No Representation without Transformation

Giorgio Giannone [*] **Jonathan Masci** [*] **Christian Osendorfer** [*]

Abstract

We propose to extend Latent Variable Models with a simple idea: learn to encode not only samples but also transformations of such samples. This means that the latent space is not only populated by embeddings but also by higher order objects that map between these embeddings. We show how a hierarchical graphical model can be utilized to enforce desirable algebraic properties of such latent mappings. These mappings in turn structure the latent space and hence can have a core impact on downstream tasks that are solved in the latent space. We demonstrate this impact on a set of experiments and also show that the representation of these latent mappings reflects interpretable properties.

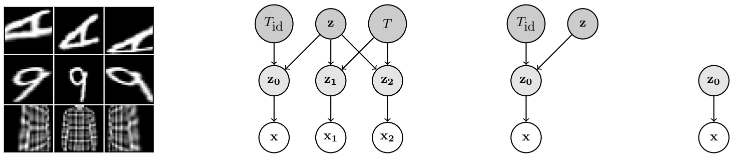
1 Introduction

A core appeal of unsupervised learning is its potentially important role for supervised [21, 2, 11] as well as reinforcement learning [23, 17, 7, 8]. It is supposed to make approaches in these two areas more sample efficient and also to improve their overall robustness as well as generalization characteristics [12]. From a probabilistic perspective, standard unsupervised learning aims at modeling a distribution over $\mathcal{X} \subseteq \mathbb{R}^D$, given a finite set of (i.i.d) observations $X = \{\mathbf{x_i} | \mathbf{x_i} \in \mathcal{X}\}_{i=0}^n$. Latent variable models are an approach for this task. For these type of models $p(\mathbf{x})$ is considered to be the result of a generative process comprising of a prior $\pi(\mathbf{z})$ over a latent variable $\mathbf{z}, \mathbf{z} \in \mathcal{Z} \subseteq \mathbb{R}^d, d \ll D$, and a conditional likelihood $p_\theta(\mathbf{x}|\mathbf{z})$. While there is an impressive amount of progress and new ideas in the area of latent variable models over the last couple of years [6, 13, 3, 25, 1, 4, 19], it is still difficult to make such low-dimensional structures both theoretically as well as practically tangible. Apart from a set of i.i.d samples what else is there that could describe such an underlying structure?

Structural properties of a set $\mathcal{X}$ can often be characterized through mappings between elements of $\mathcal{X}$, in general k-ary functions $f : \mathcal{X} \times \mathcal{X} \times \ldots \mathcal{X} \to \mathcal{X}$. If one accepts that such higher-order objects are expressive ways to represent various kinds of structural properties of $\mathcal{X}$ than it is a reasonable assumption that these objects should also be considered in the latent space of latent variable models. Mathematically speaking one is interested in *morphisms* between $\mathcal{X}$ and $\mathcal{Z}$. To the best of our knowledge such an idea has not been considered so far for latent variable models (see our remarks concerning related work in Section 4 about models for sequential data). In the following we will describe a possible method to integrate *latent mappings* into a specific latent variable model, derive a training algorithm for this new approach and discuss practical aspects to successfully realize this approach, including empirical results on some widely used datasets. In order to avoid getting lost in the fine details of mathematical complexities we will limit our further discussion of mappings to simple *Endomorphisms*, i.e. functions f of the form $f : \mathcal{X} \to \mathcal{X}$. These will be denoted *transformations* in the rest of the text. Mathematically, we assume that transformations over $\mathcal{X}$ form a group G under function composition $\circ$. That is, there exists an identity transformation $f_{\text{id}}(\mathbf{x}) = \mathbf{x}$, $\mathbf{x} \in \mathcal{X}$, function composition is associative and every $f \in G$ has an inverse $f^{-1} \in G$ such that $f \circ f^{-1} = f_{\text{id}}$. Clearly, for many real-world datasets the last two properties often are not strictly fulfilled. Yet, having such a rich structure simplifies the task to come up with a practical realization of the general idea described previously. In Section 4 we will briefly describe how these assumptions can be relaxed in a straight forward manner.

[*] NNAISENSE, `{firstname}@nnaisense.com`

Workshop on Perception as Generative Reasoning, NeurIPS 2019, Vancouver, Canada.



