# Enforcing Sparsity on Latent Space for Robust and Explainable Representations

Hanao Li
Stevens Institute of Technology
Hoboken, NJ
hli136@stevens.edu

Tian Han
Stevens Institute of Technology
Hoboken, NJ
than6@stevens.edu

## Abstract

*Recently, dense latent variable models have shown promising results, but their distributed and potentially redundant codes make them less interpretable and less robust to noise. On the other hand, sparse representations are more parsimonious, providing better explainability and noise robustness, but it is difficult to enforce sparsity due to the complexity and computational cost involved. In this paper, we propose a novel unsupervised learning approach to enforce sparsity on the latent space for the generator model, utilizing a gradually sparsified spike and slab distribution as our prior. Our model is composed of a top-down generator network that maps the latent variable to the observations. We use maximum likelihood sampling to infer latent variables in the generator's posterior direction, and spike and slab regularization in the inference stage can induce sparsity by pushing non-informative latent dimensions toward zero. Our experiments show that the learned sparse latent representations preserve the majority of the information, and our model can learn disentangled semantics, increase the explainability of the latent codes, and enhance the robustness of the classification and denoising tasks.*

## 1. Introduction

In recent years, several approaches have been developed to study the mapping between high-dimensional observations and low-dimensional latent representations. Two of the most influential works are Variational Auto-Encoders (VAE) [13, 27, 28] and Generative Adversarial Networks (GAN) [7, 11, 25]. While these methods have shown impressive results, they tend to emphasize learning dense and distributed latent representations rather than disentangled or sparse ones. This makes it difficult to understand the individual latent codes.

Furthermore, dense models are generally not robust to noise as their redundant representations are sensitive to perturbations [1]. On the contrary, sparse representations are more robust and offer other advantages such as increased explainability in the latent space by encoding semantic information into a smaller subset of latent dimensions.

Many existing works have demonstrated great performance in learning disentangled or explainable latent variables [3, 24, 26]. For example, [18] uses a desired structure to impose a decomposition on the vanilla VAE model to learn disentangled latent representations, while [16] improves latent space disentanglement by adding gradient-based attention to the VAE model. Beta-VAE incorporates an adjustable parameter $\beta$ to obtain a more interpretable, factorized latent representation by sacrificing reconstruction ability [10]. However, these approaches mainly focus on learning disentangled latent representations with isotropic Gaussian as the prior distribution. Although these works have demonstrated notable disentanglement results in their experiments, their individual latent dimensions are not entirely disentangled. A change in the latent code will typically influence and alter other features simultaneously. Additionally, their learned latent representations are not parsimonious and are not as robust as sparse representations. Hence, we aim to develop a model that can learn both sparse and disentangled latent semantics.

Currently, few works learn a sparse representation in the latent space. One notable example is Variational Sparse Coding (VSC) [29]. It employs variational inference to learn the non-linear mapping between observation and latent space. VSC approximates the Dirac Delta function using a scaled sigmoid step function. [5] proposes using a soft thresholding operator to force a sparse latent representation. They use a straight-through estimator to estimate the gradient of latent variables. However, such estimation can lead to large bias and unreliable latent codes. Due to the nature of variational models, these models require approximations for the true posterior of the latent variable instead of exact inference. A common limitation of these approaches is that an extra inference network must be carefully designed to learn the mapping from observation to latent space. Since the inference step is an approximation of the true posterior, the sparsified latent codes are not effective and accurate.

To address these limitations, we propose a new train-

ing method for learning sparse latent representations using gradually sparsified spike and slab distribution as our prior belief. We train the model using maximum likelihood estimation and infer the latent variables using Monte Carlo Markov Chain sampling (MCMC) [20] to solve the intractable expectation. We use the maximum likelihood sampling method to further improve semantics and patterns learned by latent representations. We show theoretically that the spike and slab prior regularization in the inference step forces the latent codes that do not capture important information toward zero, and it leads to the learning of sparse and disentangled semantics. With our proposed learning scheme, we can perform exact inference in the generator model without the need to design an extra encoder network for approximated inference. The learned latent representations can be more accurate and effective.

Our contributions are summarized below:

- We present a new learning method to learn sparse latent representations via gradually sparsified spike and slab prior to avoid dead latent codes.
- We propose to use maximum likelihood sampling and spike-and-slab regularization on Langevin Dynamics to learn informative sparse latent variables.
- We conduct extensive experiments to demonstrate our sparse model can capture essential information of the original observation and lead to interpretable as well as robust representation.

## 2. Related Works

### 2.1. Learning Linear Sparse Representations

There have been many applications and algorithms that involve the learning of sparse representations [30, 32, 33]. One of the most fundamental methods to learn a sparse representation is sparse coding [15, 23]. It is an approximation strategy that is developed to solve the optimization problem of finding optimal weighted linear combinations of the basis matrix and coefficient matrix from an over-complete dictionary.

The goal of sparse coding aims to learn a meaningful sparse representation without losing many details while keeping only a small set of latent codes to have strong activation. [34] proposes Convolutional Sparse Coding (CSC) to learn rich features and it is based on a convolutional decomposition of images under sparsity constraint. [2] produces a fast convolutional sparse coding algorithm with superlinear convergence. [4] proposes using an iterative shrinkage-thresholding algorithm to directly regularize the latent variables and learn sparse coding. These works have focused on learning linear mapping of sparse coding but do not have the flexibility to learn complex mapping. Especially in the era of neural networks, non-linear relationships are more expressive and preferred nowadays.

