Variational Sparse Coding

Francesco Tonolini
School of Computing Science
University of Glasgow
Glasgow, UK

Bjørn Sand Jensen
School of Computing Science
University of Glasgow
Glasgow, UK

Roderick Murray-Smith
School of Computing Science
University of Glasgow
Glasgow, UK

Abstract

Unsupervised discovery of interpretable features and controllable generation with high-dimensional data are currently major challenges in machine learning, with applications in data visualisation, clustering and artificial data synthesis. We propose a model based on variational auto-encoders (VAEs) in which interpretation is induced through latent space sparsity with a mixture of Spike and Slab distributions as prior. We derive an evidence lower bound for this model and propose a specific training method for recovering disentangled features as sparse elements in latent vectors. In our experiments, we demonstrate superior disentanglement performance to standard VAE approaches when an estimate of the number of true sources of variation is not available and objects display different combinations of attributes. Furthermore, the new model provides unique capabilities, such as recovering feature exploitation, synthesising samples that share attributes with a given input object and controlling both discrete and continuous features upon generation.

1 INTRODUCTION

Variational auto-encoders (VAEs) offer an efficient way of performing approximate posterior inference with otherwise intractable generative models and yield probabilistic encoding functions that can map complicated high-dimensional data to lower dimensional representations (Kingma & Welling, 2013; Rezende et al., 2014; Sønderby et al., 2016). Making such representations meaningful, however, is a particularly difficult task and currently a major challenge in representation learning (Burgess et al., 2018; Tomczak & Welling, 2018; Kim & Mnih, 2018). Large latent spaces often give rise to many latent dimensions that do not carry any information, and obtaining codes that properly capture the complexity of the observed data is generally problematic (Tomczak & Welling, 2018; Higgins et al., 2017b; Burgess et al., 2018).

In the case of linear mappings, sparse coding offers an elegant solution to the aforementioned problem; the representation space is induced to be sparse. In such a way, the encoding function can exploit a high-dimensional space to model a large number of possible features, while being encouraged to use a small subset of non-zero elements to describe each individual observation (Olshausen & Field, 1996a;b). Due to their efficiency of representation, sparse codes have been used in many learning and recognition systems, as they provide easier interpretation when processing natural data (Lee et al., 2007; Bengio et al., 2013). Biological system have also notably been proven to exploit signal sparsity for visual perception (Olshausen & Field, 1996a;b).

In this work, we aim to extend the aforementioned capability of linear sparse coding to non-linear probabilistic generative models thus allowing efficient, informative and interpretable representations in the general case. To this end we formulate a new variation of VAEs in which we employ a sparsity inducing prior and a discrete mixture recognition model based on the Spike and Slab distribution. We construct a flexible sparse prior as a combination of Spike and Slab recognition models through auxiliary pseudo-inputs. We derive a non-trivial analytical evidence lower bound (ELBO) for the model and design a pre-training procedure that avoids mode collapse.[1]

In our experiments, we study feature disentanglement in the general situation where no estimate of the number of

[1] An implementation of the VSC model is available from `https://github.com/ftonolini45/Variational_Sparse_Coding`.

ground truth features is available and different features can be present or absent in each observed data example. We show how our model considerably outperforms traditional disentanglement approaches (Higgins et al., 2017a; Gao et al., 2019) in such conditions. We then consider two benchmark data sets, Fashion-MNIST and UCI HAR, and demonstrate how the variational sparse coding (VSC) model recovers sparse representations that properly capture the mixed discrete and continuous nature of their variability. We further show how such representations can be used to investigate feature relationships among different objects and classes and perform conditional generation completely unsupervisedly.

2 BACKGROUND

2.1 SPARSE CODING

Sparse coding aims to approximately represent observed signals with a weighted linear combination of few unknown basis vectors (Lee et al., 2007; Bengio et al., 2013). The task of determining the optimal basis is generally formulated as an optimisation problem, where a reconstruction cost and a sparsity cost of the embedded representations are jointly minimised with respect to the coefficients of a linear transformation. Sparse coding makes the important realisation that, though large ensembles of natural signals need many variables to be described, individual samples can be well represented by a small subset of such variables. This realisation is supported by substantial empirical evidence and pioneering work by Olshausen & Field (1996a) also showed that the visual cortex in mammals processes information as sparse signals, demonstrating that biological learning exploits sparsity in natural images similarly to sparse coding based models. Olshausen & Field (2004) provide a comprehensive review of linear sparse coding.

Sparse coding can be probabilistically interpreted as a generative model, where the observed signals are generated from unobserved latent variables through a linear process (Lee et al., 2007; Bengio et al., 2013). The model can then be described with a latent prior distribution, which assigns high density to sparse latent variables, and a likelihood distribution, which quantifies the reconstruction density given a latent embedding. In fact, performing maximum a posteriori (MAP) estimation with such models recovers the common formulation of sparse coding described above. Previous work has also demonstrated variational inference with sparse coding probabilistic models, exploiting algorithms based on EM inference (Titsias & Lázaro-Gredilla, 2011; Goodfellow et al., 2012). However, EM inference becomes intractable for more complicated non-linear posteriors and a large number of input vectors (Kingma & Welling, 2013), making such an approach unsuitable to scale to our target models.

Conversely, some work has been done in generalising sparse coding to non-linear transformations, by defining sparsity on Riemannian manifolds (Ho et al., 2013; Cherian & Sra, 2017). These generalisations, however, are not probabilistic, as they define a non-linear equivalent of MAP inference and are limited to simple manifolds due to the need to compute the manifold's logarithmic map.

2.2 VARIATIONAL AUTO-ENCODERS

Variational auto-encoders (VAEs) are models for unsupervised efficient coding that aim to maximise the marginal likelihood $p(\mathbf{x}) = \prod p(x_i)$ with respect to some decoding parameters θ of the likelihood function $p_\theta(x|z)$ and encoding parameters ϕ of a recognition model $q_\phi(z|x)$ (Kingma & Welling (2013); Rezende et al. (2014); Pu et al. (2016)).

The VAE model is defined as follows; an observed vector $x_i \in \mathbb{R}^{M\times 1}$ is assumed to be drawn from a likelihood function $p_\theta(x|z)$. Common choices are a Gaussian or a Bernoulli distribution. The parameters of $p_\theta(x|z)$ are the output of a neural network having as input a latent variable $z_i \in \mathbb{R}^{J\times 1}$. The latent variable is assumed to be drawn from a prior $p(z)$ which can take different parametric forms. In the most common VAE implementations, the prior takes the form of a multivariate Gaussian with identity covariance $\mathcal{N}(z; 0, I)$ (Kingma & Welling, 2013; Rezende et al., 2014; Higgins et al., 2017b; Burgess et al., 2018; Yeung et al., 2017). The aim is then to maximise a joint posterior distribution of the form $p(\mathbf{x}) = \prod_i \int p_\theta(x_i|z)p(z)dz$, which for an arbitrarily complicated conditional $p(x|z)$ is intractable. To address this intractability, VAEs introduce a recognition model $q_\phi(z|x)$ and define an evidence lower bound (ELBO) to be estimated in place of the true posterior, which can be formulated as

$$\log p_\theta(x_i) = \log \int p_\theta(x_i|z)p(z)\frac{q_\phi(z|x_i)}{q_\phi(z|x_i)}dz \geq \\ -D_{KL}(q_\phi(z|x_i)||p(z)) + \mathbb{E}_{q_\phi(z|x_i)}\left[\log p_\theta(x_i|z)\right]. \tag{1}$$

The ELBO is composed of two terms; a prior term, which encourages minimisation of the KL divergence between the encoding distributions and the prior, and a reconstruction term, which maximises reconstruction likelihood. The ELBO is then maximised with respect to the model's parameters θ and ϕ.

2.2.1 Interpretation in VAEs

Obtaining informative and interpretable representations of unlabelled data is currently a major objective of unsupervised learning. VAEs have been recently considered as ideal models to obtain such representations, as they rely on a low dimensional latent space that necessarily embeds information about the observation they model. Inducing interpretation in a VAE latent space is generally formulated as a feature disentanglement problem; assuming observed data is generated from hidden interpretable factors of variation, the task is to obtain a latent space in which such factors are aligned with the axis. The commonly adopted method to enhance the disentanglement of features in a VAE is to assign a larger weight to the KL divergence component in equation 1. Models following this approach are known as β-VAEs, and they allow controllable improvement of latent disentanglement, at the expense of reconstruction likelihood (Higgins et al., 2017a; Burgess et al., 2018). Several alternative methods to improve disentanglement have been proposed. Recent works extended the β-VAE to explicitly control total correlation between latent dimensions, expressing it as a penalty term that quantifies disentanglement (Chen et al., 2018; Gao et al., 2019). Kim & Mnih (2018) proposed an algorithm to directly induce factorisation, demonstrating a more favourable trade-off between disentanglement and reconstruction accuracy. These approaches have proven promising, however they rely on the assumption that target features are always present in every observation with continuously varying values. Differently from these, the model presented here relies on the sparse coding realisation of natural signals, assuming that individual observations are described by only a small subset of a large ensemble of possible features. Furthermore, we assume no knowledge of the number of source factors.

2.2.2 Discrete Latent Variables and Sparsity in VAEs

Discrete latent distributions are a closely related theme to sparsity, as exactly sparse PDFs involve sampling from some discrete variables. Nalisnick & Smyth (2017) and Singh et al. (2017) model VAEs with a Stick-Breaking Process and an Indian Buffet Process priors respectively in order to allow for stochastic dimensionality in the latent space. In such a way, the prior can set unused dimensions to zero. However, the resulting representations are not truly sparse; the same elements are set to zero for every encoded observation. The scope of these works is dimensionality selection rather than sparsification.

Other models which present discrete variables in their latent space have been proposed in order to capture discrete features in natural observations. Rolfe (2017) models a discrete latent space composed of continuous variables conditioned on discrete ones in order to capture both discrete and continuous sources of variation in observations. Similarly motivated, van den Oord et al. (2017) perform variational inference with a learned discrete prior and recognition model. The resulting latent spaces can present sparsity, depending on the choice of prior. However, they do not induce directly sparse statistics in the latent space.

Other works model sparsity more directly. Yeung et al. (2017) propose to learn a deterministic selection variable that dictates which latent dimensions the recognition model should exploit in the latent space. In such a way, different embeddings can exploit different combinations of variables, which achieves the goal of counteracting over-pruning. This approach does result into sparse latent variables. However, only the continuous components are treated variationally, while the activation of elements is deterministic. More recently, Mathieu et al. (2019) modelled sparsity in the latent space with a mixture of Gaussians models using a narrow Gaussian component to encourage elements to be close to zero. In their work, a continuous relaxation of sparsity is modelled in the latent space, as elements are not encouraged to be zero exactly, but only close to zero by the narrow Gaussian component of the prior.