(a) Samples (b) $\mathcal{T}_{\text{VAE}}$ (c) VAE-id (d) VAE

Figure 1: (a) Examples (per row) for sampled triplets: The image in the middle is transformed with a random transformation (left image) and its inverse (right). (b) Our model. A latent sample $\mathbf{z}$ is transformed into two views, $\mathbf{z_1}$ and $\mathbf{z_2}$. The identity transformation maps $\mathbf{z}$ to $\mathbf{z_0}$. These latent representations are then mapped in the usual way to the observation space. (c) If no latent transformation exists, the suggested model collapses to a standard VAE. (d) A standard VAE.

2 Approach

It is not immediately obvious how transformations can be integrated into a latent variable model. One question is representing such transformations. In observation space it may be self-evident to use a representation of transformations that is associated with the domain of $\mathcal{X}$. For example, for images one may be tempted to represent transformations as affine matrices. But this could lead to models that are tied to specific data domains. Additionally there are transformations that can not be captured in a parametric way in the observation space, e.g. mapping an image to its segmented version or mapping an image to a sub-sampled version represented through a graph structure [2]. A more flexible approach is to represent transformations in observation space in an *implicit* way through tuples of samples and their respectively transformed versions. Because we assume that transformations have always inverses, we will represent a transformation f on $\mathcal{X}$ through a triplet: some sample $\mathbf{x} \in \mathcal{X}$, its transformed version $f(\mathbf{x})$ and the result of the inverse transformation $f^{-1}(\mathbf{x})$ (see Fig. 1(a)). What about the latent space $\mathcal{Z}$? From a probabilistic perspective, the latent variable $\mathbf{z}$ is a random variable, so the associated latent transformations should be represented by a distribution over functions, i.e. stochastic processes. An expressive way to represent these is by a learnable function parameterized through a random variable [5].

Model. Our model is an extension of the Variational Autoencoder (VAE) framework [16, 22]. In fact, using the ideas presented in the previous paragraphs, our suggestion resembles a hierarchical variant of a VAE, albeit with specific semantics of the hierarchy. Let $\mathbf{x} \in \mathcal{X}$ be some observed sample and let f be some arbitrarily chosen transformation on $\mathcal{X}$. Then $\mathbf{x_1} = f(\mathbf{x})$ and $\mathbf{x_2} = f^{-1}(\mathbf{x})$. In Fig. 1(b) we depict a graphical model that describes the generative process for the triplet $(\mathbf{x}, \mathbf{x_1}, \mathbf{x_2})$ utilizing a latent transformation represented by the random variable T. We denote this model the *transformation-aware* VAE, or $\mathcal{T}_{\text{VAE}}$ for short. Eq. 1 describes the equivalent probabilistic factorization of the joint distribution $p(\mathbf{x}, \mathbf{x_1}, \mathbf{x_2}, \mathbf{z}, \mathbf{z_0}, \mathbf{z_1}, \mathbf{z_2}, T)$:

$$p_\theta(\mathbf{x}|\mathbf{z_0})p_\theta(\mathbf{x_1}|\mathbf{z_1})p_\theta(\mathbf{x_2}|\mathbf{z_2})r_\chi(\mathbf{z_0}|\mathbf{z}, T_{\text{id}})r_\chi(\mathbf{z_1}|\mathbf{z}, T)r_\chi(\mathbf{z_2}|\mathbf{z}, T^{-1})\pi(\mathbf{z})\pi(T) \tag{1}$$

The proposed model is build in a way that fulfills the previously presented characteristics of transformations also for the latent space $\mathcal{Z}$: we postulate the existence of an identity mapping (T_{id}) and we enforce that every transformation T has an inverse T^{-1}. What this notation means is that for some given T the transformation $r_\chi(\cdot, T)$ has the uniquely determined inverse transformation $r_\chi(\cdot, T^{-1})$, i.e. T^{-1} is used as a short-hand notation here. Because T is a random variable, it has, like $\mathbf{z}$, a prior distribution. Note that the model collapses to a standard VAE if the set of transformations is empty – in this case the only admissible transformation is T_{id}, see Fig. 1(c).