### 2.2. Learning Non-Linear Latent Representations

There has been increasing popularity recently in learning the mapping between observation and latent space [7, 13]. The Alternating Back-Propagation (ABP) algorithm uses a unified probabilistic framework to train the generator network [8]. The generator employs a deep generative network to establish a complex non-linear mapping from latent to observation. Throughout the training, ABP uses a persistent chain for MCMC sampling [20] which involves warm start sampling to obtain posterior latent samples. [22] suggests the use of a short-run non-persistent chain for posterior sampling. However, these models have dense latent representations and do not aim to learn a sparse representation under their training procedure. The entangled latent variables lack interpretability and are not robust to noisy data. To address this issue, [9] extends the vanilla ABP for disentangled latent representations learning, but the learned latent variables remain non-sparse. [31] proposes a hierarchical AND-OR generator model to learn meaningful interpretations of the latent variables.

## 3. Proposed Method

### 3.1. Spike and Slab Prior

We present a new learning method to learn sparse representations using spike and slab distribution as our prior belief for the latent variables on the generator model. The distribution consists of spike variable and continuous slab variable [6, 19]. The spike variable has a probability $\alpha_1$ that determines whether the latent variables take values from the standard Gaussian or the slab variable distribution where it can be either a Dirac delta function or another Gaussian distribution. Since the Dirac delta function is not differentiable, we instead adopt a Gaussian distribution centered at 0 with a small variance to approximate the behavior of the Dirac delta function. The prior distribution can still be regarded as spike and slab but now it can also be viewed as a Gaussian mixture model with weights $\alpha_1 + \alpha_2 = 1$ as shown in Equation 1.

$$z \sim p_{ss}(z) = \alpha_1 N(0, \sigma_1^2) + \alpha_2 N(0, \sigma_2^2) \tag{1}$$

where $\sigma_1^2$ is fixed to be 1 as the variance of standard Gaussian distribution and $\sigma_2^2$ is the variance of slab Gaussian distribution. With this prior, the sparsity of our latent variable $z$ can be determined by changing the value of $\alpha_1$. Theoretically, we should have a small $\alpha_1$ and $\sigma_2^2$ to induce sparsity. With a small $\alpha_1$, $\alpha_2$ will be large and it's more likely to sample points from the slab variable distribution. When we have a small $\sigma_2^2$ simultaneously, most points sampled from this distribution $p_{ss}(z)$ will be small and close to 0. Thus

we can induce a sparse representation of the latent space in this setting.

**Sparsified Latent Codes:** During training, when we enforce a highly sparse latent power (e.g. $\alpha_1 = 0.01$) at the start of the iteration, some of the latent codes will not have the opportunity to learn anything before getting pushed towards zero. We demonstrate by slowly decreasing the value of $\alpha_1$ in the course of training, individual latent variables can gradually capture semantic knowledge. This can help alleviate the problem of dead latent codes.

### 3.2. Latent Inference and Model Learning

For each given observation $x$, we assume there is a corresponding latent variable $z$. The generator model $G$ that maps the latent variables into observations can be represented as:

$$x = G_\theta(z) + \epsilon;\ \epsilon \sim N(0, \sigma^2 I_D) \tag{2}$$

where $x \in \mathbb{R}^d$, $\sigma$ is the pre-specified standard deviation of the noise vector $\epsilon$ and the generator $G$ is parameterized by a top-down neural network with weights $\theta$. Equation 2 implies that the conditional distribution $p_\theta(x|z)$ is also a Gaussian distribution with $p_\theta(x|z) \sim N(G_\theta(z), \sigma^2 I_D)$. Given these two distributions $p_{ss}(z)$ and $p_\theta(x|z)$ , our complete-data log-likelihood between observation $x$ and latent variable $z$ can be formulated as follows:

$$\begin{aligned}\log p_\theta(x, z) &= \log[p_{ss}(z) p_\theta(x|z)] \\ &= -\frac{|x - G_\theta(z)|^2}{2\sigma^2} + \log p_{ss}(z) + \log \frac{1}{\sqrt{2\pi}\sigma}\end{aligned} \tag{3}$$

The log of joint distribution consists of the reconstruction error and log prior penalty term. This model can be trained using maximum likelihood estimation. The observed-data log-likelihood $L(\theta)$ given observations $\{x_1, x_2, ..., x_n\}$ can be written as:

$$L(\theta) = \sum_{i=1}^{n} \log p_\theta(x) = \sum_{i=1}^{n} \log \int_z p_\theta(x_i, z_i) dz \tag{4}$$

Differentiate with respect to model weights $\theta$, the gradient of log-likelihood can be derived as:

$$\frac{\partial}{\partial \theta} L(\theta) = \sum_{i=1}^{n} E_{p_\theta(z_i|x_i)}[\frac{\partial}{\partial \theta} \log p_\theta(x_i, z_i)] \tag{5}$$

However this posterior inside the expectation can be intractable to sample from, therefore, we consider using non-persistent short-run Langevin Dynamics with spike and slab prior to obtaining a sparse latent representation. Then the generator posterior distribution of $z$ in Equation 5 can be updated via:

$$z_{\tau+1} = z_\tau - \frac{s^2}{2} \frac{\partial}{\partial z} [\frac{|x - G_\theta(z)|^2}{2\sigma^2} - \log p_{ss}(z)] + s\epsilon_{LD,\tau} \tag{6}$$

where $s$ denotes the learning step size, $\epsilon_{LD}$ denotes the noise diffusion term sampled from a standard Gaussian distribution and $\tau$ is the time step of Langevin Dynamics.

