Local Disentanglement in Variational Auto-Encoders Using Jacobian L_1 Regularization

Anonymous Author(s)
Affiliation
Address
email

Abstract

There have been many recent advances in representation learning; however, unsupervised representation learning can still struggle with model identification issues. Variational Auto-Encoders (VAEs) and their extensions such as β-VAEs have been shown to locally align latent variables with PCA directions, which can help to improve model disentanglement under some conditions. Borrowing inspiration from Independent Component Analysis (ICA) and sparse coding, we propose applying an L_1 loss to the VAE's generative Jacobian during training to encourage local latent variable alignment with independent factors of variation in the data. We demonstrate our results on a variety of datasets, giving qualitative and quantitative results using information theoretic and modularity measures that show our added L_1 cost encourages local axis alignment of the latent representation with individual factors of variation.

1 Introduction

Unsupervised representation learning on images involves taking a collection of image data from the world and figuring out how to organize and find patterns in the data without additional information about how the images were generated. The motivation for representation learning comes from the fact that images are often generated from lower-dimensional ground-truth factors of variation in the world. The ideal representation learning would compress the high-dimensional image data from the world into a lower-dimensional representation that contains the relevant information about the ground-truth factors of variation that generated the image.

Inferring a good latent representation from a dataset is a difficult problem and is generally underspecified in algorithms. This underspecification is called the "model identification" problem, and one example is the fact that representation learning algorithms often struggle to precise the correct orientation of a latent space. That is, optimization criteria used to learn a representation function might be equally well satisfied by a similar representation that is the same as the original representation except that it rotates the latent space an arbitrary amount. There have been many recent advances in representation learning; however, unsupervised representation learning can still struggle with model identification issues.

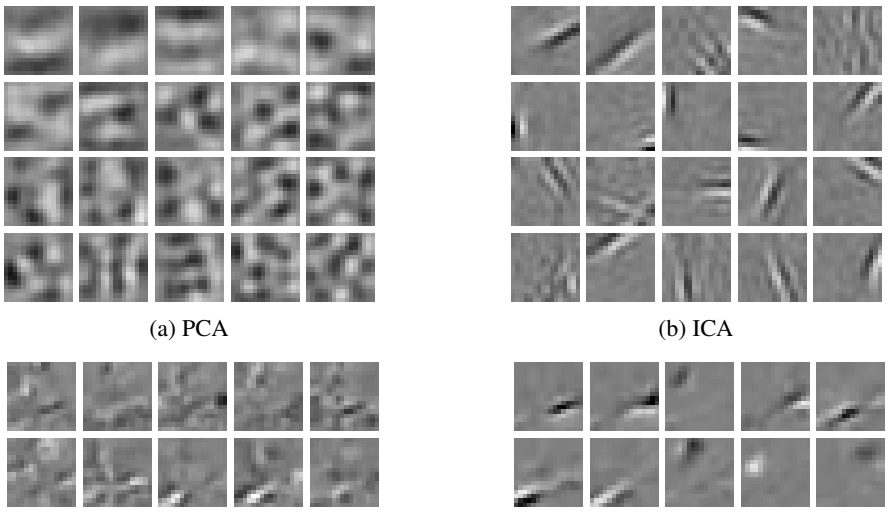
It has been shown [1, 2] that Variational Auto-Encoders (VAEs) [3] and their common extension, the β-VAE [4], are able to solve the rotational part of the model identification issue by tending to ensure that the generation function's Jacobian matrix has orthogonal columns. Kumar and Poole [2] draw the parallel between learning orthogonal columns and linear Principal Component Analysis (PCA). However, we note that learning algorithms that resolve rotational identification issues through methods related to PCA will still suffer from an identifiability issue concerning rotations that mix directions whose eigenvectors are equal.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Taking our inspiration from Independent Component Analysis (ICA) [5] and sparse coding [6], we look to use sparsity as a way to solve this rotational identifiability concern and to encourage the representation learning algorithm to isolate independent factors of variation. We extend the VAE objective with an explicit L_1 regularization cost on the Jacobian of the generator function in order to encourage sparsity and resolve the identification issue of rotations among similar eigenvalues in the latent space.

As further motivation, we consider results of sparse linear coding of images. Olshausen and Field [6] showed how sparsity in linear generative image models can generate local receptive fields in the basis of the linear latent space. This motivated us attempt to encourage similar local receptive fields in VAE-style representation learning by encouraging sparsity in the Jacobian matrix of the generative model. We call the proposed model trained with this additional regularization a Jacobian L_1 Regularized Variational Auto-Encoder (JL1-VAE).

After training our proposed JL1-VAE model on natural image data collected by Olshausen and Field [6], we observe that latent directions show more localized effects on the resulting image. This is more similar to the effect of individual directions in the ICA model, and different from the more global effect seen for PCA and β-VAE models.



(a) PCA (b) ICA

(c) β-VAE (d) JL1-VAE (ours)

Figure 1: Example columns from generative Jacobian matrices for different modeling techniques on natural image data collected in [6]. ICA used 100 latent dimensions, of which 20 samples are shown. β-VAE used $\beta = 0.01$, JL1-VAE used $\beta = 0.01$, $\gamma = 0.01$, each on ten latent dimensions.