In order to arrive at a tractable training procedure for this model through a lower bound on $\log p(\mathbf{x}, \mathbf{x_1}, \mathbf{x_2})$, we approximate the posterior distribution as:

$$q(T, \mathbf{z}, \mathbf{z_0}, \mathbf{z_1}, \mathbf{z_2}|\mathbf{x}, \mathbf{x_1}, \mathbf{x_2}) \approx q_\xi(T|\mathbf{z_1}, \mathbf{z_2})q_\psi(\mathbf{z}|\mathbf{z_1}, \mathbf{z_2})q_\phi(\mathbf{z_0}|\mathbf{x})q_\phi(\mathbf{z_1}|\mathbf{x_1})q_\phi(\mathbf{z_2}|\mathbf{x_2}) \tag{2}$$

[2] These are examples of transformations that do not have a well-defined inverse.

Because it is not clear how to model $q_\xi(T)$ as a parametric distribution, we decided to utilize an implicit distribution for it. This means we can derive a lower bound as follows:

$$
\begin{aligned}
\log p(\mathbf{x}, \mathbf{x_1}, \mathbf{x_2}) \geq\ & \mathbb{E}_{q_\phi(\mathbf{z_0}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z_0})\right] - \mathbb{E}_{q_\phi(\mathbf{z_0}|\mathbf{x})}\left[\mathcal{D}\left[q_\phi(\mathbf{z_0}|\mathbf{x}), r_\chi(\mathbf{z_0}|\mathbf{z}, T_{\text{id}})\right]\right] \\
& + \mathbb{E}_{q_\phi(\mathbf{z_1}|\mathbf{x_1})}\left[\log p_\theta(\mathbf{x_1}|\mathbf{z_1})\right] - \mathcal{D}\left[q_\phi(\mathbf{z_1}|\mathbf{x_1}), r_\chi(\mathbf{z_1}|\mathbf{z}, T)\right] \\
& + \mathbb{E}_{q_\phi(\mathbf{z_2}|\mathbf{x_2})}\left[\log p_\theta(\mathbf{x_2}|\mathbf{z_2})\right] - \mathcal{D}\left[q_\phi(\mathbf{z_2}|\mathbf{x_2}), r_\chi(\mathbf{z_2}|\mathbf{z}, T^{-1})\right] \\
& - \mathcal{D}\left[q_\psi(\mathbf{z}|\mathbf{z_1}, \mathbf{z_2}), \pi(\mathbf{z})\right]
\end{aligned}
\tag{3}
$$

where $\mathcal{D}\left[x, y\right] = \mathbb{E}_{q_\psi(\mathbf{z}|\mathbf{z_1},\mathbf{z_2})q_\phi(\mathbf{z_1}|\mathbf{x_1})q_\phi(\mathbf{z_2}|\mathbf{x_2})}\left[\log x - \log y\right]$ and T is computed by $q_\xi(T)$. While the algebraic properties of the latent mapping allowed us to derive the above model, several details are missing for running experiments. In order to assess the overall idea we decided to have the simplest functional form for the latent transformation $r_\chi(\mathbf{z_1}|\mathbf{z}, T)$: $r_\chi(\mathbf{z_1}|\mathbf{z}, T) \equiv \mathbf{z} + T$ and its inverse as $\mathbf{z} - T$. This implies that $T_{\text{id}} \equiv \mathbf{0}$. For the posterior $q_\psi(\mathbf{z}|\mathbf{z_1}, \mathbf{z_2})$ we chose a Gaussian whose mean and s.d. are parameterized by a neural network $h_\psi(\mathbf{z_1}, \mathbf{z_2})$. The implicit distribution $q_\xi(T|\mathbf{z_1}, \mathbf{z_2})$ is represented by a neural network $h_\xi(\mathbf{z_1}, \mathbf{z_2})$. The goal of $\mathcal{D}[\cdot, \cdot]$ is to ensure coherence between the inference and generative paths. Inspired by [20, 28] we choose the maximum mean discrepancy criterion [9] to implement $\mathcal{D}[\cdot, \cdot]$. While the resulting standard VAE posterior $q_\phi(\mathbf{z_0}|\mathbf{x})$ produced acceptable results with respect to the marginal likelihoods, the posterior inference over $\mathbf{z}$, i.e. $q_\psi(\mathbf{z}|\mathbf{z_1}, \mathbf{z_2})$ failed with respect to the implications of the $\mathcal{T}_{\text{VAE}}$ semantics. More specifically, $\log p_\theta(\mathbf{x}|\mathbf{z})$ was very bad. As a solution to this issue we introduced the *indirect* coherence term $\mathcal{F}_{\mathbf{x}|\mathbf{z}} \equiv -\mathbb{E}_{q_\psi(\mathbf{z}|\mathbf{z_1},\mathbf{z_2})q_\phi(\mathbf{z_1}|\mathbf{x_1})q_\phi(\mathbf{z_2}|\mathbf{x_2})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] + \mathcal{D}_{\text{KL}}\left[q_\phi(\mathbf{z_0}), \pi(\mathbf{z_0})\right]$, substituting $\mathbb{E}_{q_\phi(\mathbf{z_0}|\mathbf{x})}\left[\mathcal{D}\left[q_\phi(\mathbf{z_0}|\mathbf{x}), r_\chi(\mathbf{z_0}|\mathbf{z}, T_{\text{id}})\right]\right]$ in Eq. 3. $\mathcal{F}_{\mathbf{x}|\mathbf{z}}$ explicitly enforces the implied semantic for $\mathbf{z}$ and $\mathbf{z_0}$ (i.e. both are equal under the VAE model).