**Zero Initialization:** Theoretically, for small step size $s$, the marginal distribution of $z$ can asymptotically coverage to the target posterior distribution $p_\theta(z|x)$ as $\tau \to \infty$.This enables us to initialize $z_0$ from any fixed distribution $p_0$. The starting distribution can be either Gaussian, spike-and-slab, or zero initialization. In our experiments, we opt for zero initialization as it aligns better with our objective of achieving sparsity and it leads to improved performance compared to initializing from other distributions.

In Equation 6, the derivative of the log of spike-and slab prior regularization term with respect to the latent variable $z$ can be represented as:

$$\frac{\partial}{\partial z} \log p_{ss}(z) = -\frac{z}{\sigma_1^2} + \frac{R_1 R_2 z}{e^{-R^2 \frac{z^2}{2}} + R_1} \tag{7}$$

where we denote $R_1 = \frac{(1-\alpha_1)\sigma_1}{\alpha_1 \sigma_2}$ and $R_2 = \frac{\sigma_2^2 - \sigma_1^2}{\sigma_1^2 \sigma_2^2}$. See Appendix for detailed derivation. This regularization term can push the latent codes towards zero by giving it a larger gradient if it does not capture meaningful information about the observation which allows us to achieve sparsity in the latent space.

**Maximum Likelihood Sampling:** In Equation 5, the expectation term can be estimated using multiple samples to ensure accurate approximation. However, many MCMC-based approaches [8, 22] only obtain one sample from the Markov chain because running $m$ separate chains can be computationally expensive. Even though it is sub-optimal, it can still be more accurate than the approximation from variational inference. Alternatively, we can obtain $m$ different samples along a single MCMC trajectory by skipping $K$ steps in between. As $K$ increases, the samples can become less dependent. Specifically, assuming the first sample is obtained at time-step $\tau + K$, for each observation $x_i$, we can select $m$ samples from Langevin Dynamics:

$$z_{i,1,...,m} = [z_{\tau+K}, z_{\tau+2K}, ..., z_{\tau+mK}] \tag{8}$$

After we have obtained a list of samples for the expectation term in Equation 5, we can average over all samples to calculate the expectation. However, in the case of sparse representations, it can lead to learning repeated features [5]. Instead, we can choose the latent representation from multiple samples that can lead to maximum likelihood. At the same time, the inferred $z_{i,max}$ can be used to approximate the expectation.

$$z_{i,max} = \underset{z_{i,m}}{\operatorname{argmax}} \log[p_\theta(x_i|z_{i,m})] \tag{9}$$

With this approach, reconstructed images with the best combination of current sparse latent codes can be used to

update the model. Therefore, developed features can be reused and learned latent codes can be more informative.

After latent codes have been inferred, they can be plugged into Equation 3 to form the complete-data log-likelihood, and the generator model can be learned using stochastic gradient descent:

$$\theta_{t+1} = \theta_t + \eta \frac{1}{n} \sum_{i=1}^{n} E_{p_\theta(z_i|x_i)}[\frac{\partial}{\partial \theta} \log p_\theta(x_i, z_i)] \quad (10)$$

where $\eta$ is the learning rate and selected $z_{i,max}$ from multiple samples will be used to learn the generator model.

### 3.3. Theoretical Understanding

From Equation 1, we observe that the spike and slab distribution is essentially a Gaussian mixture model with two Gaussian components when the spike variable uses Gaussian distribution to approximate the Dirac Delta function. We denote $p(C_i) = \alpha_i$ as the prior probability of component $i$. The posterior probability of the component $p(C_i|z)$ given latent variable is bounded between 0 and 1, and can be represented as follows:

$$\begin{aligned} p(C_i|z) &= \frac{\alpha_i N(0, \sigma_i^2)}{\alpha_i N(0, \sigma_i^2) + \alpha_j N(0, \sigma_j^2)} \\ &= \frac{\frac{\alpha_i}{\sigma_i}}{\frac{\alpha_i}{\sigma_i} + \frac{\alpha_j}{\sigma_j} e^{z^2(\frac{1}{2\sigma_i^2} - \frac{1}{2\sigma_j^2})}}; \quad i, j \in \{1, 2\}; i \neq j \end{aligned} \quad (11)$$

See Appendix for detailed derivation. Theoretically, one of the component's variance must be small to induce sparsity, so we let $\sigma_2^2 < \sigma_1^2 = 1$ and $\frac{1}{2\sigma_1^2} - \frac{1}{2\sigma_2^2}$ will always be negative. With these premises, we can derive and rewrite the gradient of the logarithm of the spike and slab prior from Equation 7 in terms of the posterior probability $p(C_i|z)$ as below:

$$\frac{\partial}{\partial z} \log p_{ss}(z) = -p(C_1|z)\frac{z}{\sigma_1^2} - p(C_2|z)\frac{z}{\sigma_2^2} \quad (12)$$

See Appendix for detailed derivation. Since we want to have more points sampled from the Gaussian distribution with small variance, we set $\alpha_1 < \alpha_2$ and then $p(C_2) > p(C_1)$.