We apply our novel JL1-VAE framework to a variety of datasets, giving qualitative and quantitative results showing that our added L_1 cost encourages local axis alignment of the latent representation with individual factors of variation.

2 Background

We present a brief overview of VAEs and introduce the notation we will use throughout this paper. VAEs [3] train a model to generate a data distribution that approximately matches the distribution of unlabeled training data. For the VAEs we consider in this paper, generating a datapoint $\tilde{x} \in \mathbb{R}^n$ from a trained VAE consists of sampling a latent variable $z \in \mathbb{R}^l$ from a standard multivariate Gaussian distribution $p(z)$ and applying a generator function $g : \mathbb{R}^l \to \mathbb{R}^n$ to z to map the latent variable to a generated image $g(z)$. Around this generated image we assume a Bernoulli probability of similar images $\tilde{x} \sim p(\tilde{x}; g(z))$ with the generated image $g(z)$ as its mean.

To train the VAE we also define a multi-variate Gaussian embedding distribution $q(z|x)$ for each training image x with mean $h(x)$ (using a learned embedding function $h : \mathbb{R}^n \to \mathbb{R}^l$) and diagonal covariance $\Sigma_{z|x}(x)$ (using a learned covariance function $\Sigma_{z|x} : \mathbb{R}^n \to \mathbb{R}^l$ that returns the diagonal

elements). This embedding distribution $q(z|x)$ is motivated by a desire to approximate the posterior distribution $p(z|x)$.

The objective during training of the VAE is to maximize the Evidence Lower BOund (ELBO), which is defined as $\sum_x L(x)$, where, for each data point x,

$$L(x) = E_{z \sim q(z|x)} \left[p(x; g(z)) \right] - \mathrm{KL} \left(q(z|x) \| p(z) \right) \tag{1}$$

The ELBO is a lower bound on $\log(p(x)) = \log(\int_z p(x|z)p(z)dz)$, which is the likelihood of the data point given our model. Thus, maximizing the ELBO is a proxy for maximum likelihood estimation. The β-VAE [4] is an extension to the VAE that multiplies the second term in the ELBO by an adjustable hyperparameter β.

3 Related Work

There are several areas of related work we wish to call to the reader's attention. We note previous work in the analysis of the disentanglement properties of β-VAEs. We delve into previous uses of the term "sparse VAE," as there are (at least) two other common and well-studied meanings of "sparse VAE," which we disambiguate from the type of sparsity we study in this work. We note previous VAE work inspired by ICA. We discuss architectural choices that have been shown to improve disentanglement. Finally, we discuss modifications to the VAE objective that have previously been studied to improve disentanglement.

Disentanglement of β-VAEs Mathieu et al. [7] and Rolinek et al. [1] show that restricting the posterior covariance to diagonal (called the mean-field assumption) breaks the rotational symmetry of β-VAEs. Rolinek et al. [1] further show that this encourages the columns of the generator Jacobian to be orthogonal, relating local β-VAE latent directions with PCA decomposition. In our work, we take this a step further, explicitly regularizing the sparsity of the generator Jacobian, to break rotational symmetry between directions with equal eigenvalues.

Sparse VAEs: Sparsity in VAE codes Some previous work involving L_1 regularization and VAEs may use the term "sparse VAEs" to refer to sparsity in the *latent values* taken on by the latent codes themselves. That is, these works attempt a minimization of something like $\|z\|_1$. This meaning is studied in [8–10]. When we use sparsity in this work, however, we are not concerned with the values of the latents z, but rather how small axis-aligned changes in the latent values affect the output. That is, we are concerned with sparsity of $J_g(z)$ not z. Our associated cost is $\|J_g(z)\|_1$. Our use of "sparsity" has to do with local disentanglement, rather than sparsity in latent values, which is a type of global disentanglement.

Sparse VAEs: Sparsity in Network Weights Likewise, other work involving L_1 regularization and VAEs (and neural networks more broadly) uses "sparsity" to refer to the desire to make many network weights 0, with the motivation of reducing the size of the stored neural network architecture. This meaning is seen in [11, 12]. In this work, by contrast, we are interested in sparsity in the generator Jacobian, not in the network weights. We note the distinction between sparsity of individual network weights and sparsity in the Jacobian of the overall function. Multiplying sparse matrices does not necessarily result in sparse matrices, and non-linear activations can allow a sparse Jacobian even if the network weights themselves are not particularly sparse. Thus the $\|J_g(z)\|_1$ cost we study in this work is not what is referred to in studies of sparsity of neural networks, which consider regularizations like the L1 cost over network weights.