3 Experiments

The experiments in this section demonstrate that (i) the resulting $\mathcal{T}_{\text{VAE}}$ model is a good generative model for a given dataset (ii) the latent structure induced by the $\mathcal{T}_{\text{VAE}}$ is helpful for downstream tasks and (iii) the variable T encodes information that resembles transformations in observation space. For the first two aspects we utilize a standard VAE as a baseline. Because $\mathcal{T}_{\text{VAE}}$ is trained on an augmented dataset we also introduce VAE+ as another baseline – a VAE trained on an augmented dataset constructed through the employed transformations during training of the $\mathcal{T}_{\text{VAE}}$. These three models have the same architecture for the core encoder as well as the core decoder. We run experiments on MNIST [18], Fashion-MNIST [27] and AffNIST.

Generative Modeling. On MNIST, the $\mathcal{T}_{\text{VAE}}$ achieves a marginal likelihood of ~ 92 nats on the test set (using a simple convolutional encoder and decoder, see the Appendix). A VAE achieves ~ 93 nats and VAE+ ~ 94 nats. Similar qualitative behaviour is observed over various other datasets and architectures. We therefore conclude that the $\mathcal{T}_{\text{VAE}}$ is able to learn a reasonable generative model for a given dataset.

Latent Space Structure. The core motivation for introducing a latent transformation object was the hypothesis that the structure of the latent space can be shaped better. Here, we test this hypothesis in two ways. If two datasets are similar in the sense that they have a substantial overlap in their respective sets of transformations, the latent embedding learned (by a $\mathcal{T}_{\text{VAE}}$) on the one dataset should also be useful for the other. We test *usefulness* of an embedding through a KNN classifier on these. We run experiments for the dataset pairs MNIST-AffNIST and FashionMNIST-AffNIST. That means, we train a $\mathcal{T}_{\text{VAE}}$ on MNIST and F-MNIST respectively and then evaluate the embedding on AffNIST. For that a KNN classifier is trained on the training set embedding of AffNIST and evaluated on its test set embeddings.

Because MNIST and AffNIST are relatively similar with respect to their content we would expect that at least VAE+ should perform well, too In the case of F-MNIST/AffNIST however the structured latent space of a $\mathcal{T}_{\text{VAE}}$ should be much more informative. This is what Table 1 shows (M-Aff and F-Aff columns). For the standard case of no domain/content shift (the M-M and F-F columns in Table 1) all the models perform comparably, corroborating our hypothesis that the performance gain

Model	z_{dim}	M-Aff	M-M	F-Aff	F-F
VAE	10	55.9	96.2	32.8	81.7
VAE+	10	70.2	96.7	36.3	82.2
$\mathcal{T}_{\text{VAE}}$	10	**73.5**	96.6	**55.1**	84.0
VAE	100	66.3	96.2	43.1	82.6
VAE+	100	79.2	97.4	43.3	82.7
$\mathcal{T}_{\text{VAE}}$	100	**90.6**	98.1	**81.2**	86.9

Table 1: MNIST, Fashion-MNIST and AffNIST test set accuracy using the latent mean μ_z and a KNN classifier. Models trained on MNIST (M) and Fashion-MNIST (F).