Differently from these prior works, we directly model the mixed continuous-discrete nature of sparsity in the latent space through an exactly sparse prior and find a suitable evidence lower bound and training procedure to perform approximate variational inference.

3 VARIATIONAL SPARSE CODING

We propose to use the framework of VAEs to perform approximate variational inference with neural network sparse coding architectures. With this approach, we aim to discover and discern the non-linear features that constitute variability in data and represent them as few non-zero elements in sparse vectors.

3.1 RECOGNITION MODEL

In order to encode observations as sparse vectors in the latent space, the recognition model is chosen to be a Spike and Slab distribution. The Spike and Slab distribution is defined over two variables; a binary spike variable s_j and a continuous slab variable z_j (Mitchell & Beauchamp, 1988). The spike variable is either one or zero with defined probabilities γ and $(1 - \gamma)$ respectively and the slab variable has a distribution which is either a Gaussian or a Delta function centered at zero, conditioned on whether the spike variable is one or zero respectively. The resulting recognition model is

$$q_\phi(z|x_i) = \prod_{j=1}^{J} [\gamma_{i,j}\mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \\ +(1-\gamma_{i,j})\delta(z_{i,j})], \tag{2}$$

where the distribution parameters $\mu_{z,i,j}$, $\sigma^2_{z,i,j}$ and $\gamma_{i,j}$ are the outputs of a neural network having parameters ϕ and input x_i, J is the number of latent dimensions and $\delta(\cdot)$ indicates the Dirac delta function centered at zero. A description of the recognition model neural network can be found in supplementary A.2.

3.2 PRIOR DISTRIBUTION

In order to induce sparsity in the latent space while allowing the model to flexibly adjust to represent different combinations of features, we build upon two recent advances in VAEs. Following the prior structure presented in Tomczak & Welling (2018), we build the prior with recognition models $q_\phi(z|x_u)$ from pseudo-inputs x_u, which are trained along with the networks' weights. However, differently from this previous work, which builds the prior with the sum of all pseudo inputs' encodings $p(z) = \frac{1}{U}\sum_u q_\phi(z|x_u)$, we implement a classifier $u^* = C_\omega(x_i)$ with parameters ω to select a specific pseudo-input x_{u^*} and consequentially a single component $q_\phi(z|x_{u^*})$ for an observation x_i, hence assuming that each observation x_i is generated from a single component in the latent space. This feature is similar to the latent variable selection presented in Yeung et al. (2017), with the difference that the selector in the VSC model assigns a different prior for each observations, rather then different latent dimensions. The sparse prior is then

$$p_s(z) = q_\phi(z|x_{u^*}), \\ u^* = C_\omega(x_i), \tag{3}$$

where $C_\omega(x)$ is a neural network classifier. The use of the classifier $C_\omega(x)$ rather than taking the whole ensemble as prior allows us to compute the KL divergence analytically, hence rendering optimisation efficient while maintaining the flexibility of a prior composed of multiple PDFs. A detailed description of the selection function $C_\omega(x)$ can be found in supplementary A.3.

3.3 VSC OBJECTIVE FUNCTION

As in the standard VAE setting, we aim to perform approximate variational inference by maximising an ELBO of the form detailed in equation 1, with the Spike and Slab probability density function of equation 3 as prior and the recognition model of equation 2. Additionally, we need to infer a defined prior sparsity α in the latent space. This is done by inducing the average Spike probability of each pseudo-input's recognition model $\overline{\gamma}_u$ to match the prior one α. a sparsity KL divergence penalty term $D_{KL}(\overline{\gamma}_u||\alpha)$ is then minimised jointly to the maximisation of the ELBO

$$\arg\max_{\theta,\phi,\omega,\mathbf{x_u}} \sum_i -D_{KL}(q_\phi(z|x_i)||q_\phi(z|x_{u^*})) \\ + \mathbb{E}_{q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - J \cdot D_{KL}(\overline{\gamma}_{u^*}||\alpha). \tag{4}$$

The KL divergence between the pseudo-input's prior and target sparsity $D_{KL}(\overline{\gamma}_{u^*}||\alpha)$ has a simple form and can readily be differentiated with respect to the weights and pseudo-inputs. This term induces the encodings to have the prior sparsity on average, acting similarly to the aggregate posterior regularisation term described in Mathieu et al. (2019). In the following subsections, we elaborate on the remaining two terms: the reconstruction and KL divergence terms of the ELBO.

3.3.1 Reconstruction Term

The reconstruction component of the ELBO is estimated stochastically as follows

$$\mathbb{E}_{q_\phi(z|x_i)}[\log p_\theta(x_i|z)] \simeq \frac{1}{L}\sum_{l=1}^{L} \log p_\theta(x_i|z_{i,l}), \tag{5}$$

where the samples $z_{i,l}$ are drawn from the recognition model $q_\phi(z|x_i)$ and L is the number of such draws. As in the standard VAE, to make the reconstruction term differentiable with respect to the encoding parameters ϕ, we employ a reparameterization trick to draw from $q_\phi(z|x_i)$. To parametrise samples from the discrete binary component of $q_\phi(z|x_i)$ we use a continuous relaxation of binary variables analogous to that presented in Maddison et al. (2017) and Rolfe (2017). We make use of two auxiliary noise variables ϵ and η, normally and uniformly distributed respectively. ϵ is used to draw from the Slab distributions, resulting in a reparametrisation analogous to the standard VAE (Kingma & Welling, 2013). η is used to parametrise draws of the Spike variables through a non-linear binary selection function $T(y_{i,l})$. The two variables are then multiplied together to obtain the parametrised draw from $q_\phi(z|x_i)$. A more detailed description of the reparametrisation of sparse samples is reported in supplementary B.

3.3.2 Spike and Slab KL Divergence

KL divergences with discrete mixture PDFs have been used in various discrete latent variables models (Rolfe, 2017; Nalisnick & Smyth, 2017). However, in these works, they are estimated and optimised stochastically.

Gal (2016) derives an analytic form for a particular case in which the recognition model contains the prior. Differently form these, we derive a closed-form expression for the KL divergence between two arbitrary Spike and Slab distributions, hence rendering the optimisation of the ELBO for our model of comparable complexity to the standard VAE case.

By solving the KL divergence between the prior of equation 3 and the recognition model of equation 2 we derive with a novel approach the closed-form expression

$$\begin{aligned} &D_{KL}(q_\phi(z|x_i)||q_\phi(z|x_{u^*})) = \\ &= \sum_j^J \Bigg[\gamma_{i,j} \Bigg(\log \frac{\sigma_{z,u^*,j}}{\sigma_{z,i,j}} + \\ &\qquad \frac{\sigma_{z,i,j} + (\mu_{z,i,j} - \mu_{z,u^*,j})^2}{2\sigma_{z,u^*,j}} - \frac{1}{2} \Bigg) + \\ &\qquad (1-\gamma_{i,j}) \log \left(\frac{1-\gamma_{i,j}}{1-\gamma_{u^*,j}} \right) + \gamma_{i,j} \log \left(\frac{\gamma_{i,j}}{\gamma_{u^*,j}} \right) \Bigg]. \end{aligned} \tag{6}$$

A detailed derivation of this expression is reported in supplementary C. This expression naturally presents two components. The first term in the sum is the negative KL divergence between the distributions of the Slab variables, multiplied by the probability of $z_{i,j}$ being non-zero $\gamma_{i,j}$. This component gives a similar regularisation to that of the standard VAE and encourages the Gaussian components of the recognition model to match those of the prior, proportionally to the Spike probabilities $\gamma_{i,j}$. The second term is the negative KL divergence between the distributions of the Spike variables, which encourages the probabilities of the latent variables being non-zero $\gamma_{i,j}$ to match those of the pseudo-input prior $\gamma_{u^*,j}$.

3.4 TRAINING

The VSC model is trained by maximizing the objective function in equation 4 using gradient descent, with the KL divergence of equation 6 and the empirical reconstruction term of equation 5. As other VAE models, VSC suffers from the inherent problem of posterior collapse; during training, certain latent variables tend to store all information about the encoded observations while the remaining ones perfectly satisfy the KL divergence constraint. In a VSC model with a very low prior Spike probability α this results in a dimensionality collapse; some latent variables are set to zero for almost all encodings, while others store most of the information necessary to represent data. This may be desirable in some settings, such as dimensionality selection, but hinders the ability to adequately describe observations with different combinations of recovered features.

Algorithm 1 Training the VSC Model

Inputs: observations $\mathbf{x}$; initial model parameters, $\{\theta^{(0)}, \phi^{(0)}, \omega^{(0)}, \mathbf{x}_u^{(0)}\}$; user-defined latent dimensionality, J; user-defined prior sparsity level, α; user-defined number of warm-up steps, N_{warmup}; user-defined linear increment in warm-up coefficient λ, $\Delta\lambda$; user-defined number of iterations, N_{iter}; user-defined number of samples, L, used in estimating the reconstruction term.

```
$\lambda^{(k=0)} \leftarrow 0$
for *the* $k$'th *iteration* in $[0 : N_{iter} - 1]$
    for *the* $i$'th *observation*
        $z_{i,l} \sim q_{\phi^{(k)},\lambda^{(k)}}(z|x_i) \;\; \forall\, l \in [1 : L]$    $(Eq. 5)$
        $\mathbf{E}_i^{(k)} \leftarrow \sum_l \log p_{\theta^{(k)}}(x_i|z_{i,l})$    $(Eq. 5)$
        $u^* \leftarrow C_{\omega^{(k)}}(x_i)$    $(Eq. 3)$
        $\mathbf{K}_i^{(k)} \leftarrow D_{KL}(q_{\phi^{(k)}}(z|x_i)||q_{\phi^{(k)}}(z|x_{u^*}))$
        $\mathbf{D}_i^{(k)} \leftarrow J \cdot D_{KL}(\overline{\gamma}_{u^*}||\alpha)$    $(Eq. 6)$
    end

    $\mathbf{F}^{(k)} = \sum_i \left( \mathbf{E}_i^{(k)} - \mathbf{K}_i^{(k)} - \mathbf{D}_i^{(k)} \right)$    $(Eq. 4)$
    $\theta^{(k+1)}, \phi^{(k+1)}, \omega^{(k+1)}, \mathbf{x}_u^{(k+1)} \leftarrow \arg\max(\mathbf{F}^{(k)})$
        $(Eq. 4)$
    if $\lambda^{(k)} < 1$ and $k > N_{warmup}$    $(Eq. 7)$
        $\lambda^{(k+1)} \leftarrow \lambda^{(k)} + \Delta\lambda$
    else
        $\lambda^{(k+1)} \leftarrow \lambda^{(k)}$
    end
end
```

To counteract the aforementioned posterior collapse effect, we employ a straightforward Spike variable warm-up strategy. During a pre-training phase, the recognition model $q_\phi(z|x_i)$ is modified as follows:

$$\begin{aligned} q_{\phi,\lambda}(z|x_i) = \prod_{j=1}^J [&\gamma_{i,j}\mathcal{N}(z_{i,j}; \lambda\mu_{z,i,j}, \lambda\sigma^2_{z,i,j} \\ &+ (1-\lambda)) + (1-\gamma_{i,j})\delta(z_{i,j})], \end{aligned} \tag{7}$$

where λ is a constant which is initially set to zero, then linearly increased from zero to one and lastly set equal to one throughout training. When λ is equal to zero, the Slab components of the latent variables is fixed to a zero mean and unit covariance Gaussian. This implies that the recognition model can initially store information only in the Spike variables patterns, similarly to a binary encoding, hence forcing latent vectors from different observations to activate different elements. As λ is increased to one, the model can store more information in the continuous Slab variables, however maintaining different combinations of active latent variables for distinct observations.