ICA within the VAE Literature Independent Component Analysis (ICA) [5] is often mentioned in the VAE literature in relation to the role ICA has played in the theory of identifiability and disentanglement of representations [2, 7, 1, 13, 14]. Stuehmer et al. [15] propose using a structured, rotationally asymmetric prior to encourage disentanglement in the embedding. This, and other approaches that attempt to globally match the embedding distribution to a desired shape, are very different from the local, Jacobian-based approach we take in this paper. Khemakhem et al. [16] relate nonlinear ICA with VAEs in the case where the data has an additionally observed variable, and they give a proof that in that case their model is identifiable and correctly disentangles the ground-truth factors of variation. We focus on fully unsupervised training data and assume that we are not given

access to any data labels. We are not aware of any prior work applying a sparsity cost to the generator Jacobian, which is the inspiration we take from ICA.

Architectures shown to improve disentanglement Previous work has shown impressive results in modifying the network architecture in order to explicitly represent multiple objects by, for example, learning object masks [17], or by modifying how the latent variable is read in to the generative model architecture [18]. Ainsworth et al. [19] and Khan et al. [20] both construct model architectures that explicitly encourage sparsity in the generative network weights. In this work, we focus on how we can regularize the objective function to improve disentanglement, rather than studying how different network architectures can improve disentanglement.

Modification of VAE prior Previous work has also investigated modifications to the unit Gaussian prior commonly used in VAEs. Tomczak and Welling [21] use a learnable Gaussian mixture prior. Stuehmer et al. [15] use a generalized Gaussian distribution (that is not rotationally invariant) for the prior. Bauer and Mnih [22] use rejection sampling to form a more complicated, non-rotationally-invariant prior. Davidson et al. [23] and Perez Rey et al. [24] even modify the prior to lie on non-Euclidean surfaces. Kim and Mnih [13] do not explicitly enforce a prior distribution, but rather use a regularization term to encourage the prior distribution $q(z)$ to be a factorized distribution. While these approaches pressure the entire embedded distribution to have certain properties, we are instead focused on how to modify the learning objective to give *local* bias toward disentanglement, rather than using more global methods based on the overall distribution of the embedded dataset.

Regularization of VAEs Several previous works explicitly or implicitly use L_2 normalization of the generator Jacobian [2, 25–27]. Chen et al. [28] regularize by $\|J_g^\top(z)J_g(z) - c\mathbb{1}\|_2$ for some constant c. We are not aware of previous investigations of $\|J_g(z)\|_1$ regularization for VAEs.

4 Model Architecture and Loss Calculation

Our L_1 regularization of the generator Jacobian is partially motivated by the already-present implicit L_2 regularization of the generator Jacobian in $beta$-VAEs. Kumar and Poole [2] approximate the ELBO of a β-VAE as:

$$L(x) \approx \log p(x|h(x)) - \beta \mathrm{KL}\left(q(z|x)\|p(z)\right) + \frac{1}{2}\mathrm{tr}\left(J_g(h(x))^\top H_{p_x}(g(h(x)))J_g(h(x))\Sigma_{z|x}(x)\right) \tag{2}$$

where h is the deterministic embedding function from $\mathbb{R}^n \to \mathbb{R}^l$, g is the deterministic generation function, and J_g is its Jacobian. $\Sigma_{z|x}(x)$ is the covariance matrix of the Gaussian posterior, and $H_{p_x}(g(h(x)))$ is the Hessian with respect to $g(z)$ of the generative probability $p(x; g(z))$ parameterized by $g(z)$ and evaluated at $g(h(x))$. For pixel-factorized generative probabilities, with diagonal Gaussian posteriors, $\Sigma_{z|x}(x)$ and $H_{p_x}(g(h(x)))$ are both diagonal. Thus, the regularization term $\frac{1}{2}\mathrm{tr}\left(J_g(h(x))^\top H_{p_x}(g(h(x)))J_g(h(x))\Sigma_{z|x}(x)\right)$ is a (weighted) L_2 regularization cost on the Jacobian $J_g(h(x))$, with columns weighted by $\Sigma_{z|x}(x)$ and rows weighted by $H_{p_x}(g(h(x)))$. This L_2 regularization cost encourages local alignment of the latent axes to the right singular vectors of J_g. In cases where the right singular vectors of J_g are not uniquely defined—that is, where two singular values are equal—then the VAE objective function will not locally prefer a particular rotation of the latent variables within the subspace spanned by those singular vectors. This issue, combined with the presence of an implicit L_2 loss on the generator Jacobian already, motivates our choice to add an L_1 regularization to that Jacobian.

In this work, we show how penalizing the sparsity of the Jacobian, that is, penalizing the L_1 norm of the Jacobian J_g, helps resolve this ambiguity in the latent space among directions with equal eigenvalues. We define a Jacobian L_1 Regularized Variational Auto-Encoder (JL1-VAE) as a VAE that is trained using the loss for a β-VAE augmented with an L_1 regularization of the elements of the Jacobian matrix of the data generation map from latent values to mean generated images. The regularization term is modulated by a hyperparameter γ.