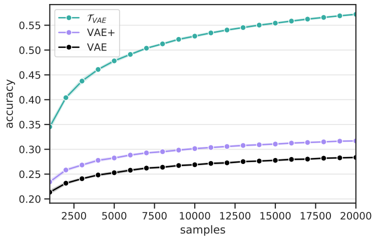


Figure 2: Classification accuracy on AffNIST test set over 20 runs using μ_z. Models trained on Fashion-MNIST.

for $\mathcal{T}_{\text{VAE}}$ in the presence of a domain shift is due to a better way to organize information in the latent space, and not unfit baselines. Similarly, in the setting of a domain shift (e.g. embedding trained on F-MNIST, used for AffNIST), the structure induced by $\mathcal{T}_{\text{VAE}}$ should be very helpful when only a small set of labelled samples for the KNN are available. Fig. 2 documents this effect (which is aligned with the previous experiment) and shows that $\mathcal{T}_{\text{VAE}}$ embeddings utilize additional samples much more efficiently.

Latent transformations. Finally we want to investigate what kind of transformation information is encoded in T. We take an arbitrary triplet $(\mathbf{x}, \mathbf{x_1}, \mathbf{x_2})$ constructed according to our data generation scheme and infer T. We then take a random data element, sample from its posterior $q_\phi(\mathbf{z}|\mathbf{x})$ apply $r_\chi(\mathbf{z_1}|\mathbf{z}, T)$ and $r_\chi(\mathbf{z_2}|\mathbf{z}, T^{-1})$ and decode $\mathbf{z_1}$ and $\mathbf{z_2}$.

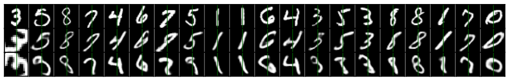


Figure 3: The transformation T is extracted from the triplet in the first column. It is then applied to the latent embedding of the top image in the subsequent columns. The green vertical lines in every image should support inspecting for right/left rotation, which is the (invertible) transformation used in the first column.

Figure 3 shows the qualitative result of this experiment. The triplet for extracting T is the first column: The top image shows an unmodified digit 3 and the two images below show a right/left rotation of it. Note that both images also show the effects of a cropping transformation. This means that the two overall transformations are not inverse to each other! Nevertheless, one would expect that the inferred T should mostly obtain some representation of the rotation transformation. The following columns then show the result of applying T (second row) and T^{-1}(third row) to the latent embedding of the first row. We see that most examples are rotated to the right and left respectively, however content and style are mostly unmodified.

4 Discussion and Outlook

Recurrent neural networks are the premier example when talking about mappings in latent spaces. Thinking about these implicitly available mappings lead to a broad family of (recurrent) state-space models [26, 14, 10]. Differently from these approaches, our work considers mappings as explicit objects – what properties these objects should have and how these properties can be enforced. The goal is to induce a structure on the latent space of VAEs such that embeddings in this space are more useful for downstream tasks (differently from most approaches this structure is not seeking to improve the generative modeling per se). Clearly, this structure is heavily determined by $r_\chi(\cdot, T)$. A more powerful functional form then the one presented in this work will lead to more interesting properties of the latent space. Preliminary results for utilizing a latent transformation function of the form $T\mathbf{z}$ (i.e. a matrix-vector product) look very promising. We also started to represent $\mathbf{z}$ itself as a matrix [24]. Additionally, one might argue that only considering invertible transformations is too strict. It turns out that this requirement can be relaxed and the model in Fig. 1 can be suitably adapted to handle an arbitrary number of transformations (i.e. views of a sample) where no restrictions are put onto these transformations. Obviously, posterior inference in this case is much more challenging. In preliminary experiments we successfully trained a $\mathcal{T}_{\text{VAE}}$ that utilized a Super-pixel transformation.

References

[1] Achille, A. and Soatto, S. (2018). Emergence of invariance and disentanglement in deep representations. *The Journal of Machine Learning Research*, 19(1):1947–1980.

[2] Dwibedi, D., Aytar, Y., Tompson, J., Sermanet, P., and Zisserman, A. (2019). Temporal cycle-consistency learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1801–1810.

[3] Edwards, H. and Storkey, A. (2016). Towards a neural statistician. *arXiv preprint arXiv:1606.02185*.