In summary, we contribute two main elements of novelty; the derivation of a closed form expression for the Spike and Slab KL divergence and the Spike warm-up strategy to avoid posterior collapse. Both are crucial in handling the mixed continuous-discrete nature of sparse representations and efficiently train the proposed model, hence obtaining sparse probabilistic representation of observations. For completeness we outline the training procedure in Algorithm 1. The values for each user defined parameters we employ in our experiments are reported in supplementary D. The maximisation at each iteration is carried out as an ADAM optimisation step.

4 EXPERIMENTS

4.1 ELBO EVALUATION

We evaluate and compare the ELBO values for VSC and β-VAE models. β-VAEs are VAEs in which the KL divergence term of equation 1 is assigned a controllable weight β. Typically, they are used with a coefficient β greater than one to induce interpretation Higgins et al. (2017a). We make use of the Fashion-MNIST dataset, composed of 28×28 grey-scale images of different pieces of clothing (Xiao et al., 2017), and the UCI HAR dataset, which consists of filtered accelerometer signals from mobile phones worn by different people during common activities (Anguita et al., 2012). In all cases, the latent space is chosen to have 60 latent dimensions and the neural networks of the two models are of equal capacity. For the β-VAE we vary the β coefficient, while for the VSC model we vary the prior spike probability α. Further details can be found in supplementary D.1. Results are shown in figure 1.

Overall, the ELBO values for the VSC model are comparable to those recovered with β-VAEs and a standard VAE ($\beta = 1$). The VSC results in lower ELBO values at decreasing prior spike probability α, similarly to the trend experienced by the β-VAE at increasing KL coefficient β. In fact, the two models present a similar behaviour; as more structure is imposed in the latent space by enforcing a 'stronger' prior, the ELBO is necessarily reduced. The VSC model, however, imposes structure in the latent space through sparsity, rather than an increased weight on the KL divergence term of the ELBO. In the next subsection we show how this subtle difference results into important representation advantages.

4.2 FEATURE DISENTANGLEMENT

We investigate the VSC model's ability to disentangle generating features and align them with the latent space axis. To do so, we make use of an artificial dataset, where the different examples are synthesised from a set of parameters and therefore the generative source features are known and can be used as ground truth. Previous work similarly evaluated disentanglement with artificial samples (Higgins et al., 2017a; Kim & Mnih, 2018). Data sets used in these investigations, however, contain signals generated by altering each feature continuously, leading to all examples expressing variability in all the generating factors. Following a sparse realisation of natural signals, we are instead interested in situations where groups of generating features are present or absent in different combination. To this end, we contribute the Smiley sparse data set, in which four different attributes (mouth, eyes, hat and bow tie) can be present or absent, each with 0.5 probability. Each attribute constitutes a features group and, if present in an example, is controlled by a different number of continuous source features between 3 and 6, for a total of 18 source variables. Examples from the Smiley sparse data set are shown in figure 2, while a detailed description is provided in supplementary E.

In our investigation, we consider both the situation in which the total number of source variables is known and that in which it is unknown. We obtain representations using $60,000$ examples from the Smiley sparse data set with both a β-VAE and a VSC and latent spaces of 18

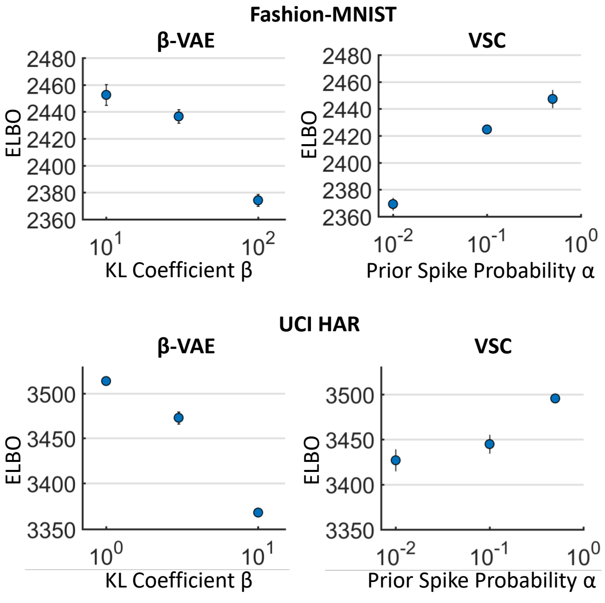

Figure 1: ELBO evaluation for β-VAEs and VSCs (including standard deviations over several repetitions with different random seeds). In both models, the ELBO is reduced by imposing an increasingly more stringent structure in the latent space through the prior, in the β-VAE case through the KL divergence coefficient and in the VSC through the prior sparsity.

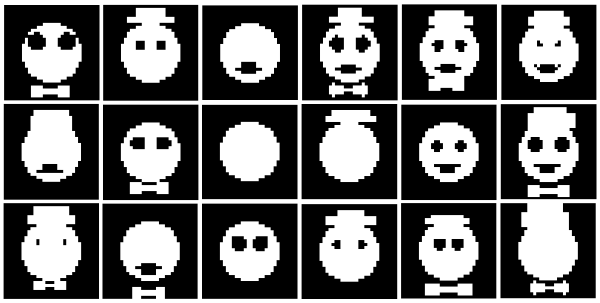

Figure 2: Examples from the Smiley sparse data set. Each is composed from a sparse superposition of 4 different attributes, including mouth, eyes, hat and bow tie.

dimension, for experiments in which the source dimensionality is assumed to be known, and 60 dimensions, to test instead the situation in which knowledge of the source features' number is not available. The VSC model is trained in both instances with a purposely low prior sparsity coefficient $\alpha = 0.01$. In such a way, the prior regularisation encourages almost all variables to be inactive and the model activates only those that are needed to describe each observation.

Both models are given neural networks of equal capacity. We then test features disentanglement with a test set of $20,000$ new samples by computing the absolute value of the correlation between each source feature and each recovered latent variable. Additional details about these experiments can be found in supplementary D.2. Absolute value of correlation matrices for the two models are shown in figure 3. If the number of latent dimensions is chosen to match that of the generating features, the β-VAE displays appreciable correlation between the true attributes and the recovered latent variables. The VSC model displays higher correlation contrast and hence better disentanglement. This is due to the fact that VSC is able to model with its prior both the presence or absence of different features in different observations and their continuous variability, while a β-VAE attempts to model the data only with continuous variables.

As shown in figure 3(c), in the situation where the number of generating dimensions was assumed to be unknown, the β-VAE did not recover latent variables that well correlate with each source feature. β-VAEs encourage encoded data to be distributed as a univariate Gaussian distribution which is factorisable along all its latent space axis. This is effective in situations where the number of latent variables is chosen to match fairly well the number of source features, as shown in figure 3(a), and even more so if such features are all present and continuously distributed in each example, as demonstrated by Higgins et al. (2017a). However, in a more general situation, where data is described by different superpositions

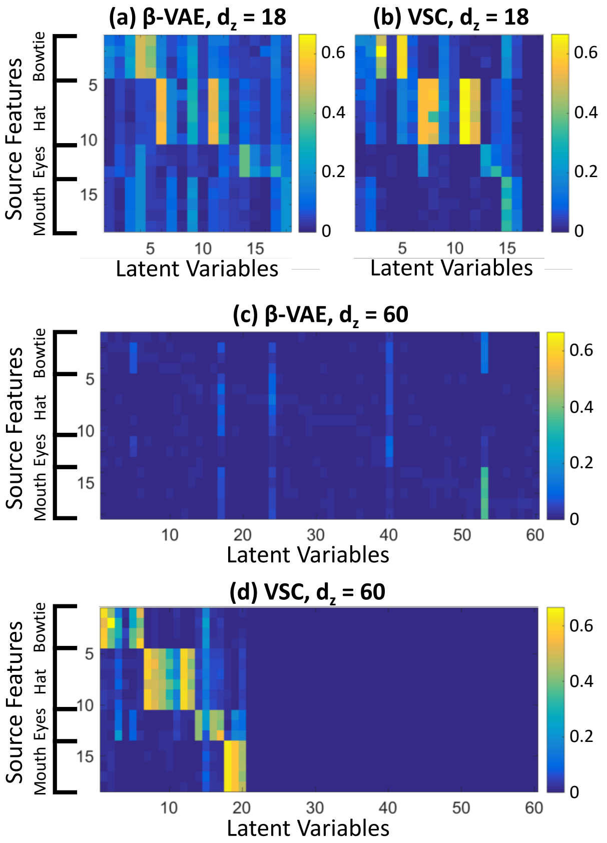

Figure 3: Absolute value of correlation between source features and latent variables for the Smiley data set. The x-axis indicates the index of the latent variable. Variables have been permuted to group dimensions that display the highest correlation with the same feature. Feature disentanglement is achieved if each latent variable correlates strongly with one source attribute but weakly with the remaining ones thus leading to a block diagonal structure.

of features and their number is not known a priori, enforcing a latent distribution that factorises along all axis does not result into good source features recovery; the model forces data to stretch along many more dimensions than it is necessary to describe it, dispersing the correlation with the true sources of variation.