Specifically, the loss for the JL1-VAE is the sum over all datapoints x of

$$L_{\mathrm{JL1}}(x) = E_{z\sim q(z|x)}\left[\log p(x|z) + \gamma\left|\left|J_g(z)\right|\right|_1\right] - \beta \mathrm{KL}\left(q(z|x)\|N(\mathbb{0}, \mathbb{1})\right) \tag{3}$$

Since we use a Gaussian posterior $q(z|x) = N\left(h(x), \Sigma_{z|x}(x)\right)$, we can use an explicit calculation for the KL-divergence. We estimate the expectation of $\log p(x|z)$ and of $\gamma \|J_g(z)\|_1$ using a single sample z from the distribution over which we are taking the expectation.

4.1 Jacobian Estimation

We estimate the full Jacobian matrix $J_g(z)$ using the finite difference method along each latent dimension. That is, for any given latent value z at which we wish to compute the Jacobian matrix, we generate a set of k data points $z_i = z + \epsilon e_i$, where ϵ is a small fixed value and e_i a unit vector in the ith latent direction. We then run the forward model on the batch of z_i to generate $g(z_i)$ and estimate the ith column of the Jacobian matrix as $(g(z) - g(z_i))/\epsilon$ This Jacobian matrix estimate is itself backward differentiable using the standard backward differentiation provided by PyTorch [29], and can be directly used in our JL1-VAE loss. This leads to a runtime that scales roughly linearly with the number of latent variables in the VAE architecture.

4.2 Architecture

We use a convolutional architecture for our VAEs. In particular, our embedding architecture consists of convolutional layers followed by a shared fully connected layer with ReLU activations. This base model is shared between the mean and log variance embedding networks. Each embedding network then appends its own linear fully connected head to the shared model. We use a diagonal structure for the log variance estimates to reduce the number of parameters we need to estimate. We use a latent dimension of ten for all experiments, though we have seen similar results for other latent dimension sizes. The reconstruction architecture consists of fully connected layers followed by convolutional layers, using ReLU activations, with a final sigmoidal activation function. A full set of hyperparameter choices for each experiment can be found in the Appendix. When we compare JL1-VAE with other methods, we ensure consistent architecture choices.

5 Experiments

5.1 Datasets

To evaluate the ability of JL1-VAE to locally disentangle factors of variation, we apply it to a variety of datasets.

The first are natural images in grayscale taken by Olshausen and Field [6] and cropped to 16×16 pixel regions. We do not have labeled "ground-truth factors of variation" for this data, but we are able to provide qualitative results by inspecting the columns of the generator Jacobian matrix. This data was made publicly available without a specific license, so we analyze it under "fair use."

The second is a dataset of simulated 64×64 pixel grayscale images of three black dots on a white background, inspired by [30]. We re-implement their code to generate that dataset and modify it to label the ground-truth factors of variation used to generate an image, and we allow dots to have centers close enough that the dots overlap. We discretize the possible x,y coordinates of the center of each dot to 64 different values. We note that the generative map is not injective, as the dots are identical, so the same resulting image can be formed from multiple permutations of ground-truth factor values. There are $64^6 \sim 68.7$ billion different possible input latent combinations, from which we pre-generate a cache of 500,000 images on which we train. During evaluation, we generate new images at runtime based on the desired ground-truth factors of variation. Because the three dots are identical in this dataset, we expect that PCA-based model identification methods should not be able to disentangle individual dot motions. If a model were to disentangle individual dot motions into separate latent components, then we note, by symmetry, that we would expect identical eigenvalues of $J^T J$ for those components. However, methods that align to PCA directions suffer from model identifiability issues when presented with data with identical eigenvalues. Thus, we expect that β-VAEs will be unable to isolate individual dot motions on this dataset.

Finally, we also apply our approach to tiled images of a real robotic arm taken from the MPI3D-real dataset [31], licensed under Creative Commons Attribution 4.0 International License. For each data point, we downsample four random images of the robot arm holding a large blue square in different locations and tile the random images in a 2×2 pattern to generate a new, more complicated 64×64

pixel image containing four different images of a real robotic arm. We call this tiled image dataset MPI3D-Multi.

5.2 Training

For the three-dots and MPI3D-Multi datasets, we train using a Bernoulli loss on batches of 64 images over a total of 300,000 batches. We use the Adam optimizer with a learning rate of 0.0001 (matching [32]). We use linear annealing from 0 to the final hyperparameter value over the first 100,000 batches for both the beta hyperparameter and JL1-VAE's γ parameter in our implementations for JL1-VAE and β-VAE (unlike [32]). We note annealing to be beneficial to avoid model collapse when adding our L_1 regularization term. We train these models on a Nvidia Quadro V100 hosted locally and one hosted on Google Cloud. Training each JL1-VAE model on ten latent variables takes approximately 2.5 hours, while training each β-VAE model takes approximately 45 minutes. In total, training ten LIRJ-VAE models and ten β-VAE models for quantitative evaluation takes roughly 33 hours.

For the natural image dataset, we train for 100,000 batches of 128 images. We use the Adam optimizer with a learning rate of 0.001 and train on a Nvidia Quadro V100s hosted locally. Training takes 9 minutes for the β-VAE and 23 minutes for the JL1-VAE.