When latent variable $z$ tends to be large, $p(C_1|z)$ will approach 1 while $p(C_2|z) * z$ will approach 0, so the gradient term will become $-\frac{z}{\sigma_1^2}$ since it is more likely to be sampled from standard Gaussian component. But when $z$ is relatively small, $p(C_1|z)$ will approach to 0 while $p(C_2|z)$ will approach to 1, and the gradient will become $-\frac{z}{\sigma_2^2}$. Since $\sigma_2^2$ is a very small number, we will have a larger gradient for a small value of $z$ which means a stronger power to push it towards 0. This indicates that the value of $z$ for this dimension must contain important information to overcome this power. Otherwise, if $z$ holds inconsequential information, the prior term will penalize it and thus enforce sparsity to a certain degree.

**Algorithm 1** Sparse Latent Representations Learning

**Input:** observations $x_i$, number of epochs $T$, learning rate $\eta$, current sparsity level $\alpha_C$, sparsity decay constant $\gamma$, sparsity threshold $\alpha_T$, number of samples $m$.
Let $t = 0$;
**repeat**
  **Sparsify Latent**: If $\alpha_C > \alpha_T$, then $\alpha_C = \alpha_C - \gamma$.
  **Latent Initialization**: For each $x_i$, initialize a new $z_i$ from fixed prior distribution $z_i \sim p_0(z)$.
  **Latent Update**: Infer $m$ samples $z_{i,1}, ..., z_{i,m}$ from posterior distribution with Equation 6 and 7.
  **Latent Selection**: Choose $z_{i,max}$ from $m$ samples that can maximize the likelihood using Equation 9.
  **Model Update**: Fix the inferred $z_{i,max}$ and observation $x_i$, update the model parameters with learning rate $\eta$ according to Equation 10.
  Let $t = t + 1$;
**until** $t = T$

## 4. Experiments

**Datasets:** We test our model on MNIST [14], Fashion-MNIST [17], CelebA [17] and SVHN [21] datasets. MNIST and Fashion-MNIST are grey-scaled images while CelebA and SVHN are color images. CelebA images are cropped and resized to dimensions of 64 by 64.

**Experimental Settings:** All training and testing images are scaled to [0, 1]. Since sparse coding usually requires an overcomplete basis set [23], we use 200 latent dimensions for MNIST and Fashion-MNIST, 400 for SVHN, and 800 for CelebA. We fix batch size = 100, $\sigma$ = 0.3, langevin step size = 0.1, langevin steps = 30, slab variable variance = 0.1, and $\alpha_1 = 0.01$. Initial sparsity is set to 1 and the decay constant $\gamma$ is 0.033. We choose $m$ = 5 and $K$ = 5 for maximum likelihood sampling to balance between the efficiency and performance. The noise diffusion term in Langevin Dynamics has a weight of 0.0001. We utilize cold start to train the model with zero initialization at the start of every epoch. Adam optimizer [12] is used with a learning rate of 0.0001. For a fair comparison with the VSC model [29], we adopt their model structure for our experiments.

### 4.1. Sparsity Control

We begin by verifying that our training can enforce sparsity on the latent codes. In this experiment, we demonstrate that our model can effectively learn a sparse representation

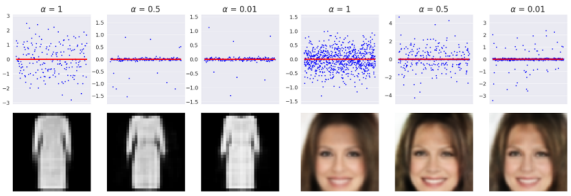

Figure 1. Latent Representations with Different Levels of sparsity level $\alpha_1$. **Top:** Learned Latent Representations. **Bottom:** Corresponding Reconstructed Images.

of the latent codes by simply changing the prior probability $\alpha_1$. We obtain the learned latent representations of the current observation using Langevin Dynamics, and we set the value of $\alpha_1$ to be [1, 0.5, 0.01] for comparison and illustration.

Our results, shown in Fig 1, reveal that when the prior probability for the standard Gaussian component ($\alpha_1$) is set to 1, the learned latent representation is dense, and nearly all the latent codes are activated. However, as we gradually decrease the value of $\alpha_1$ to 0.01, we observe fewer latent codes with stronger activation, and the rest of the latent codes are pushed towards 0. This demonstrates the effectiveness of our algorithm in learning sparse representations. We also note that the reconstructed images using sparse latent codes do not lose much information compared to those using dense latent codes on a given observation.

### 4.2. Reconstruction of Sparse Representations

One important factor in measuring whether learned latent codes can capture essential information from observations is their ability to reconstruct. We demonstrate that our model can reconstruct testing images well with only a small number of activated latent variables across various datasets. We compare our results with VSC using the same value of $\alpha$ ($\alpha = 0.01$) [29], VAE [13], Beta-VAE ($\beta = 4$) [10], short-run inference model [22], and thresholding model [5]. We assess reconstruction performance by comparing the quality of reconstruction and evaluating the peak signal-to-noise ratio (PSNR) in Table 1.