Conversely, as shown in figure 3(d), the VSC model is able to disentangle well the different features groups. Because it is designed to model different combinations of features, it can activate or deactivate latent variables for each encoding according to the features recognised in each observation and it successfully disentangles each attribute in distinct sub-spaces with little interdependence, regardless of the choice of latent space dimensionality. The VSC model adjusted to a suitable number of latent variables needed to represent the data. In the matrix of figure 3(d), the zero columns correspond to collapsed dimensions where the latent variables were consistently inactive. Given 60 latent dimensions prior to training and no knowledge of the source dimensionality or sparsity, the model collapsed to 20 exploited dimensions to describe data generated independently from 18 source variables. The number of latent variables assigned to each feature group is also recovered with fairly good accuracy, as the VSC correlation matrix in figure 3 presents a near-block diagonal structure. We also compare feature disentanglement with the β-TCVAE (Gao et al., 2019) and its behaviour at increasing latent space dimensionality was found to be analogous to that of the $\beta-$VAE. The correlation matrices from these experiments are shown in full in supplementary F.

It is not possible to quantify feature disentanglement in natural data, as the source features are not known. However, similarly to Kim & Mnih (2018), we can qualitatively examine the effect of changing single latent variables on generated samples. To this end, we train a VSC model with $100,000$ examples from the CelebA data set, encode examples from a test set, alter individually exploited dimensions in the latent space and finally generate samples from these altered latent vectors. We find that several of the dimensions exploited by the VSC model control interpretable aspects in the generated data, as shown in the examples of figure 4.

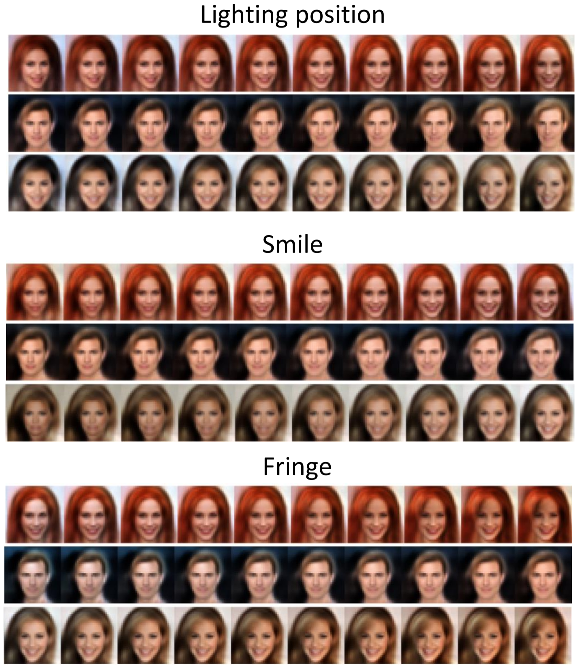

Figure 4: Examples are generated by altering individual latent variables in a VSC model trained on the celebA data set. Individual latent variables control interpretable aspects of the generated images, i.e., interpretable sources of variation are aligned with the representation space axis.

4.3 FEATURE ACTIVATION RECOVERY

The Spike probabilities retrieved when encoding an observation can be interpreted as the probabilities of certain recognised features being present or absent in a given sample. These activation patterns can be used to investigate what features different objects are expected to have in common. As an example, we train two VSC models having 60 latent dimensions with the training sets from the Fashion-MNIST and UCI HAR data sets respectively, encode the entire test sets and examine the

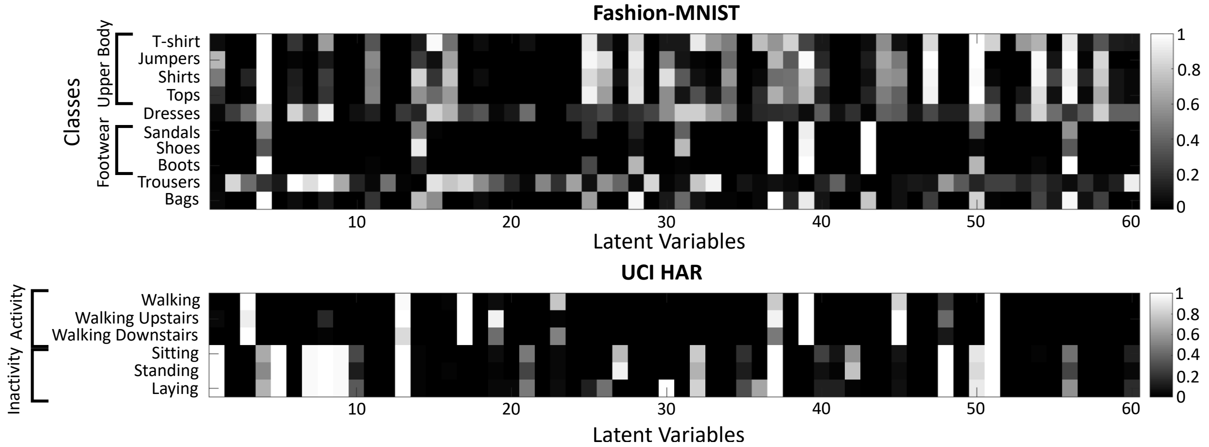

Figure 5: Average Spike probability per class in (a) Fashion-MNIST and (b) UCI HAR. Black corresponds to 0, or always inactive, and white to 1, or always active. The Spike probabilities show the recovered features classes have in common. Objects that are similar, such as T-shirts and shirts in the Fashion-MNIST example, activate mostly the same latent dimensions, i.e., they can largely be described with the same features. More distinct objects, such as T-shirts and trousers, activate different latent dimensions, i.e., they exploit different features to express their variability.

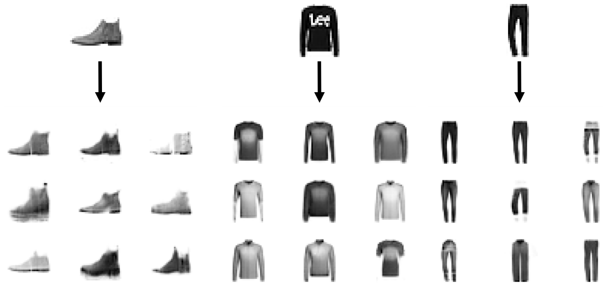

Figure 6: Generation conditioned on single observations with the VSC model. An object is encoded in the latent space and synthetic examples are then generated by sampling only along the activated latent dimensions. This makes it possible to generate a variety of objects that share the same features with the input object without any supervision.

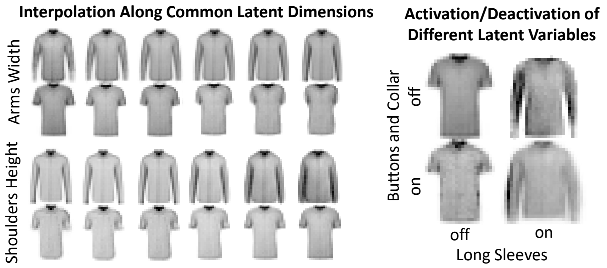
Interpolation Along Common Latent Dimensions
Arms Width
Shoulders Height
Activation/Deactivation of Different Latent Variables
Buttons and Collar
off
on
off
on
Long Sleeves

Figure 7: Investigating the difference between two objects with VSC. Single examples of a T-shirt and a shirt are encoded in the latent space. On the left, generations obtained by individually altering two latent dimensions which are active for both objects. On the right, generations obtained by activating and deactivating latent dimensions exploited by one object, but not the other.

average Spike probabilities per class recovered by the recognition model. Results are shown in figure 5. In the VSC latent space, similar classes present a high overlap of active latent variables, while classes that are very different exploit different sets of latent dimensions to describe their samples. These Spike variable activation patterns readily provide a visualisation of the similarity of different objects in terms of the factors of variation they are described by.

The capability of VSC to recover feature exploitation for a given observation can also be used to control generation. For instance, it is possible to generate samples conditioned on a single object by exploring the subspace defined by the activated latent variables of such object. Figure 6 shows examples of images generated by encoding a single observation from Fashion-MNIST and subsequently sample the subspace defined by the resulting active dimensions. As shown in figure 6, the VSC model naturally provides a way of performing conditional generation without the need for any supervision.

The VSC model can also be used to study the nature of the difference between objects through controlled generation. Two objects may have some active latent variables in common, describing characteristics that they both retain, but that might have different values, and some that are different, describing features that they do not have in common. For instance, in figure 7, we consider the features of a T-shirt and a shirt taken from the Fashion-MNIST test set. The VSC latent variables for the two observations share some dimensions, which we individually alter and generate from. Examples are shown on the left of figure 7. As shown, these dimensions control features the two objects have in common. Conversely, there are some latent dimensions that are active for one observation, but not the other and vice versa. These dimensions correspond to the recovered features the two objects do not share. The effects on generation of activating and deactivating two of these for the T-shirt example are shown on the right of figure 7.

All of the controlled generation examples described above are carried out without any supervision, but simply by examining and individually controlling the latent dimensions activated by single observation examples.

5 CONCLUSION

We present a new model to retrieve non-linear sparse representations of data through a variational auto-encoding approach. The proposed VSC model is capable of retrieving and disentangling sources of variation from diverse data, where attributes can be present and absent in different combinations and the total number of factors of variation and their occurrence is unknown a priori. The sparse representations also offer novel visualisation and generation capability, thanks to the ability to examine and exploit latent variable activation. In defining the VSC model, we also contribute general components and methods that can be used to perform sparse probabilistic inference in different settings, such as an analytical expression for the Spike and Slab KL divergence and a Spike pre-training strategy.

Acknowledgements:
F.T. acknowledges support from Amazon Research and EPSRC grant EP/M01326X/1, B.S.J. from EPSRC grant EP/R018634/1, and R.M-S. EPSRC grants EP/M01326X/1 and EP/R018634/1.

References

D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L Reyes-Ortiz. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *Proceedings of the International Workshop on Ambient Assisted Living*, pp. 216–223. Springer, 2012.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1798–1828, 2013.

C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner. Understanding disentangling in β-VAE. *arXiv:1804.03599*, 2018.

T. Q. Chen, X. Li, R.B. Grosse, and D.K. Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2610–2620, 2018.

A. Cherian and S. Sra. Riemannian dictionary learning and sparse coding for positive definite matrices. *IEEE Transactions on Neural Networks and Learning Systems*, 28(12):2859–2871, 2017.

Y. Gal. *Uncertainty in deep learning*. PhD thesis, University of Cambridge, 2016.

S. Gao, R. Brekelmans, G.V. Steeg, and A. Galstyan. Auto-encoding total correlation explanation. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2019.

I. Goodfellow, A. Courville, and Y. Bengio. Large-scale feature learning with Spike-and-Slab sparse coding. In *Proceedings of the International Conference on Machine Learning*, 2012.

I. Higgins, L Matthey, A Pal, C Burgess, X Glorot, M. Botvinick, S Mohamed, and A Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. In *Proceedings of the International Conference on Learning Representations*, 2017a.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. In *Proceedings of the International Conference on Learning Representations*, 2017b.