5.3 Evaluation Metrics

There are several metrics commonly used to measure "disentanglement" of a latent representation. In this work we address two common metrics, the Mutual Information Gap (MIG) [33] and Modularity [34], and show how we are able to provide extensions to these metrics that give a measure of how well a representation *locally* disentangles factors of variation.

The original MIG and modularity metrics measure global disentanglement—that is, they measure across the whole dataset how well each latent variable maps to a unique ground-truth factor of variation. Since the JL1-VAE does not add an explicitly global disentanglement incentive to β-VAEs, but instead is designed to encourage disentanglement locally using the Jacobian of the generative map, we do not expect it to necessarily improve the global disentanglement of factors of variation. For example, the JL1-VAE may align factors of variation with latent variables using one pairing in one local region of the latent space and a different pairing in a different region of the latent space. This could lead to good average local disentanglement, but would not lead to good global disentanglement.

We are therefore interested in defining *local* disentanglement metrics based on MIG and modularity. We call these metrics "local MIG" and "local modularity," and the general form of their calculation is to compute each metric several times on randomly chosen "local" samples from the global dataset and average the results. As with MIG and modularity, the calculation of these disentanglement metrics requires a generative model of the data from ground-truth factor values $z \in \mathbb{R}^K$ (where K is the number of ground-truth factors of variation).

The key technique for each of our local metrics is to repeatedly compute the disentanglement metric on random local samples of data. To generate a random local sample of data, we proceed as follows. First, we randomly choose a ground-truth "centroid" $\hat{z}$ across the entire space of ground-truth factor values. Then, we pick a set of N random ground-truth datapoints z^j (for $0 \leq j < N$) where each ground-truth factor value z_i^j (for $0 \leq j < N, 0 \leq i < K$) is within a distance ρ of the centroid value $\hat{z}_i$. The radius ρ is a hyperparameter determining how close ground-truth factors of variation need to be in order to be considered "local." This set of N datapoints comprises our local data sample. For our experiments we choose $N = 10,000$.

For discrete factors of variation that do not come with a natural distance metric such as "shape" or "color," we require equality in order for values to be considered "close." That is, for any such factor of variation, a "local" sampled dataset will be constant on that factor. That was not an issue for the three-dots dataset, as all ground-truth factors of variation were ordered. Additionally, we write the hyperparameter ρ as a fraction of the total range of available values for each latent variable. This allows the hyperparameter to take into account different scaling of different latent ground-truth values.

For each local sample of data, we apply the MIG and modularity metrics to that sample to determine the disentanglement of the latent space in that local region. We use the MIG and modularity

(a) Jacobian matrix columns for a β-VAE (b) Jacobian matrix columns for an JL1-VAE

Figure 2: Qualitative results for three-dots. Both models used $\beta = 4.0$. For JL1-VAE, $\gamma = 0.1$. We show the six largest (by L_2 norm) Jacobian matrix columns.

calculation implementations from the open-source (Apache License 2.0) `disentanglement_lib` library [32] which uses TensorFlow [35].

We repeat this algorithm 20 times and report the average as the local disentanglement score.

6 Results

We present qualitative results for both the three-dots and the natural image datasets, and quantitative results for the three-dots dataset (for which we have ground-truth factors of variation).

6.1 Qualitative Results

Qualitative results are generated by picking an arbitrary image, finding the (deterministic) latent representation of that image, and then computing the generative Jacobian matrix evaluated at that latent point. The generative Jacobian matrix columns for natural images are shown in Figure 1 and are compared to the PCA and ICA bases of the image crops. We note that the generative Jacobian matrix columns of the JL1-VAE show localized regions of activity, which is more similar to what is seen in the ICA basis of the natural image patches than in the PCA basis. This is in contrast to β-VAE matrix columns, which do not show the same locality in the receptive field, but seem to show similar levels of receptivity across the whole image. We show the largest 20 PCA components, an arbitrary sample of 20 ICA components (after training 100 ICA components using FastICA [36]), and the ten Jacobian matrix columns of the generative map with the largest L_2 norm for the β-VAE and JL1-VAE models in Figure 1. More PCA and ICA components are included in the Appendix as well as additional results for β-VAEs and JL1-VAEs. We discern more specific locality in the JL1-VAE and ICA receptive fields, when compared to the β-VAE and PCA receptive fields.

For the three-dots data and MPI3D-Multi, to generate qualitative results, we similarly choose an arbitrary image from the dataset, and then embed the image to the latent representation space and compute the generator Jacobian matrix at that latent value.

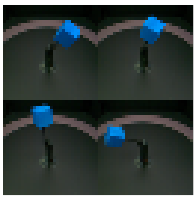
The six Jacobian columns with the largest L_2 norms for the three-dots dataset are shown in Figure 2 for both our JL1-VAE and β-VAE. All ten Jacobian columns are visualized in the Appendix. The model trained with the L_1 cost shows a stronger alignment between latent variable motion and individual dot motions. Qualitatively, we see that, when evaluating the Jacobian of the generator function for our JL1-VAE, individual dot motions are separated into different latent components. The β-VAE does not exhibit this behavior.