Our model outperforms other sparse and disentangled models while remaining competitive with the dense VAE model in terms of the PSNR metric. Although our model obtained a lower PSNR than the short-run model, we used very few activated latent codes ($\alpha$ = 1). As demonstrated in Fig 2, the reconstructed images using learned sparse latent representations visually preserve the majority of information from the observations.

Figure 2. Reconstructed Images on various datasets. **Top:** Original Images. **Bottom:** Reconstructed Images.

| Model | MNIST | Fashion | CelebA | SVHN |
|---|---|---|---|---|
| VAE [13] | 19.63 | 19.59 | 23.87 | 23.77 |
| Beta-VAE [10] | 16.84 | 17.01 | 20.53 | 19.82 |
| TC-VAE [3] | 16.19 | 16.82 | 20.37 | 18.44 |
| Short-run [22] | 21.06 | 21.81 | 26.78 | 29.21 |
| VSC [29] | 16.37 | 17.23 | 20.06 | 20.66 |
| Threshold Gaussian [5] | 12.91 | 12.13 | 16.57 | 17.08 |
| Threshold Laplacian [5] | 12.63 | 11.64 | 16.11 | 16.88 |
| Ours ($\alpha = 0.01$) | 19.77 | 19.95 | 24.71 | 26.57 |

Table 1. Peak Signal-to-Noise Ratio (dB). Higher PSNR metric means better reconstruction quality.

### 4.3. Latent Code Classifier

We investigate whether our learned sparse representations can retain important information and increase the robustness of the classifier. We hypothesize that our learned representations should generalize well with smaller training datasets. We test the model using a varying number of latent dimensions, ranging from 10 to 250 and randomly select 500 images from the MNIST training dataset. We encode the training observations to the latent codes and use them as input to train a one hidden linear layer classifier. We then encode the testing dataset to the latent space and test the accuracy of the encoded latent codes using the trained classifier.

As shown in Fig 3, our model can encode key class information into the latent variables without losing accuracy using a small amount of training data and can outperform other models at different latent dimensions. When the dimension of the latent space is increased, sparse models can fill in more details to further aid in distinguishing digits. In contrast, dense models saturate, and the accuracy plateaus.

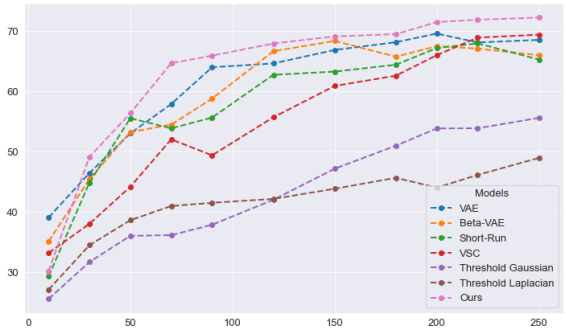


Figure 3. Classification Accuracy on MNIST using encoded latent variables. The accuracy results are averaged over 5 trials. Our model obtains higher accuracy with increased latent dimensions. Dense models will saturate and lose accuracy over time.

Therefore, we demonstrate that our learned sparse representations can lead to a more robust classifier, as the classifier learned with sparse codes will not overfit as the dimension increases.

To verify our classifier results, we also present t-SNE plots using the learned latent codes with 200 latent dimensions for the MNIST dataset. As shown in Fig 4, the sparse latent representations learned by our model can be well separated into different clusters compared to existing models. The ability to separate the clusters implies that our model has captured pivotal knowledge of each class, leading to a more robust classifier.

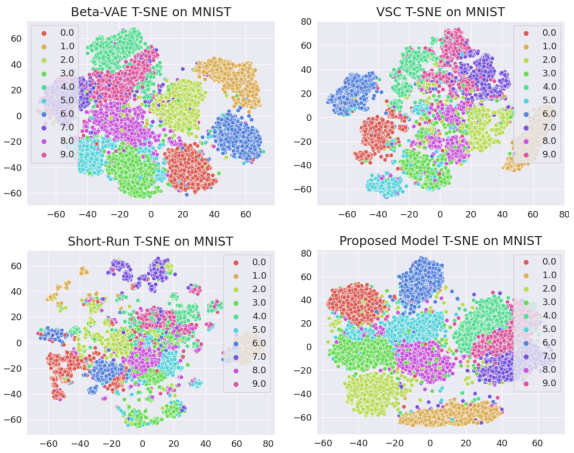


Figure 4. T-SNE plots of different models. Top left: Beta VAE; Top Right: VSC; Bottom Left: Short-run model; Bottom Right: Proposed Model.

| Model | $\sigma = 0.1$ | $\sigma = 0.3$ | $\sigma = 0.5$ |
|---|---|---|---|
| Short-run [22] | 0.337 | 0.182 | 0.119 |
| TC-VAE [3] | 0.278 | 0.144 | 0.099 |
| VSC [29] | 0.293 | 0.146 | 0.078 |
| Threshold Gaussian [5] | 0.134 | 0.074 | 0.041 |
| Threshold Laplacian [5] | 0.179 | 0.093 | 0.058 |
| Ours | **0.355** | **0.201** | **0.132** |

Table 2. Performance on Structural Similarity Index (SSIM). Higher SSIM implies better image quality. Our model can produce the best reconstructions from noisy images in terms of Luminance, Contrast, and Structure.