J. Ho, Y. Xie, and B. Vemuri. On a nonlinear generalization of sparse coding and dictionary learning. In *Proceedings of the International Conference on Machine Learning*, pp. 1480–1488, 2013.

H. Kim and A. Mnih. Disentangling by factorising. In *Proceedings of the International Conference on Machine Learning*, pp. 2649–2658, 2018.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations*, 2013.

H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. *Neural Computation*, pp. 801–808, 2007.

C. J. Maddison, A. Mnih, and Y. W. Teh. The Concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings of the International Conference on Learning Representations*, 2017.

E. Mathieu, T. Rainforth, N. Siddharth, and Y. W. Teh. Disentangling disentanglement in variational autoencoders. In *Proceedings of the International Conference on Machine Learning*, pp. 4402–4412, 2019.

T. J. Mitchell and J. J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

E. Nalisnick and P. Smyth. Stick-breaking variational autoencoders. In *Proceedings of the International Conference on Learning Representations*, 2017.

B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607, 1996a.

B. A. Olshausen and D. J. Field. Natural image statistics and efficient coding. *Network: Computation in Neural Systems*, 7(2):333–339, 1996b.

B. A. Olshausen and D. J. Field. Sparse coding of sensory inputs. *Current opinion in neurobiology*, 14(4): 481–487, 2004.

Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin. Variational autoencoder for deep learning of images, labels and captions. In *Proceedings of the Advances in Neural Information Processing Systems*, 2016.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the International Conference on Machine Learning*, 2014.

J. T. Rolfe. Discrete variational autoencoders. In *Proceedings of the International Conference on Learning Representations*, 2017.

R. Singh, J. Ling, and F. Doshi-Velez. Structured Variational Autoencoders for the Beta-Bernoulli Process. In *Proceedings of the Advances in Neural Information Processing Systems*, 2017.

C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. How to train deep variational autoencoders and probabilistic ladder networks. In *Proceedings of the International Conference on Machine Learning*, 2016.

M. K. Titsias and M. Lázaro-Gredilla. Spike and Slab variational inference for multi-task and multiple kernel learning. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2339–2347, 2011.

J. M. Tomczak and M. Welling. VAE with a VampPrior. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 1214–1223, 2018.

A. van den Oord, O. Vinyals, and K. Koray. Neural discrete representation learning. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 6306–6315, 2017.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv:1708.07747*, 2017.

S. Yeung, A. Kannan, Y. Dauphin, and L. Fei-Fei. Tackling over-pruning in variational autoencoders. *International Conference on Machine Learning: Workshop on Principled Approaches to Deep Learning*, 2017.

Variational Sparse Coding - Supplementary material

A DETAILS OF THE VSC MODEL

We describe here the details of the VSC model and the architecture of all the neural networks the VSC model is constructed with.

A.1 LIKELIHOOD FUNCTION

The likelihood function $p(x|z_i)$ is composed of a neural network which takes as input a latent variable $z_i \in \mathbb{R}^{J\times 1}$ and outputs the mean $\mu_i \in \mathbb{R}^{M\times 1}$ and log variance $\log(\sigma_i^2) \in \mathbb{R}^{M\times 1}$. The log likelihood of a sample x_i is then computed evaluating the log probability density assigned to x_i by a Gaussian having mean μ_i and standard deviation σ_i. In our experiments we use a one hidden layer fully connected neural network for all experiments. The hidden layer between the latent space and the observation space was chosen to have between $1,000$ and $3,000$ units, depending on the experimental settings.

A.2 RECOGNITION MODEL

The recognition model $p(z|x_i)$ is composed of a neural network which takes as input an observation $x_i \in \mathbb{R}^{M\times 1}$ and outputs the mean $\mu_{z,i} \in \mathbb{R}^{J\times 1}$, the log variance $\log(\sigma_{z,i}^2) \in \mathbb{R}^{J\times 1}$ and the log Spike probabilities vector $\log(\gamma_i) \in \mathbb{R}^{J\times 1}$. The elements of γ_i need to be constrained between 0 and 1, therefore, other than using $\log(\gamma_i)$ as output, which ensures $\gamma_i > 0$, we employ a ReLU non-linearity at this output of the neural network as follows

$$\log(\gamma_i) = -ReLU(-v_{out,i}),$$

where $v_{out,i}$ is output to the same standard neural network that outputs $\mu_{z,i}$ and $\log(\sigma_{z,i}^2)$. This ensures that $\gamma_i < 1$. Samples in the latent space $z_{i,l}$ can then be drawn as detailed in supplementary B.1. The structure of the neural network is analogous to that of the likelihood function, with one hidden layer of $1,000$ to $3,000$ units between the observation space and the latent space.

A.3 SELECTION FUNCTION

The selection function $C_\omega(x_i)$ is composed of a one layer neural network which takes observations x_i as input and returns a vector with the dimensionaliy equal to the number of possible pseudo-inputs $\mathbf{u}$. The output is normalised to unitary sum, then, to encourage the selection of a single pseudo-input while retaining differentiability, the resulting vector is passed through a scaled and displaced Sigmoid function as follows

$$\mathbf{u}^* = S(a(\mathbf{u} - b)),$$

where $\mathbf{u}^*$ is the output selection vector, a is chosen to be equal to 60 in our experiments and b is chosen to be 0.5. The ELBO KL divergence for a given input x_i is then computed as a weighted sum of the KL divergences of the recognition model with each pseudo-input encoding, where the weights are the elements of $\mathbf{u}^*$.

B SPIKE AND SLAB DRAWS REPARAMETRISATION

B.1 REPARAMETRISATION OF THE DRAWS

The draws $z_{i,l}$ are computed as follows

$$z_{i,l} = T(\eta_l - 1 + \gamma_i) \odot (\mu_{z,i} + \sigma_{z,i} \odot \epsilon_l),$$

where $\odot$ indicates an element wise product. The function $T(y_{i,l})$ is in principle a step function centered at zero, however, in order to maintain differentiability, we employ a scaled Sigmoid function $T(y) = S(cy)$. In the limit $c \to \infty$, $S(cy)$ tends to the true binary mapping. In practice, the value of c needs to be small enough to provide stability of the gradient ascent. In our implementation we employ a warm-up strategy to gradually increase the value of c during training.

B.2 SPIKE VARIABLE REPARAMETRISATION

We report here a detailed description of the Spike variable reparametrisation, similar to the relaxation of discrete variables in Maddison et al. (2017) and Rolfe (2017). Our aim is to find a function $f(\eta_{l,j}, \gamma_{i,j})$ such that a binary variable $w_{i,l,j} \sim p(w_{i,l,j})$ drawn from the discrete distribution $p(w_{i,l,j} = 1) = \gamma_{i,j}, p(w_{i,l,j} = 0) = (1 - \gamma_{i,j})$ can be expressed as $w_{i,l,j} = f(\eta_{l,j}, \gamma_{l,j})$, where $\eta_{l,j}$ is some noise variable drawn from a distribution which does not depend on $\gamma_{i,j}$.

The function of choice $f(\eta_{l,j}, \gamma_{i,j})$ should ideally only take values 1 and 0, as these are the only values of $w_{i,l,j}$ permitted by $p(w_{i,l,j})$. Furthermore, the probabilities of $w_{i,l,j}$ being 1 or 0 are linear in $\gamma_{i,j}$, therefore the distribution of the noise variable $\eta_{i,j}$ should have evenly distributed mass. The simplest function which satisfy these conditions and yields our reparametrisation is then a step function $f(\eta_{l,j}, \gamma_{i,j}) = T(\eta_{l,j} - p(w_{i,l,j} = 0)) = T(\eta_{l,j} - 1 + \gamma_{i,j})$ where $\eta_{l,j}$ is uniformly distributed and $T(y)$ is the following step function

$$T(y) = \begin{cases} 1, & \text{if } y \geq 0. \\ 0, & \text{if } y < 0. \end{cases}$$

An illustration of this reparametrisation is shown in figure 8.

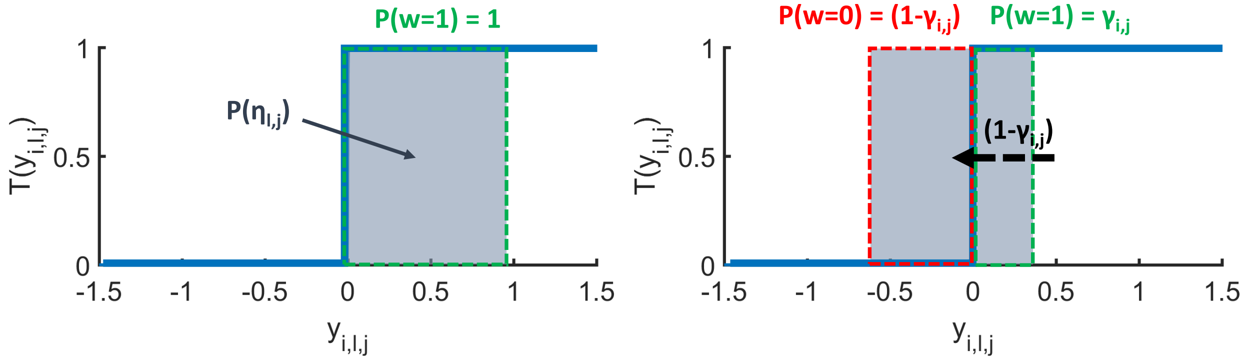

Figure 8: Schematic representation of the reparametrisation of the Spike variable. The variable $y_{i,l,j}$ is drawn in the range covered by the grey square with probability proportional to its height. On the left, for a spike probability $\gamma_{i,j} = 1$, the variable $y_{i,l,j}$ is drawn to always be greater than zero and the Spike variable $w_{i,l,j}$ is always one. On the right, for an arbitrary $\gamma_{i,j}$, the probability density of $y_{i,l,j}$ is displaced to the left by $1 - \gamma_{i,j}$ and $y_{i,l,j}$ has probability $\gamma_{i,j}$ of being ≥ 0, in which case $w_{i,l,j}$ is one, and probability $1 - \gamma_{i,j}$ of being < 0, in which case $w_{i,l,j}$ is zero.

The function $T(y_{i,l,j})$ is not differentiable, therefore we approximate it with a scaled Sigmoid $S(cy_{i,l,j})$, where c is a real positive constant. In our implementation, we gradually increase c from 50 to 200 during training to achieve good approximations without making convergence unstable.