For the MPI3D-Multi dataset, containing tiled images of a real robot, we again see that the JL1-VAE does a better job separating the four robots contained in each image into separate latent variables. These results are presented in Figure 3.

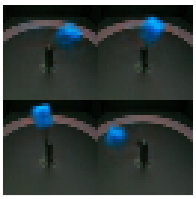
6.2 Quantitative Results

We generate local disentanglement scores for the models trained on the three-dots images.

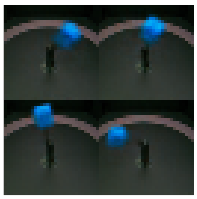
To observe the effect of the ρ parameter of our local disentanglement metrics, we first plot the varying local disentanglement scores as we change the ρ parameter for both our JL1-VAE with $\beta = 4.0$ and $\gamma = 0.1$ and for a standard β-VAE with $\beta = 4.0$ in Figure 4. The hyperparameter for β was chosen near the middle of the range used in [32]. We also tried other hyperparameter values and saw similar results. Too large a γ can lead to model collapse, so we chose a small enough γ to avoid that collapse but otherwise large enough to start to see reconstruction performance degradation so we knew that its regularization was affecting model training.



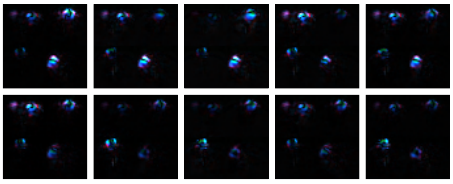
(a) MPI3D-Multi image



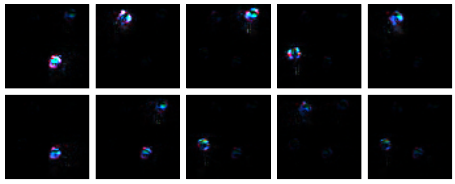
(b) β-VAE reconstruction



(c) JL1-VAE reconstruction



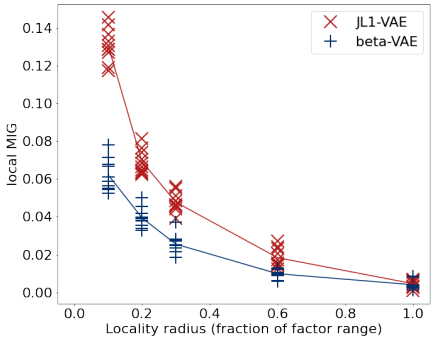
(d) Jacobian matrix columns for a β-VAE



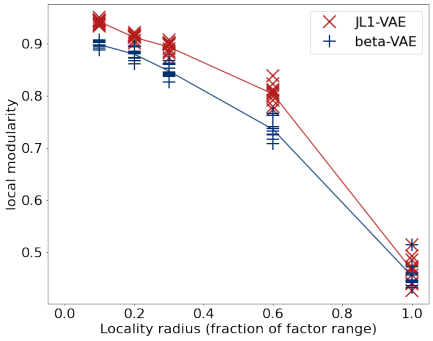
(e) Jacobian matrix columns for a JL1-VAE

Figure 3: Qualitative results for MPI3D-Multi. JL1-VAE shows stronger pressure to locally disentangle individual robot motions. Both models used $\beta = 0.01$. For JL1-VAE, $\gamma = 0.01$.

We note quantitatively that JL1-VAE attains higher local disentanglement scores compared to β-VAEs, which is especially true as we look at more local samples of data, corresponding to a smaller ρ parameter. For $\rho = 0.01$ we see significantly higher local disentanglement scores for JL1-VAE compared to β-VAE ($p < 0.001$ for T-test), but for $\rho = 1$, corresponding to the global range of factor values, we see indistinguishable disentanglement scores between the two ($p > 0.05$ for T-Test). Our JL1-VAE is able to *locally* disentangle factors of variation for the three-dots dataset, but does not globally disentangle factors of variation.



(a) Local MIG scores



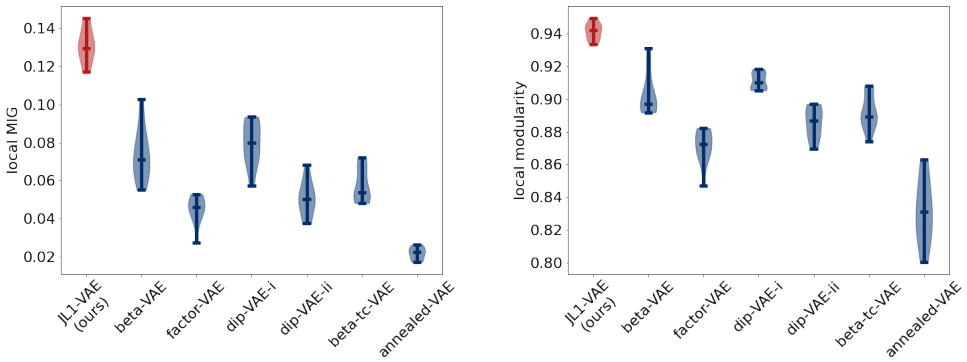
(b) Local modularity scores

Figure 4: Local disentanglement scores varying the locality parameter ρ. Ten JL1-VAE and β-VAE models were trained on the three-dots dataset with $\beta = 4$ and, for JL1-VAE, $\gamma = 0.1$.

