ to $8e^{-5}$. A fixed batch size of 128 was used across methods and datasets. The $\beta$, $\tau$, $\lambda$ parameters for the $\beta$-VAE, SimCLR and CRVAE methods were tuned across the [0.1,0.2,0.5], [0.1,0.5,1.0] and [0.01,0.1,1.0] ranges respectively based on downstream performance. $\beta = 0.1$, $\lambda = 0.01$ were selected and $\tau = 1.0$, $\tau = 0.5$ were chosen for simple and natural datasets respectively. The likelihood probability variance for VAE-based methods including SimVAE was kept to $\sigma^2 = 1.0$ and the prior probability, $p(z|y)$, variance parameter for SimVAE was tuned and fixed to 0.003, 0.005, 0.005 for FashionMNIST, CIFAR10 and Celeb-A respectively.

#### A.4.4 Evaluation Implementation Details

Following common practices (3), downstream performance is assessed using a linear probe, a multi-layer perceptron probe, a k-nearest neighbors (kNN) algorithm, and a Gaussian mixture model (GMM). The linear probe consists of a fully connected layer whilst the mlp probe consists of two fully connected layers with a relu activation for the intermediate layer. Both probes were trained using an Adam optimizer with a learning rate of 3e-4 for 200 epochs with batch size fixed to 128. Scikit-learn's Gaussian Mixture model with a full covariance matrix and 200 initialization was fitted to the representations using the ground truth cluster number. The kNN algorithm from Python's Scikit-learn library was used with k spanning from 1 to 15 neighbors. The best performance was chosen as the final performance measurement. No augmentation strategy was used at evaluation.

#### A.4.5 Generation Protocol

In this section, we detail the image generation protocol as well as the evaluation of the quality of the generated samples.

**Ad-hoc decoder training** VAE-based approaches, including SimVAE, are fundamentally generative methods aimed at approximating the logarithm of the marginal likelihood distribution, denoted as $\log p(x)$. In contrast, most traditional self-supervised methods adopt a discriminative framework without a primary focus on accurately modeling $p(x)$. However, for the purpose of comparing representations, and assessing the spectrum of features present in $z$, we intend to train a decoder model for SimCLR & VicReg models. This decoder model is designed to reconstruct images from the fixed representations initially trained with these approaches. To achieve this goal, we train decoder networks using the parameter configurations specified in Tables 4b and 5b, utilizing the mean squared reconstruction error as the loss function. The encoder parameters remain constant, while we update the decoder parameters using an Adam optimizer with a learning rate of $1e^{-4}$ until convergence is achieved (i.e. $\sim 200$ epochs).

**Conditional Image Generation** To allow for a fair comparison, all images across all methods are generated by sampling $z$ from a multivariate Gaussian distribution fitted to the training samples' representations. More precisely, each Gaussian distribution is fitted to $z$ conditioned on a label $y$. Scikit-Learn Python library Gaussian Mixture model function (with full covariance matrix) is used.

## A.5 Additional Results

### A.5.1 Self-supervised classification

**Clustering metrics** Table 6 and table 7 report the normalized mutual information (NMI) and adjusted rank index (ARI) for the fitting of a GMM to latent representations $z$.

| Dataset | | Random | VAE | $\beta$-VAE | CR-VAE | SimVAE |
|---|---|---|---|---|---|---|
| **Fashion** | ARI | $28.7 \pm 0.6$ | $44.2 \pm 1.1$ | $44.7 \pm 0.2$ | $23.3 \pm 0.8$ | $\mathbf{55.7 \pm 0.0}$ |
| | NMI | $51.5 \pm 0.2$ | $66.7 \pm 0.7$ | $66.4 \pm 0.4$ | $46.1 \pm 2.2$ | $\mathbf{76.8 \pm 0.2}$ |
| **Celeb-A** | ARI | $3.4 \pm 0.3$ | $5.7 \pm 0.2$ | $6.2 \pm 0.7$ | $6.6 \pm 0.9$ | $2.6 \pm 0.7$ |
| | NMI | $4.2 \pm 0.4$ | $3.9 \pm 0.2$ | $4.7 \pm 0.9$ | $5.0 \pm 0.7$ | $2.9 \pm 0.7$ |
| **CIFAR10** | ARI | $0.09 \pm 0.0$ | $0.7 \pm 0.2$ | $0.7 \pm 0.2$ | $0.9 \pm 0.1$ | $\mathbf{8.6 \pm 0.3}$ |
| | NMI | $27.9 \pm 0.1$ | $17.7 \pm 0.5$ | $18.7 \pm 0.3$ | $18.9 \pm 0.1$ | $\mathbf{37.2 \pm 0.4}$ |