### 4.4. Robust Latent Representations

Sparse latent representations should also be robust to noisy images semantically and visually as they can encode important structural information from the observation to a small subset of activated latent codes. We apply zero mean Gaussian noise with different variances on testing images to evaluate the model's performance of denoising. In our case, we first try out the ability to reconstruct from noisy images. The latent variables are obtained from the inference step to recover clean images from these noisy ones.

In Fig 5, we can observe that for the short-run and VSC models, they will restore wrong digits when the noise variance is high. Their dense latent variables are sensitive to small changes in the latent space and they are not robust to the added noise. At the same time, our model can faithfully recover from the Gaussian noise and obtain the correct digit. This is because the sparse model can learn key structure information of each class while pushing non-informative latent towards zero so they are not easily affected by the noise.

In addition, we also calculate their Structural Similarity Index (SSIM) between noisy and denoised images to compare the perceptual changes and reconstructed image quality. We add various Gaussian noise where $\sigma = [0.1, 0.3, 0.5]$ into Fashion testing data. Our model can outperform both the sparse VSC model and the dense short-run model in Table 2. It implies the learned sparse latent representations using our algorithm are more robust to noises.

Moreover, we use training MNIST data to train a simple one-layer classifier to test the model's generalization on noisy images. By adding Gaussian noise with different variances to the testing data, we can obtain the latent representations from the inference step and feed the reconstructed images into the classifier to compute classification accuracy. The classification accuracy is averaged over 5 runs.

In Table 3, our model achieves the highest accuracy among other sparse and dense models. It shows our model is resistant to the perturbations from the Gaussian noise. The model has learned to encode important semantics into

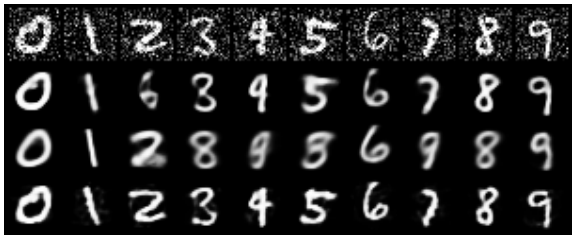

Figure 5. Denoising on MNIST Dataset. **First Row:** Noisy Images; **Second Row:** Short-run dense model; **Third Row:** VSC; **Last Row:** Proposed model. Noisy images are obtained from zero-mean Gaussian with standard deviation $\sigma$ = 0.3.

| Model | $\sigma = 0.3$ | $\sigma = 0.5$ | $\sigma = 0.7$ |
|---|---|---|---|
| Beta-VAE [10] | 72.31 | 62.13 | 59.49 |
| ABP [8] | 74.16 | 47.74 | 55.39 |
| Short-run [22] | 78.78 | 74.23 | 68.17 |
| VSC [29] | 72.41 | 57.94 | 53.54 |
| Threshold Gaussian [5] | 68.15 | 52.74 | 43.52 |
| Threshold Laplacian [5] | 67.29 | 56.51 | 51.07 |
| Ours | **81.16** | **75.30** | **71.79** |

Table 3. Model classification accuracy on noisy images. Higher accuracy means the model can better capture structure information of the observations and be robust to the added noise.

the sparse latent space while discarding latent codes with trivial contributions. So when there is noise added to the image, small perturbations will not have a large impact on the inference latent codes.

For color images in Fig 6, we use $\sigma$ = [0.1, 0.2, 0.3] as the standard deviation for zero mean Gaussian noise. The noise is added to each image channel. We observe that our model can also reconstruct the corrupted face and digit images without changing too much of the observations. This implies the learned sparse latent representations can be robust to the Gaussian perturbation. The added distortion from noise does not have much impact on well-learned latent codes. The sparse representation can capture meaningful and structural information about the observed images.

### 4.5. Latent Code Exploration

A well-trained model can capture a meaningful sparse latent representation. In this experiment, we demonstrate that our model can learn semantic meanings from images and encode them into individual latent codes. By altering a single activated dimension in the latent space, we observe disentangled changes in the reconstructed image, making the latent codes more explainable. In Fig 7, we demonstrate how changes in disentangled factors, such as facial

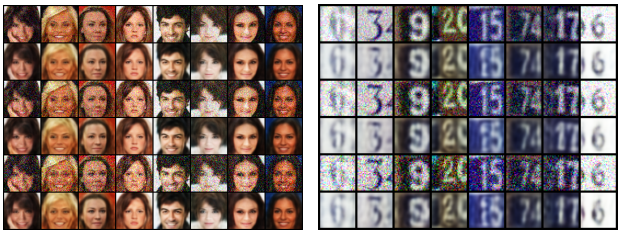

Figure 6. Denoising on CelebA and SVHN Images with various Gaussian noise. **Odd rows:** Given noisy images. **Even rows:** Reconstructed Images.