Fixing the ρ parameter to 0.1, we also compute the local MIG and local modularity scores for six different comparative methods, namely β-VAE, FactorVAE, DIP-VAE-I, DIP-VAE-II, β-TCVAE, and AnnealedVAE, using the implementations of `disentanglement_lib` with our convolutional architecture. A description of each of these models can be found in [32]. We trained ten iterations of those models with different random seeds using hyperparameters chosen near the middle of the suggested ranges in that work. That included training ten new β-VAE models with new random seeds. All models were trained with ten latent dimensions.

Local MIG and local modularity scores are shown in Figure 5. We see a range of disentanglement scores due to the random seeds used to generate our models (ten models for each learning algorithm). Additionally, our local MIG and modularity metrics have some additional stochasticity due to the randomness in sampling 20 local samples of 10,000 points during calculation of those metrics.

Nevertheless, we observe significantly higher disentanglement scores ($p < 0.001$ for T-test) for our JL1-VAE compared to every baseline method.



(a) Local MIG scores

(b) Local modularity scores

Figure 5: Local disentanglement scores for JL1-VAE models and baseline implementations from [32]. The baseline implementations use default hyperparameters from that paper, choosing values near the middle when a range of hyperparameters are listed. Each model is run ten times with new random seeds. Local disentanglement is calculated using $\rho = 0.1$ with 20 different local samples.

7 Discussion

In this work, we presented JL1-VAE, a VAE augmented with an L_1 regularization to the Jacobian to improve local disentanglement. We extended the MIG and modularity disentanglement metrics to generate metrics that can measure local disentanglement. We evaluated our model on natural images, simulated images of dots, and tiled images of a real robot, and showed qualitatively and quantitatively that our method can improve local disentanglement in the generated representation.

Our added L_1 regularization to the Jacobian of the generator function is motivated by the implicit L_2 regularization already present in β-VAEs [2] and by ICA and sparse coding [6]. We show that this L_1 regularization term can encourage latent axes to locally align with ground-truth factors of variation of the dataset. While this approach shows promise for local alignment, it does not address global alignment issues. That is, in one part of the dataset, the learned representation may assign latent variable e_1 to follow a certain latent factor of variation, and in a different part of the dataset it might be a different latent variable e_2 that follows that latent factor of variation.

Regarding "no free lunch" theorems that show unsupervised disentanglement is impossible without inductive biases [32], we note that L_1 regularization of the generative Jacobian generates an inductive bias. The inductive bias encourages small axis-aligned perturbations of the latent space to result in sparse changes to the image space, whether that be expressed as localized receptive fields which act only on small regions of the image space, as seen in Figure 1, or motions of only a single dot or robot, as seen in Figures 2 and 3. Cases where this inductive bias might not add additional value would include, for example, single robot images from MPI3D, where the primary ground-truth factors of local variation affect the same object within the same image patch.

This work has only applied this learning method to image data, and we would like to apply this approach to multimodal data gathered from a real robotic system to understand broader applications.

Finally, this approach computes the full Jacobian of the generator during training, leading to training times that scale linearly with the number of latent dimensions. Future work should look to sampling-based methods to approximate the L_1 cost by only computing a subset of the Jacobian values in order to speed up training.

References

[1] Michal Rolinek, Dominik Zietlow, and Georg Martius. Variational autoencoders pursue pca directions (by accident). In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 12398–12407, 2019. ISBN 9781728132938.

[2] Abhishek Kumar and Ben Poole. On implicit regularization in β-VAEs. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5480–5490. PMLR, 13–18 Jul 2020. URL `http://proceedings.mlr.press/v119/kumar20d.html`.

[3] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014, 1312.6114.

[4] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 2017.

[5] Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

[6] Bruno A. Olshausen and David J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996. ISSN 00280836.

[7] Emile Mathieu, Tom Rainforth, N Siddharth, and Yee Whye Teh. Disentangling disentanglement in variational autoencoders. In *36th International Conference on Machine Learning, ICML 2019*, volume 2019-June, pages 7744–7754, 2019, 1812.02833. ISBN 9781510886988. URL `http://github.com/iffsid/disentangling-disentanglement`.

[8] Alireza Makhzani and Brendan Frey. k-Sparse autoencoders. In *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, 2014, 1312.5663.

[9] Andrew Ng. CS294A Lecture Notes Sparse Autoencoder. *Cs294a*, pages 1–19, 2011, arXiv:1506.03733v1. ISSN 19326203. URL `http://www.stanford.edu/class/cs294a/`.

[10] Linxing Preston Jiang and Luciano de la Iglesia. Improved training of sparse coding variational autoencoder via weight normalization. *CoRR*, abs/2101.09453, 2021, 2101.09453. URL `https://arxiv.org/abs/2101.09453`.

[11] Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through L0 regularization. In *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018, 1712.01312. ISBN 1712.01312v2.