C DERIVATION OF THE SPIKE AND SLAB KL DIVERGENCE

We report here a detailed derivation of the KL divergence between two Spike and Slab distributions shown in equation 6. The KL divergence can be separated into four cross entropy components in each latent dimension

$$
\begin{aligned}
D_{KL}(q_\phi(z|x_i)||q_\phi(z|x_{u^*})) &= \int q_\phi(z|x_i)(\log q_\phi(z|x_i) - \log q_\phi(z|x_{u^*}))dz \\
&= \sum_{j}^{J} \Big[- \underbrace{\gamma_{i,j} \int \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \log \left[\gamma_{u^*,j} \mathcal{N}(z_{i,j}; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j}) + (1-\gamma_{u^*,j})\delta(z_j)\right] dz_j}_{\textcircled{1}} \\
&\quad - \underbrace{(1-\gamma_{i,j}) \int \delta(z_{i,j}) \log \left[\gamma_{u^*,j} \mathcal{N}(z_{i,j}; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j}) + (1-\gamma_{u^*,j})\delta(z_j)\right] dz_j}_{\textcircled{2}} \\
&\quad + \underbrace{\gamma_{i,j} \int \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \log \left[\gamma_{i,j} \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) + (1-\gamma_{i,j})\delta(z_{i,j})\right] dz_j}_{\textcircled{3}} \\
&\quad + \underbrace{(1-\gamma_{i,j}) \int \delta(z_{i,j}) \log \left[\gamma_{i,j} \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) + (1-\gamma_{i,j})\delta(z_{i,j})\right] dz_j}_{\textcircled{4}} \Big].
\end{aligned}
$$

$\textcircled{1}$ and $\textcircled{3}$ are of a similar form; the cross entropy between a Gaussian and a discrete mixture distributions. These components reduce to the corresponding Gaussian-Gaussian entropy terms, as the point mass contributions vanish. In fact, for any finite density distributions $f(z_j)$ and $g(z_j)$, the point mass contribution to the cross entropy between $f(z_j)$ and a discrete mixture $h(z_j) = \alpha g(z_j) + (1-\alpha)\delta(z_j - c)$ is infinitesimal. The proof is as follows; the cross entropy between the functions $f(z_j)$ and $h(z_j)$ is

$$\int f(z_j) \log \left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] dz_j.$$

We can split this integral in two components over two different domains, the first in the region where $z_j \neq c$ and the second in the region where $z_j = c$. By using a Dirac Delta function, the first component can be expressed as follows

$$
\begin{aligned}
&\int_{z_j \neq c} f(z_j) \log \left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] dz_j = \\
&\int_{z_j \neq c} f(z_j) \log \left[\alpha g(z_j)\right] dz_j = \\
&\int \left(1 - \frac{\delta(z_j - c)}{\delta(0)}\right) f(z_j) \log \left[\alpha g(z_j)\right] dz_j,
\end{aligned}
$$

where from the first to the second line we can ignore the component containing $\delta(z_j - c)$, as the domain does not include $z_j = c$. We then use a coefficient which is zero at $z_j = c$ and one otherwise to write the integral over the whole domain of z_j. Similarly, we can write the term in the domain $z_j = c$ as

$$
\begin{aligned}
&\int_{z_j = c} f(z_j) \log \left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] dz_j = \\
&\int \frac{\delta(z_j - c)}{\delta(0)} f(z_j) \log \left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] dz_j,
\end{aligned}
$$

Now combining the two terms we obtain

$$
\begin{aligned}
&\int f(z_j)\log\left[\alpha g(z_j)+(1-\alpha)\delta(z_j-c)\right]dz_j\\
&=\int\Bigg[\left(1-\frac{\delta(z_j-c)}{\delta(0)}\right)f(z_j)\log\left[\alpha g(z_j)\right]+\\
&\quad+\frac{\delta(z_j-c)}{\delta(0)}f(z_j)\log\left[\alpha g(z_j)+(1-\alpha)\delta(z_j-c)\right]\Bigg]dz_j\,.
\end{aligned}
$$

Rearranging to gather the terms in $\delta(z_j-c)/\delta(0)$ we get

$$
\begin{aligned}
&\int f(z_j)\log\left[\alpha g(z_j)\right]dz_j+\\
&\int\frac{\delta(z_j-c)}{\delta(0)}\Bigg[f(z_j)\log\left[\alpha g(z_j)+(1-\alpha)\delta(z_j-c)\right]-f(z_j)\log\left[\alpha g(z_j)\right]\Bigg]dz_j\\
&=\int f(z_j)\log\left[\alpha g(z_j)\right]dz_j+\int\frac{\delta(z_j-c)}{\delta(0)}f(z_j)\log\left[\frac{\alpha g(z_j)+(1-\alpha)\delta(z_j-c)}{\alpha g(z_j)}\right]dz_j\,.
\end{aligned}
$$

Simplifying the argument of the second logarithm and solving the second integral we get

$$
\begin{aligned}
&\int f(z_j)\log\left[\alpha g(z_j)+(1-\alpha)\delta(z_j-c)\right]dz_j\\
&=\int f(z_j)\log\left[\alpha g(z_j)\right]dz_j+\lim_{u\to\infty}\frac{f(c)}{u}\log(1+\frac{1-\alpha}{\alpha}\frac{u}{g(c)}),
\end{aligned}
$$

where the second term tends to zero, leaving the cross entropy between $f(z_j)$ and $\alpha g(z_j)$. Applying this result to $\textcircled{1}$ and $\textcircled{3}$ we obtain the following

$$
\begin{aligned}
\textcircled{3}-\textcircled{1}&=\gamma_{i,j}\int\Bigg[\mathcal{N}(z_{i,j};\mu_{z,i,j},\sigma^2_{z,i,j})\log\left[\gamma_{i,j}\mathcal{N}(z_{i,j};\mu_{z,i,j},\sigma^2_{z,i,j})\right]\\
&\qquad-\mathcal{N}(z_{i,j};\mu_{z,i,j},\sigma^2_{z,i,j})\log\left[\gamma_{u^*,j}\mathcal{N}(z_{i,j};\mu_{z,u^*,j},\sigma^2_{z,u^*,j})\right]\Bigg]dz_j\\
&=\gamma_{i,j}\int\mathcal{N}(z_{i,j};\mu_{z,i,j},\sigma^2_{z,i,j})\log\left[\frac{\gamma_{i,j}\mathcal{N}(z_{i,j};\mu_{z,i,j},\sigma^2_{z,i,j})}{\gamma_{u^*,j}\mathcal{N}(z_{i,j};\mu_{z,u^*,j},\sigma^2_{z,u^*,j})}\right]dz_j\\
&=\gamma_{i,j}D_{KL}\left(\mathcal{N}(z_{i,j};\mu_{z,i,j},\sigma^2_{z,i,j})\ ||\ \mathcal{N}(z_{i,j};\mu_{z,u^*,j},\sigma^2_{z,u^*,j})\right)+\gamma_{i,j}\log\left(\frac{\gamma_{i,j}}{\gamma_{u^*,j}}\right)\,.
\end{aligned}
\tag{8}
$$

The KL divergence $D_{KL}\left(\mathcal{N}(z_{i,j};\mu_{z,i,j},\sigma^2_{z,i,j})\ ||\ \mathcal{N}(z_{i,j};\mu_{z,u^*,j},\sigma^2_{z,u^*,j})\right)$ is the Gaussian-Gaussian KL divergence and has a simple analytic form:

$$
D_{KL}\left(\mathcal{N}(z_{i,j};\mu_{z,i,j},\sigma^2_{z,i,j})\ ||\ \mathcal{N}(z_{i,j};\mu_{z,u^*,j},\sigma^2_{z,u^*,j})\right)=\log\frac{\sigma_{z,u^*,j}}{\sigma_{z,i,j}}+\frac{\sigma_{z,i,j}+(\mu_{z,i,j}-\mu_{z,u^*,j})^2}{2\sigma_{z,u^*,j}}-\frac{1}{2}
\tag{9}
$$

$\textcircled{2}$ and $\textcircled{4}$ take the form of the cross entropy between a Dirac Delta function and a discrete mixture distribution. In this case, instead, the continuous density contributions vanish:

$$\begin{aligned}\textcircled{4} - \textcircled{2} &= (1-\gamma_{i,j}) \int \delta(z_{i,j}) \big(\log \left[\gamma_{i,j}\mathcal{N}(z_{i,j};\mu_{z,i,j},\sigma^2_{z,i,j}) + (1-\gamma_{i,j})\delta(z_{i,j})\right] \\ &\qquad - \log \left[\gamma_{u^*,j}\mathcal{N}(z_{i,j};\mu_{z,u^*,j},\sigma^2_{z,u^*,j}) + (1-\gamma_{u^*,j})\delta(z_{i,j})\right] \big) dz_j \\ &= \lim_{u \to \infty} (1-\gamma_{i,j}) \log \left[\frac{\gamma_{i,j}\mathcal{N}(0;\mu_{z,i,j},\sigma^2_{z,i,j}) + (1-\gamma_{i,j})u}{\gamma_{u^*,j}\mathcal{N}(0;\mu_{z,u^*,j},\sigma^2_{z,u^*,j}) + (1-\gamma_{u^*,j})u} \right] \\ &= (1-\gamma_{i,j}) \log \left(\frac{1-\gamma_{i,j}}{1-\gamma_{u^*,j}} \right).\end{aligned} \tag{10}$$

Substituting the results of equations 8, 9 and 10 into equation 8, we obtain the KL divergence between two general Spike and Slab distributions

$$\begin{aligned}D_{KL}\left(q_\phi(z|x_i)||q_\phi(z|x_{u^*})\right) &= \sum_j^J \left[\textcircled{3} - \textcircled{1} + \textcircled{4} - \textcircled{2} \right] \\ &= \sum_j^J \Bigg[\gamma_{i,j} \underbrace{\log \frac{\sigma_{z,u^*,j}}{\sigma_{z,i,j}} + \frac{\sigma_{z,i,j} + (\mu_{z,i,j} - \mu_{z,u^*,j})^2}{2\sigma_{z,u^*,j}} - \frac{1}{2}}_{\text{Slab KL Divergence}} \\ &\qquad + \underbrace{(1-\gamma_{i,j}) \log \left(\frac{1-\gamma_{i,j}}{1-\gamma_{u^*,j}} \right) + \gamma_{i,j} \log \left(\frac{\gamma_{i,j}}{\gamma_{u^*,j}} \right)}_{\text{Spike KL Divergence}} \Bigg].\end{aligned}$$

This prior term presents two components. The first is the negative KL divergence between the distributions of the Slab variables, multiplied by the probability of $z_{i,j}$ being non-zero $\gamma_{i,j}$. The second term is the negative KL divergence between the distributions of the Spike variables. We find of particular interest that by computing the KL divergence analytically we recover a linear combination of the Spike and Slab components divergences.