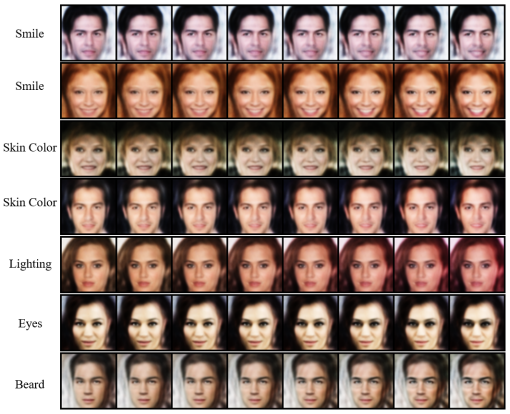


Figure 7. Latent Traversal using CelebA. First Column: Reconstructed image given observation. Second to Last Columns: Reconstructed Images after incrementally altering single activated latent code.

features (beard or without beard) and expressions (smile or non-smile), can be observed while keeping other features unaffected. The learned latent codes have shown their ability to discover disentangled factors from the training dataset in an unsupervised manner.

Furthermore, we trained our model on Fashion-MNIST with 30 latent dimensions to examine the features learned by the model. Since Fashion-MNIST contains 10 classes with shared structures between classes (e.g., Top and Pullover), explainable models should be able to extract this information and encode it into corresponding latent representations. For all testing data, we separated them by their class label, obtained their latent representations using Langevin Dynamics, and set an activation threshold of 0.3. We set activated latent codes to 1 and non-activated codes to 0, then averaged all latent codes by class and plotted a heatmap to examine the features learned from the dataset.

In Fig 8, we can clearly visualize the learned latent structure for each Fashion class. Each category forms an apparent pattern with only a few activated codes, while most latent dimensions remain inactive. These activated codes

have strong activation, and some other latent codes do not activate for certain categories. This makes the learned latent variables more interpretable when altering their latent code.

### 4.6. Ablation Study

In this section, we investigate the effect of the sparsified latent code and the maximum likelihood sampling method. We aim to evaluate the accuracy of reconstruction with learned latent codes and their performance in recovering structural knowledge from noisy images. SSIM is calculated from noisy testing datasets using zero mean Gaussian noise with $\sigma = 0.1$ for color images and $\sigma = 0.5$ for grey-scale images.

**Gradually Sparsified Latent vs. Constant Sparsity:** From Fig 9, we observe that the model can learn some structural information from the images with constant sparsity. However, it mixes some of the structural information into the same latent code for different categories. Although this method can lead to slightly better performance for certain datasets, its learned latent codes are not as disentangled as the model with sparsity decay while some of them are always kept activated. A possible explanation for this phenomenon is that some of the latent codes are pushed toward zero at the start of the training when we have a small sparsity level. Starting with a dense model can give us the benefit of letting each latent code has the ability to learn from images, and then non-informative codes will be forced toward zero as we gradually increase the sparsity value.

| Model | MNIST | | Fashion | | SVHN | | CelebA | |
|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM |
| Constant Sparsity | 19.54 | 0.33 | 19.34 | 0.37 | 25.38 | 0.41 | 22.51 | 0.67 |
| Average Sampling | 19.89 | 0.36 | 20.13 | 0.39 | 25.47 | 0.44 | 25.96 | 0.65 |
| Proposed Model | 19.77 | 0.38 | 19.95 | 0.36 | 24.71 | 0.44 | 26.57 | 0.69 |

Table 4. PSNR and SSIM metric on ablation models.

**Average Sampling vs. Maximum Sampling:** From Table 4, we observe that average sampling performs well in terms of reconstruction and similar performance for SSIM metric. However, from the right side of Fig 9, we notice that it does not learn a clear pattern as shown in Fig 8, which shows that there are more activated dimensions than the maximum sampling method. When averaging with $M$ different samples for the expectation, each of them can lead to different combinations of bases, resulting in the development of different sparse features along the latent dimensions.

## 5. Limitations

While our method only needs the generator to infer the sparse latent variables, the use of MCMC can still be time-consuming. For each iteration, variational models will cost about 3 seconds for the MNIST dataset while our method will cost 6.6 seconds. But at the same time, without an encoder, we could reduce about 50% of the number of total parameters. The other limitation is that since we are using a Gaussian mixture model to approximate the spike-and-slab distribution, the inactivated codes are close to zero but not exactly zero.

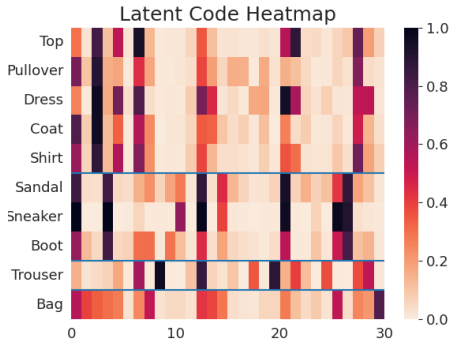


Figure 8. Heatmap of activated latent codes. Different categories are separated by a horizontal line. The first five rows contain upper wear, the middle three rows consist of footwear and the last two rows are bottom and hand-wear. Objects with the same category should have similar activated latent maps with small variations.

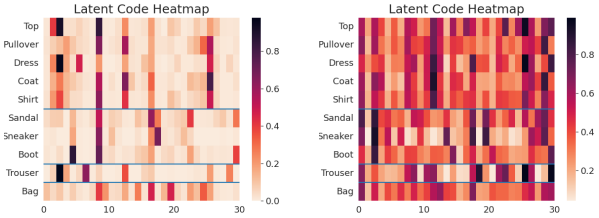


Figure 9. Latent heatmap for ablation models. **Left:** Model without sparsity decay. **Right:** Model with Average Sampling.