[12] Misha Denil, Babak Shakibi, Laurent Dinh, Marc'aurelio Ranzato, and Nando De Freitas. Predicting parameters in deep learning. In *Advances in Neural Information Processing Systems*, 2013, 1306.0543.

[13] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *35th International Conference on Machine Learning, ICML 2018*, volume 6, pages 4153–4171, 2018, 1802.05983. ISBN 9781510867963.

[14] Francesco Locatello, Ben Poole, Gunnar Raetsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6348–6359. PMLR, 13–18 Jul 2020. URL `http://proceedings.mlr.press/v119/locatello20a.html`.

[15] Jan Stuehmer, Richard Turner, and Sebastian Nowozin. Independent subspace analysis for unsupervised learning of disentangled representations. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1200–1210. PMLR, 26–28 Aug 2020. URL `http://proceedings.mlr.press/v108/stuehmer20a.html`.

[16] Ilyes Khemakhem, Diederik Kingma, Ricardo Monti, and Aapo Hyvarinen. Variational autoencoders and nonlinear ica: A unifying framework. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2207–2217. PMLR, 26–28 Aug 2020. URL `http://proceedings.mlr.press/v108/khemakhem20a.html`.

[17] Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2424–2433. PMLR, 09–15 Jun 2019. URL `http://proceedings.mlr.press/v97/greff19a.html`.

[18] Nicholas Watters, Loic Matthey, Christopher P. Burgess, and Alexander Lerchner. Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes, 2019, 1901.07017.

[19] Samuel K. Ainsworth, Nicholas J. Foti, Adrian K. C. Lee, and Emily B. Fox. oi-VAE: Output interpretable VAEs for nonlinear group factor analysis. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 119–128. PMLR, 10–15 Jul 2018. URL `http://proceedings.mlr.press/v80/ainsworth18a.html`.

[20] Rayyan Ahmad Khan, Muhammad Umer Anwaar, and Martin Kleinsteuber. Epitomic variational graph autoencoder. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 7203–7210, 2021.

[21] Jakub M. Tomczak and Max Welling. VAE with a vampprior. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2018*, pages 1214–1223, may 2018, 1705.07120. URL `http://arxiv.org/abs/1705.07120`.

[22] Matthias Bauer and Andriy Mnih. Resampled priors for variational autoencoders. In *AISTATS 2019 - 22nd International Conference on Artificial Intelligence and Statistics*, oct 2020, 1810.11428. URL `http://arxiv.org/abs/1810.11428`.

[23] Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. Hyperspherical variational auto-encoders. In *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, volume 2, pages 856–865, apr 2018, 1804.00891. ISBN 9781510871601. URL `http://arxiv.org/abs/1804.00891`.

[24] Luis A. Perez Rey, Vlado Menkovski, and Jim Portegies. Diffusion variational autoencoders. In *IJCAI International Joint Conference on Artificial Intelligence*, volume 2021-Janua, pages 2704–2710, 2020, 1901.08991. ISBN 9780999241165.

[25] Salah Rifai, Yann N Dauphin, Pascal Vincent, Yoshua Bengio, and Xavier Muller. The manifold tangent classifier. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011*, 2011. ISBN 9781618395993.

[26] Dániel Varga, Adrián Csiszárik, and Zsolt Zombori. Gradient Regularization Improves Accuracy of Discriminative Models. *Schedae Informaticae*, 27:31–45, 2018, 1712.09936. ISSN 20838476.

[27] Judy Hoffman, Daniel A. Roberts, and Sho Yaida. Robust learning with jacobian regularization, 2019, 1908.02729.

[28] Nutan Chen, Alexej Klushyn, Francesco Ferroni, Justin Bayer, and Patrick van der Smagt. Learning flat latent manifolds with VAEs. In *37th International Conference on Machine Learning, ICML 2020*, volume PartF16814, pages 1565–1574, 2020, 2002.04881. ISBN 9781713821120.

[29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy,

Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[30] Shengjia Zhao, Hongyu Ren, Arianna Yuan, Jiaming Song, Noah D. Goodman, and Stefano Ermon. Bias and generalization in deep generative models: An empirical study. *CoRR*, abs/1811.03259, 2018, 1811.03259. URL http://arxiv.org/abs/1811.03259.

[31] Muhammad Waleed Gondal, Manuel Wuthrich, Djordje Miladinovic, Francesco Locatello, Martin Breidt, Valentin Volchkov, Joel Akpo, Olivier Bachem, Bernhard Schölkopf, and Stefan Bauer. On the transfer of inductive bias from simulation to the real world: a new disentanglement dataset. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/d97d404b6119214e4a7018391195240a-Paper.pdf.

[32] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, pages 4114–4124, 2019.

[33] Ricky TQ Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in vaes. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 2615–2625, 2018.

[34] Karl Ridgeway and Michael C. Mozer. Learning deep disentangled embeddings with the f-statistic loss. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 185–194, Red Hook, NY, USA, 2018. Curran Associates Inc.

[35] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 7
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] The authors were not able to identify potential negative societal impacts of this work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Figure 4 and Figure 5 show results for training each model ten times with different random seeds
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]