D DETAILS OF THE EXPERIMENTS

D.1 ELBO EVALUATION EXPERIMENTAL DETAILS

For the ELBO evaluation experiments, we train identical VSC models for the Fashion-MNIST and UCI-HAR datasets, with the exception of the first layer in the recognition model and the last layer in the likelihood function, as the two data sets have different dimensionality (784 and 561 respectively). The likelihood function takes as input of a fully connected network a latent variable z_i and maps it to a first deterministic layer of $3,000$ dimensions. Two separate fully connected network then map this layer to the observation space mean μ and log variance $\log(\sigma^2)$ respectively. The recognition model takes as input of a fully connected network an observation x_i and maps it to a first deterministic layer of $3,000$ dimensions. Three separate fully connected networks then map this layer to the latent space mean $\mu_{z,i}$, log variance $\log(\sigma^2_{z,i})$ and spike probabilities γ_i respectively. The selection function $u^* = C_\omega(x)$ is composed of a single fully connected layer (linear matrix and ReLu non-linearity) taking as input observations x_i and outputting a selection vector, as described in supplementary A.3. The total number of pseudo-inputs was set to 20.

The models were then trained with the ADAM optimiser in Tensorflow, with a batch size of 500. The spike pre-training, carried out as described in section 3.4, was performed over $15,000$ iterations with $\lambda = 0$. λ was then linearly increased between 0 and 1 over $5,000$ iterations. During this phase, the initial training rate was set to 10^{-3}. The model is then trained further for $50,000$ iterations and an initial training rate of 10^{-4}.

The β-VAE was trained with as an identical structure as possible; The likelihood function was identical to that of the VSC and the recognition model was given the same structure, a side of the fact that there is no mapping to a Spike variable. The β-VAE was trained for $70,000$ iterations with the same batch size and an initial training rate of

10^-4. Each data point in figure 1 is obtained by performing the same experiment five times with different random initialisation and seeds. the points are obtained as the means and the error bars as the standard deviations of the results.

D.2 FEATURES DISENTANGLEMENT EXPERIMENTAL DETAILS

For the feature disentanglement experiments using the Smiley sparse data set we use a VSC and β-VAE identical to those used in the Fashion-MNIST ELBO evaluation, as the two types of data have the same dimensionality. The VSC model is trained with the ADAM optimiser over a total of $200,000$ iterations with a batch size of 500. $50,000$ iterations are dedicated to pre-training, with $\lambda = 0$, then λ is linearly increased to 1 over $10,000$ iterations and the model is then trained with $\lambda = 1$ for the remaining training duration. During the first pre-training $60,000$ iterations, the optimiser is given a step size of 5×10^{-4}, which is then decreased to 5×10^{-5} for the rest of training. The β-VAE was given analogous structure and was trained over $200,000$ iterations and a step size of 5×10^{-5}. The value of β was cross validated between 1 and 50 at increasing steps between 1 and 8 and the model giving the best correlation contrast in the matrices shown was chosen.

The results obtained with the CelebA data set were obtained with the same VSC architecture described above, with the only differences that the observation space is of 3072 dimensions (the CelebA examples were down-sampled to 32×32) and, due to this higher dimensionality, the latent space was given 300 dimensions. The model was trained with the same training parameters described above. However, the total number of iterations was extended to $500,000$, maintaining the same Spike pre-training procedure.

D.3 FEATURES ACTIVATION EXPERIMENTAL DETAIL

The matrices of figure 6 were obtained from the exact models trained for the ELBO evaluations at a prior sparsity of $\alpha = 0.01$. The images shown for the examples of conditional sampling in figure 6 and controlled continuous and discrete interpolation of figure 7 are also generated from the same VSC model, trained with the Fashion-MNIST data set.

E SMILEY DATA SET DETAILS

The Smiley sparse data set is composed of 32×32 binary images of automatically generated smileys. the base of every example is a centered filled circle 10 pixels in radius. Each example is then generated with a sparse superposition of 4 attributes, each defined by a variable number of features. The attribute are assigned a fixed probability of being present or absent in each example. The attributes are the following:

- *Eyes* - The eyes are added as two symmetric circular holes in the circular head and are determined by 3 variables: vertical position, horizontal separation and radius. If eyes are active, each of these features is drawn from a normal distribution with standard deviation of 0.5 pixels.
- *Mouth* - The mouth is added as a central horizontal rectangular hole and two smaller horizontal rectangular holes at its side at the bottom of the head. It is determined by 5 variables: vertical position, horizontal position, vertical width, horizontal length and vertical position of side holes. If mouth is active, each of these features is drawn from a normal distribution with standard deviation of 0.5 pixels.
- *Hat* - The hat is added as a larger rectangle above the smiley's head and a smaller rectangle above it. It is determined by 6 variables: vertical and horizontal position, height and width of larger rectangle, height and width of smaller rectangle. If hat is active, features are drawn from normal distributions with the following standard deviations: 0.5 pixels for the two position variables, 1 pixels for the vertical position of the larger rectangle, 1.5 pixels for the horizontal position of the larger rectangle, 0.5 pixels for the vertical position of the smaller rectangle and 1 pixels for the horizontal position of the smaller rectangle.
- *Bowtie* - The Bowtie is inserted by adding an image of two triangles connected at a corner at the bottom of the smiley's head. It is determined by 4 variables: vertical position, horizontal position, vertical width and horizontal length. If mouth is active, these features are drawn from normal distributions with the following standard deviations: 0.5 pixels for vertical position, 0.25 pixels for horizontal position, 1 pixels for height, and 1.5 pixels for length.

All of the aforementioned standard deviations and other parameters can be altered in the generating code to make different variations of the smiley sparse data set. THe dataset used in our experiments was generated with the parameters detailed above and an attribute presence probability of 0.5.

F ADDITIONAL FEATURE DISENTANGLEMENT RESULTS

We show in figure 9 absolute value of correlation matrices analogous to those shown in figure 3, but for β-VAE, β-TCVAE, and VSC for different choices of latent space dimensionality d_z. As the number of latent dimensions increases, the correlation between ground-truth features and latent variables recovered with the β-VAE and the β-TCVAE decreases, as these unsupervised disentanglement models force to disperse the 18 original generating variables into an increasingly larger number of factors of variation. Conversely, the VSC model maintains good feature disentanglement regardless of latent dimensionality, as the correlation contrast remails strong in all experiments. Furthermore, VSC consistently activates a number of variables which is close to the true number of sources of variation, both in total and for each attribute (zero column indicate latent variables that were never used), as the correlation matrices all present a close to square matrix with block diagonal structure.

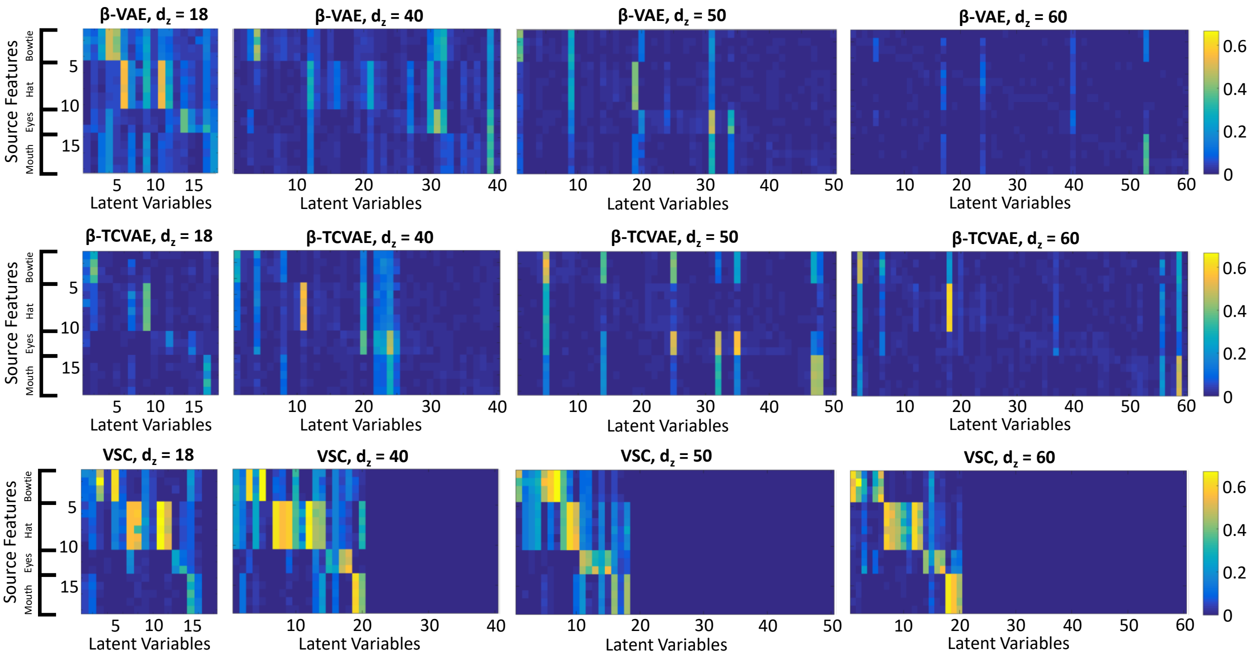

Figure 9: Absolute value of correlation between source features and recovered latent variables with the Smiley sparse data set for multiple choices of latent space dimensionality. While the β-VAE and the β-TCVAE gradually loses their feature disentanglement properties as the number of dimensions is made increasingly different from the number of true sources of variation, the VSC maintains strong disentanglement properties, independently of the choice of dimensionality.

Variational Sparse Coding - Supplementary material

A DETAILS OF THE VSC MODEL

We describe here the details of the VSC model and the architecture of all the neural networks the VSC model is constructed with.

A.1 LIKELIHOOD FUNCTION

The likelihood function $p(x|z_i)$ is composed of a neural network which takes as input a latent variable $z_i \in \mathbb{R}^{J\times 1}$ and outputs the mean $\mu_i \in \mathbb{R}^{M\times 1}$ and log variance $\log(\sigma_i^2) \in \mathbb{R}^{M\times 1}$. The log likelihood of a sample x_i is then computed evaluating the log probability density assigned to x_i by a Gaussian having mean μ_i and standard deviation σ_i. In our experiments we use a one hidden layer fully connected neural network for all experiments. The hidden layer between the latent space and the observation space was chosen to have between $1,000$ and $3,000$ units, depending on the experimental settings.