Table 6: Normalized mutual information (NMI) and Adjusted Rank Index (ARI) for all generative methods and datasets; Average scores and standard errors are computed across three random seeds

| Dataset | | MoCo | VicReg | SimCLR |
|---|---|---|---|---|
| **Fashion** | ARI | $30.9 \pm 0.5$ | $37.1 \pm 1.3$ | $\mathbf{50.3 \pm 1.9}$ |
| | NMI | $50.4 \pm 0.6$ | $64.5 \pm 0.7$ | $\mathbf{71.2 \pm 1.0}$ |
| **Celeb-A** | ARI | $-$ | $\mathbf{18.7} \pm 0.8$ | $0.0 \pm 0.1$ |
| | NMI | $-$ | $\mathbf{24.3} \pm 0.3$ | $0.0 \pm 0.0$ |
| **CIFAR10** | ARI | $27.2 \pm 1.0$ | $31.2 \pm 0.2$ | $\mathbf{49.6 \pm 1.3}$ |
| | NMI | $16.5 \pm 0.4$ | $\mathbf{53.4 \pm 0.1}$ | $26.9 \pm 0.8$ |

Table 7: Normalized mutual information (NMI) and Adjusted Rank Index (ARI) for all discriminative baselines and datasets; Average scores and standard errors are computed across three random seeds

**Augmentation Protocol Strength** Figure 3 reports the downstream CA across methods for various augmentations stategy. More precisely, we progressively increase the cropping scale and color jitter amplitude. Unsurprinsingly (3), discriminative methods exhibit high sensitivity to the augmentation strategy with stronger disruption leading to improved content prediction. The opposite trend is observed with vanilla generative methods where reduced variability amongst the data leads to increased downstream performance. Interestingly, SimVAE is robust to augmentation protocol and performs comparably across settings.

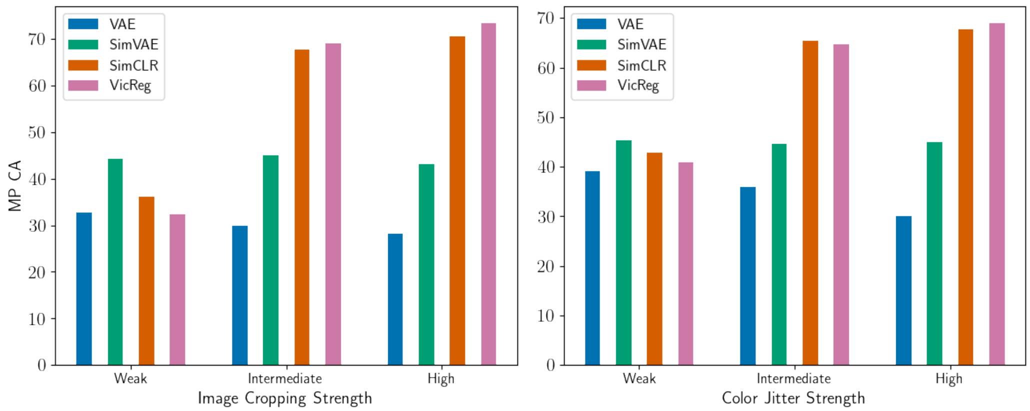


Figure 3: Ablation experiment across the number of augmentations considered during training of the SimVAE model using the MNIST (left) and FashionMNIST (right) datasets. Two, four, six and eight augmentations were considered. The average and standard deviation of the downstream classification accuracy using KNN and GMM probes are reported across three seeds.

**# Augmentation Ablation** Figure 4 reports the downstream classification accuracy for increasing numbers of augmentations considered simultaneously during the training of SimVAE. A larger number of augmentations result in a performance increase up to a certain limit (i.e., 6-8 augmentations). Further exploration is needed to understand how larger sets of augmentations can be effectively leveraged potentially by allowing for batch size increase.