## 6. Conclusions

In this work, we present a new learning method to learn a sparse latent representation with a gradually sparsified spike and slab as our prior distribution. The model uses only one top-down generator to map from the latent space to observed images. The latent variable is inferred from zero initialization with Langevin Dynamics and selected using maximum likelihood sampling. Our model shows competitive reconstruction capability with only a few activated latent codes while preserving important information about the given observation. We also performed extensive experiments to demonstrate the sparse latent representation has improved explainability and boosted robustness in the task of denoising and latent classification.

## References


[1] Subutai Ahmad and Luiz Scheinkman. How can we be so dense? the benefits of using highly sparse representations. *arXiv preprint arXiv:1903.11257*, 2019. 1

[2] Hilton Bristow, Anders P. Eriksson, and Simon Lucey. Fast convolutional sparse coding. In *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pages 391–398. IEEE Computer Society, 2013. 2

[3] Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, 31, 2018. 1, 5, 6

[4] Katrina Evtimova and Yann LeCun. Sparse coding with multi-layer decoders using variance regularization. *CoRR*, abs/2112.09214, 2021. 2

[5] Kion Fallah and Christopher J Rozell. Variational sparse coding with learned thresholding. *arXiv preprint arXiv:2205.03665*, 2022. 1, 3, 5, 6, 7

[6] Ian J. Goodfellow, Aaron C. Courville, and Yoshua Bengio. Large-scale feature learning with spike-and-slab sparse coding. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012. 2

[7] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial networks. *CoRR*, abs/1406.2661, 2014. 1, 2

[8] Tian Han, Yang Lu, Song-Chun Zhu, and Ying Nian Wu. Alternating back-propagation for generator network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017. 2, 3, 7

[9] Tian Han, Xianglei Xing, and Ying Nian Wu. Learning multi-view generator network for shared representation. In *24th International Conference on Pattern Recognition, ICPR 2018, Beijing, China, August 20-24, 2018*, pages 2062–2068. IEEE Computer Society, 2018. 2

[10] Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 1, 5, 7

[11] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 4401–4410. Computer Vision Foundation / IEEE, 2019. 1

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 4

[13] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. 1, 2, 5

[14] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database, 2010. 4

[15] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann, editors, *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pages 801–808. MIT Press, 2006. 2

[16] WenQian Liu, Runze Li, Meng Zheng, Srikrishna Karanam, Ziyan Wu, Bir Bhanu, Richard J. Radke, and Octavia I. Camps. Towards visually explaining variational autoencoders. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 8639–8648. Computer Vision Foundation / IEEE, 2020. 1

[17] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015. 4

[18] Emile Mathieu, Tom Rainforth, N. Siddharth, and Yee Whye Teh. Disentangling disentanglement in variational autoencoders. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 4402–4412. PMLR, 2019. 1

[19] Toby J Mitchell and John J Beauchamp. Bayesian variable selection in linear regression. *Journal of the american statistical association*, 83(404):1023–1032, 1988. 2

[20] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011. 2

[21] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 4

[22] Erik Nijkamp, Bo Pang, Tian Han, Linqi Zhou, Song-Chun Zhu, and Ying Nian Wu. Learning multi-layer latent variable model via variational optimization of short run MCMC for approximate inference. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part VI*, volume 12351 of *Lecture Notes in Computer Science*, pages 361–378. Springer, 2020. 2, 3, 5, 6, 7

[23] Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997. 2, 4

[24] Antoine Plumerault, Hervé Le Borgne, and Céline Hudelot. Controlling generative models with continuous factors of variations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 1

[25] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional gen-

erative adversarial networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. 1

[26] Scott E. Reed, Kihyuk Sohn, Yuting Zhang, and Honglak Lee. Learning to disentangle factors of variation with manifold interaction. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1431–1439. JMLR.org, 2014. 1

[27] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1530–1538. JMLR.org, 2015. 1

[28] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1278–1286. JMLR.org, 2014. 1

[29] Francesco Tonolini, Bjørn Sand Jensen, and Roderick Murray-Smith. Variational sparse coding. In Amir Globerson and Ricardo Silva, editors, *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, July 22-25, 2019*, volume 115 of *Proceedings of Machine Learning Research*, pages 690–700. AUAI Press, 2019. 1, 4, 5, 6, 7

[30] John Wright, Allen Y. Yang, Arvind Ganesh, S. Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(2):210–227, 2009. 2

[31] Xianglei Xing, Tianfu Wu, Song-Chun Zhu, and Ying Nian Wu. Inducing hierarchical compositional model by sparsifying generator network. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 14284–14293. Computer Vision Foundation / IEEE, 2020. 2

[32] Jianchao Yang, John Wright, Thomas S. Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE Trans. Image Process.*, 19(11):2861–2873, 2010. 2

[33] Meng Yang, Lei Zhang, Jian Yang, and David Zhang. Robust sparse coding for face recognition. In *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*, pages 625–632. IEEE Computer Society, 2011. 2

[34] Matthew D. Zeiler, Dilip Krishnan, Graham W. Taylor, and Robert Fergus. Deconvolutional networks. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pages 2528–2535. IEEE Computer Society, 2010. 2