[4] Eslami, S. A., Rezende, D. J., Besse, F., Viola, F., Morcos, A. S., Garnelo, M., Ruderman, A., Rusu, A. A., Danihelka, I., Gregor, K., et al. (2018). Neural scene representation and rendering. *Science*, 360(6394):1204–1210.

[5] Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S., and Teh, Y. W. (2018). Neural processes. *arXiv preprint arXiv:1807.01622*.

[6] Gregor, K., Danihelka, I., Graves, A., Rezende, D. J., and Wierstra, D. (2015). Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*.

[7] Gregor, K., Papamakarios, G., Besse, F., Buesing, L., and Weber, T. (2018). Temporal difference variational auto-encoder. *arXiv preprint arXiv:1806.03107*.

[8] Gregor, K., Rezende, D. J., Besse, F., Wu, Y., Merzic, H., and Oord, A. v. d. (2019). Shaping belief states with generative environment models for rl. *arXiv preprint arXiv:1906.09237*.

[9] Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., and Smola, A. J. (2007). A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pages 513–520.

[10] Guo, Z. D., Azar, M. G., Piot, B., Pires, B. A., Pohlen, T., and Munos, R. (2018). Neural predictive belief representations. *arXiv preprint arXiv:1811.06407*.

[11] Han, K., Vedaldi, A., and Zisserman, A. (2019). Learning to discover novel visual categories via deep transfer clustering. *arXiv preprint arXiv:1908.09884*.

[12] Higgins, I., Amos, D., Pfau, D., Racaniere, S., Matthey, L., Rezende, D., and Lerchner, A. (2018). Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230*.

[13] Hoffman, M. D. and Johnson, M. J. (2016). Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in Advances in Approximate Bayesian Inference, NIPS*.

[14] Karl, M., Soelch, M., Bayer, J., and van der Smagt, P. (2016). Deep variational bayes filters: Unsupervised learning of state space models from raw data. *arXiv preprint arXiv:1605.06432*.

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

[16] Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

[17] Kulkarni, T., Gupta, A., Ionescu, C., Borgeaud, S., Reynolds, M., Zisserman, A., and Mnih, V. (2019). Unsupervised learning of object keypoints for perception and control. *arXiv preprint arXiv:1906.11883*.

[18] LeCun, Y. (1998). The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

[19] Locatello, F., Bauer, S., Lucic, M., Gelly, S., Schölkopf, B., and Bachem, O. (2018). Challenging common assumptions in the unsupervised learning of disentangled representations. *arXiv preprint arXiv:1811.12359*.

[20] Mohamed, S. and Lakshminarayanan, B. (2016). Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*.

[21] Oord, A. v. d., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

[22] Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*.

[23] Sharma, A., Gu, S., Levine, S., Kumar, V., and Hausman, K. (2019). Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*.

[24] Sutskever, I. and Hinton, G. E. (2009). Using matrices to model symbolic relationship. In *Advances in neural information processing systems*, pages 1593–1600.

[25] van den Oord, A., Vinyals, O., et al. (2017). Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pages 6306–6315.

[26] Watter, M., Springenberg, J., Boedecker, J., and Riedmiller, M. (2015). Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in neural information processing systems*, pages 2746–2754.

[27] Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

[28] Zhao, S., Song, J., and Ermon, S. (2017). Infovae: Information maximizing variational autoencoders. *arXiv preprint arXiv:1706.02262*.