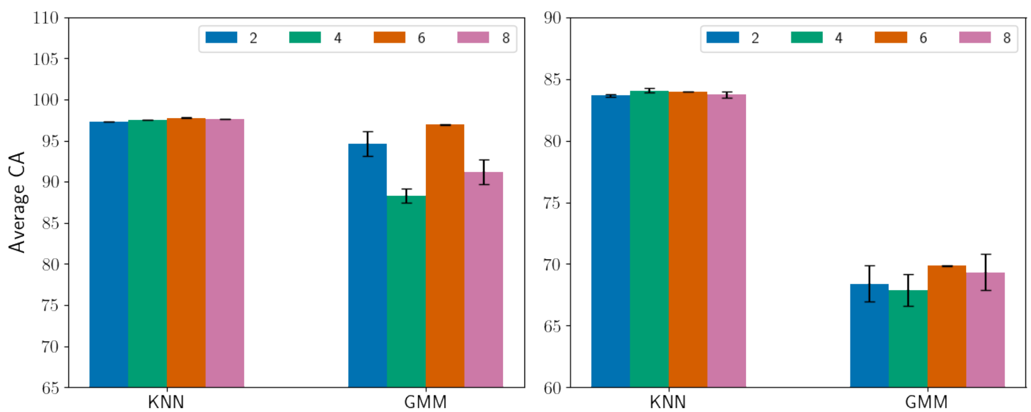


Figure 4: Ablation experiment across the number of augmentations considered during training of the SimVAE model using the MNIST (left) and FashionMNIST (right) datasets. Two, four, six and eight augmentations were considered. The average and standard deviation of the downstream classification accuracy using KNN and GMM probes are reported across three seeds. Batch size of 128 for all reported methods and number of augmentations.

### A.5.2 Image Generation

In this section, we explore and report the quality of images generated through SimVAE and all considered baselines through visualisations (for VAE-based approaches only) and quantitative measurements.

**Generated Images** Figure 5 report examples of randomly generated images for each digit class and clothing item using the SimVAE trained on MNIST and FashionMNIST respectively.

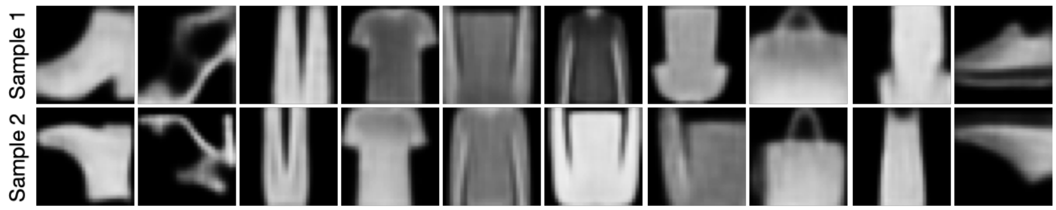


Figure 5: Conditional sampling for each one of the FashionMNIST clothing type using pre-trained SimVAE model

| | | RE | FID | NLL |
|---|---|---|---|---|
| Fashion | VAE | $4.4 \pm 0.1$ | $99.4 \pm 0.6$ | $5696.5 \pm 0.1$ |
| | $\beta$-VAE | $4.6 \pm 0.1$ | $99.9 \pm 0.7$ | $5696.7 \pm 0.1$ |
| | CR-VAE | $4.3 \pm 0.0$ | $98.7 \pm 0.0$ | $5696.7 \pm 0.0$ |
| | SimVAE | $\mathbf{3.4} \pm 0.1$ | $\mathbf{96.1} \pm 1.0$ | $5695.6 \pm 0.0$ |
| Celeb-A | VAE | $56.6 \pm 0.2$ | $162.9 \pm 2.8$ | $-$ |
| | $\beta$-VAE | $60.3 \pm 1.0$ | $163.8 \pm 2.3$ | $-$ |
| | CR-VAE | $57.4 \pm 0.1$ | $\mathbf{159.3} \pm 5.4$ | $-$ |
| | SimVAE | $\mathbf{35.3} \pm 0.2$ | $\mathbf{157.8} \pm 2.3$ | $-$ |
| CIFAR10 | VAE | $\mathbf{21.4} \pm 0.2$ | $365.4 \pm 3.3$ | $22330.8 \pm 0.2$ |
| | $\beta$-VAE | $22.3 \pm 0.2$ | $376.7 \pm 1.7$ | $22327.7 \pm 0.2$ |
| | CR-VAE | $22.5 \pm 0.0$ | $374.4 \pm 0.4$ | $22327.3 \pm 0.8$ |
| | SimVAE | $22.1 \pm 0.1$ | $\mathbf{349.9} \pm 2.1$ | $22327.3 \pm 0.2$ |

Table 8: Generation quality evaluation of all generative methods across three random seeds: (from left to right) mean squared reconstruction error (RE, $\downarrow$), fréchet inception distance (FID, $\downarrow$), negative log-likelihood (NLL,$\downarrow$)

**Generative quality** Table 8 reports the FID scores, reconstruction error and approximate negative log-likelihoods using 1000 importance-weighted samples for all generative baselines and SimVAE.