A Visualizations

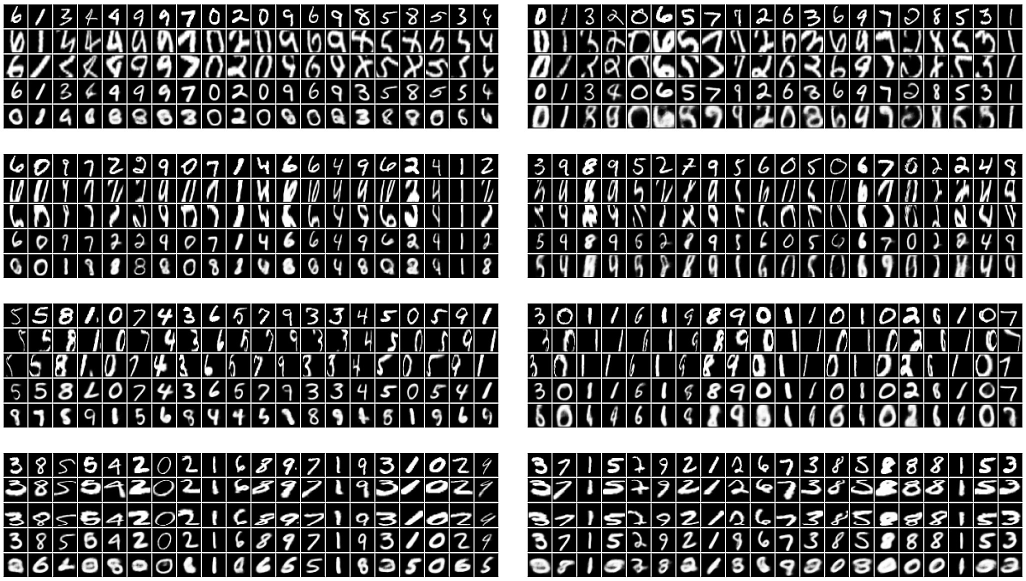


Figure 4: Interpolating the latent space on MNIST. First row (per image): original sample. Second and third rows: views with applied transformations (resulting in z_1 and z_2). Fourth and Fifth rows: $\mathcal{T}_{\text{VAE}}$ and VAE (left column)/ VAE+ (right column) interpolation. Interpolation is done by decoding $\frac{z_1+z_2}{2}$ – this is clearly helpful for the utilized function $r_{\mathcal{X}}(z,T) \equiv z+T$ from $\mathcal{T}_{\text{VAE}}$.

B Technical Details

We use mini-batches of size 100 and train the models for 200 epochs. We set $\alpha = 10^{-4}$ and use Adam [15] for training. Every 50 epochs α is halved. We use 4x4 filters for all the encoder and decoder layers and stride 2 for all the encoder convolutions and the last two transposed convolutions in the decoder [3].

The encoder parametrizes the moments of a multivariate Gaussian distribution with diagonal covariance matrix. The decoder parametrizes the moments of a Bernoulli distribution. We train the models with a binary cross-entropy loss and minimizing a KL divergence with standard normal as prior. We train models with latent dimensions $z_d \in [10, 100]$.

For MNIST and Fashion-MNIST :

$q_\phi(\mathbf{z}|\mathbf{x})$:

$$\begin{aligned} x \in \mathcal{R}^{28\text{x}28} &\rightarrow \text{Conv}_{32} \rightarrow \text{ReLU} \\ &\rightarrow \text{Conv}_{32} \rightarrow \text{ReLU} \\ &\rightarrow \text{Conv}_{64} \rightarrow \text{ReLU} \\ &\rightarrow \text{Conv}_{128} \rightarrow \text{ReLU} \\ &\rightarrow \text{Conv}_{2z_d} \end{aligned}$$

[3] We write for all the architectures Module$_k$ where k is the number of filters in output for convolutions and transpose convolutions and number of units in output for fully connected layers.

$p_\theta(\mathbf{x}|\mathbf{z})$:

$$\begin{aligned} z \in \mathcal{R}^{z_d} &\rightarrow \text{Conv}_{128} \rightarrow \text{ReLU} \\ &\rightarrow \text{ConvT}_{64} \rightarrow \text{ReLU} \\ &\rightarrow \text{ConvT}_{32} \rightarrow \text{ReLU} \\ &\rightarrow \text{ConvT}_{32} \rightarrow \text{ReLU} \\ &\rightarrow \text{ConvT}_{1} \end{aligned}$$

$q_\psi(\mathbf{z}|\mathbf{z_1}, \mathbf{z_2})$:

$$\begin{aligned} z \in \mathcal{R}^{2z_d} &\rightarrow \text{FC}_{1000} \rightarrow \text{ReLU} \\ &\rightarrow \text{FC}_{1000} \rightarrow \text{ReLU} \\ &\rightarrow \text{FC}_{2z_d} \end{aligned}$$

$T_\xi(\mathbf{z_1}, \mathbf{z_2})$:

$$\begin{aligned} z \in \mathcal{R}^{2z_d} &\rightarrow \text{FC}_{1000} \rightarrow \text{ReLU} \\ &\rightarrow \text{FC}_{1000} \rightarrow \text{ReLU} \\ &\rightarrow \text{FC}_{z_d}. \end{aligned}$$