A.2 RECOGNITION MODEL

The recognition model $p(z|x_i)$ is composed of a neural network which takes as input an observation $x_i \in \mathbb{R}^{M\times 1}$ and outputs the mean $\mu_{z,i} \in \mathbb{R}^{J\times 1}$, the log variance $\log(\sigma_{z,i}^2) \in \mathbb{R}^{J\times 1}$ and the log Spike probabilities vector $\log(\gamma_i) \in \mathbb{R}^{J\times 1}$. The elements of γ_i need to be constrained between 0 and 1, therefore, other than using $\log(\gamma_i)$ as output, which ensures $\gamma_i > 0$, we employ a ReLU non-linearity at this output of the neural network as follows

$$\log(\gamma_i) = -ReLU(-v_{out,i}),$$

where $v_{out,i}$ is output to the same standard neural network that outputs $\mu_{z,i}$ and $\log(\sigma_{z,i}^2)$. This ensures that $\gamma_i < 1$. Samples in the latent space $z_{i,l}$ can then be drawn as detailed in supplementary B.1. The structure of the neural network is analogous to that of the likelihood function, with one hidden layer of $1,000$ to $3,000$ units between the observation space and the latent space.

A.3 SELECTION FUNCTION

The selection function $C_\omega(x_i)$ is composed of a one layer neural network which takes observations x_i as input and returns a vector with the dimensionaliy equal to the number of possible pseudo-inputs $\mathbf{u}$. The output is normalised to unitary sum, then, to encourage the selection of a single pseudo-input while retaining differentiability, the resulting vector is passed through a scaled and displaced Sigmoid function as follows

$$\mathbf{u}^* = S(a(\mathbf{u} - b)),$$

where $\mathbf{u}^*$ is the output selection vector, a is chosen to be equal to 60 in our experiments and b is chosen to be 0.5. The ELBO KL divergence for a given input x_i is then computed as a weighted sum of the KL divergences of the recognition model with each pseudo-input encoding, where the weights are the elements of $\mathbf{u}^*$.

B SPIKE AND SLAB DRAWS REPARAMETRISATION

B.1 REPARAMETRISATION OF THE DRAWS

The draws $z_{i,l}$ are computed as follows

$$z_{i,l} = T(\eta_l - 1 + \gamma_i) \odot (\mu_{z,i} + \sigma_{z,i} \odot \epsilon_l),$$

where $\odot$ indicates an element wise product. The function $T(y_{i,l})$ is in principle a step function centered at zero, however, in order to maintain differentiability, we employ a scaled Sigmoid function $T(y) = S(cy)$. In the limit $c \to \infty$, $S(cy)$ tends to the true binary mapping. In practice, the value of c needs to be small enough to provide stability of the gradient ascent. In our implementation we employ a warm-up strategy to gradually increase the value of c during training.

B.2 SPIKE VARIABLE REPARAMETRISATION

We report here a detailed description of the Spike variable reparametrisation, similar to the relaxation of discrete variables in Maddison et al. (2017) and Rolfe (2017). Our aim is to find a function $f(\eta_{l,j}, \gamma_{i,j})$ such that a binary variable $w_{i,l,j} \sim p(w_{i,l,j})$ drawn from the discrete distribution $p(w_{i,l,j} = 1) = \gamma_{i,j}, p(w_{i,l,j} = 0) = (1 - \gamma_{i,j})$ can be expressed as $w_{i,l,j} = f(\eta_{l,j}, \gamma_{l,j})$, where $\eta_{l,j}$ is some noise variable drawn from a distribution which does not depend on $\gamma_{i,j}$.

The function of choice $f(\eta_{l,j}, \gamma_{i,j})$ should ideally only take values 1 and 0, as these are the only values of $w_{i,l,j}$ permitted by $p(w_{i,l,j})$. Furthermore, the probabilities of $w_{i,l,j}$ being 1 or 0 are linear in $\gamma_{i,j}$, therefore the distribution of the noise variable $\eta_{i,j}$ should have evenly distributed mass. The simplest function which satisfy these conditions and yields our reparametrisation is then a step function $f(\eta_{l,j}, \gamma_{i,j}) = T(\eta_{l,j} - p(w_{i,l,j} = 0)) = T(\eta_{l,j} - 1 + \gamma_{i,j})$ where $\eta_{l,j}$ is uniformly distributed and $T(y)$ is the following step function

$$T(y) = \begin{cases} 1, & \text{if } y \geq 0. \\ 0, & \text{if } y < 0. \end{cases}$$

An illustration of this reparametrisation is shown in figure 8.

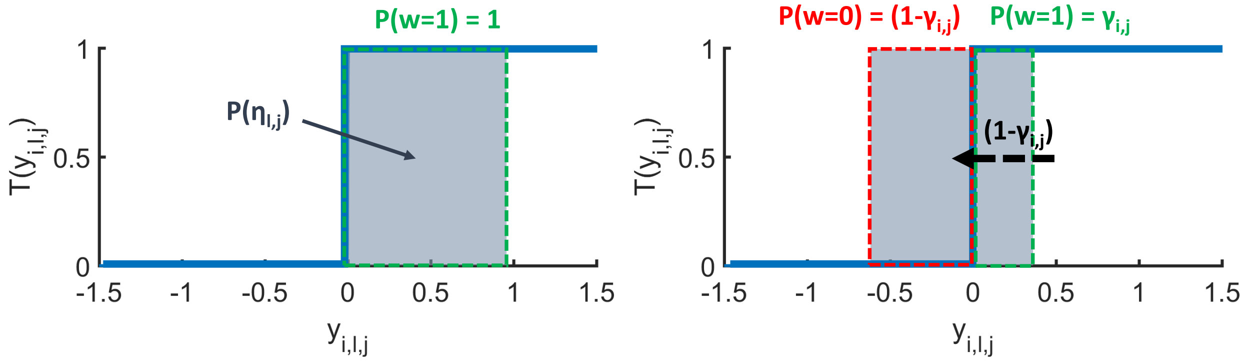

Figure 8: Schematic representation of the reparametrisation of the Spike variable. The variable $y_{i,l,j}$ is drawn in the range covered by the grey square with probability proportional to its height. On the left, for a spike probability $\gamma_{i,j} = 1$, the variable $y_{i,l,j}$ is drawn to always be greater than zero and the Spike variable $w_{i,l,j}$ is always one. On the right, for an arbitrary $\gamma_{i,j}$, the probability density of $y_{i,l,j}$ is displaced to the left by $1 - \gamma_{i,j}$ and $y_{i,l,j}$ has probability $\gamma_{i,j}$ of being ≥ 0, in which case $w_{i,l,j}$ is one, and probability $1 - \gamma_{i,j}$ of being < 0, in which case $w_{i,l,j}$ is zero.

The function $T(y_{i,l,j})$ is not differentiable, therefore we approximate it with a scaled Sigmoid $S(cy_{i,l,j})$, where c is a real positive constant. In our implementation, we gradually increase c from 50 to 200 during training to achieve good approximations without making convergence unstable.

C DERIVATION OF THE SPIKE AND SLAB KL DIVERGENCE

We report here a detailed derivation of the KL divergence between two Spike and Slab distributions shown in equation 6. The KL divergence can be separated into four cross entropy components in each latent dimension

$$
\begin{aligned}
&D_{KL}(q_\phi(z|x_i)||q_\phi(z|x_{u^*})) = \int q_\phi(z|x_i)(\log q_\phi(z|x_i) - \log q_\phi(z|x_{u^*}))dz \\
&= \sum_j^J \Bigg[\underbrace{-\gamma_{i,j} \int \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \log\left[\gamma_{u^*,j}\mathcal{N}(z_{i,j}; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j}) + (1-\gamma_{u^*,j})\delta(z_j)\right] dz_j}_{\textcircled{1}} \\
&\quad \underbrace{-(1-\gamma_{i,j}) \int \delta(z_{i,j}) \log\left[\gamma_{u^*,j}\mathcal{N}(z_{i,j}; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j}) + (1-\gamma_{u^*,j})\delta(z_j)\right] dz_j}_{\textcircled{2}} \\
&\quad \underbrace{+\gamma_{i,j} \int \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \log\left[\gamma_{i,j}\mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) + (1-\gamma_{i,j})\delta(z_{i,j})\right] dz_j}_{\textcircled{3}} \\
&\quad \underbrace{+(1-\gamma_{i,j}) \int \delta(z_{i,j}) \log\left[\gamma_{i,j}\mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) + (1-\gamma_{i,j})\delta(z_{i,j})\right] dz_j}_{\textcircled{4}} \Bigg].
\end{aligned}
$$

$\textcircled{1}$ and $\textcircled{3}$ are of a similar form; the cross entropy between a Gaussian and a discrete mixture distributions. These components reduce to the corresponding Gaussian-Gaussian entropy terms, as the point mass contributions vanish. In fact, for any finite density distributions $f(z_j)$ and $g(z_j)$, the point mass contribution to the cross entropy between $f(z_j)$ and a discrete mixture $h(z_j) = \alpha g(z_j) + (1-\alpha)\delta(z_j - c)$ is infinitesimal. The proof is as follows; the cross entropy between the functions $f(z_j)$ and $h(z_j)$ is

$$\int f(z_j) \log\left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] dz_j.$$

We can split this integral in two components over two different domains, the first in the region where $z_j \neq c$ and the second in the region where $z_j = c$. By using a Dirac Delta function, the first component can be expressed as follows

$$
\begin{aligned}
&\int_{z_j \neq c} f(z_j) \log\left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] dz_j = \\
&\int_{z_j \neq c} f(z_j) \log\left[\alpha g(z_j)\right] dz_j = \\
&\int \left(1 - \frac{\delta(z_j - c)}{\delta(0)}\right) f(z_j) \log\left[\alpha g(z_j)\right] dz_j,
\end{aligned}
$$

where from the first to the second line we can ignore the component containing $\delta(z_j - c)$, as the domain does not include $z_j = c$. We then use a coefficient which is zero at $z_j = c$ and one otherwise to write the integral over the whole domain of z_j. Similarly, we can write the term in the domain $z_j = c$ as

$$
\begin{aligned}
&\int_{z_j = c} f(z_j) \log\left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] dz_j = \\
&\int \frac{\delta(z_j - c)}{\delta(0)} f(z_j) \log\left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] dz_j,
\end{aligned}
$$

Now combining the two terms we obtain

$$
\begin{aligned}
&\int f(z_j) \log\left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] dz_j \\
&= \int \Bigg[\left(1 - \frac{\delta(z_j - c)}{\delta(0)}\right) f(z_j) \log\left[\alpha g(z_j)\right] + \\
&\quad + \frac{\delta(z_j - c)}{\delta(0)} f(z_j) \log\left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] \Bigg] dz_j \, .
\end{aligned}
$$

Rearranging to gather the terms in $\delta(z_j - c)/\delta(0)$ we get

$$
\begin{aligned}
&\int f(z_j) \log\left[\alpha g(z_j)\right] dz_j + \\
&\int \frac{\delta(z_j - c)}{\delta(0)} \Bigg[f(z_j) \log\left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] - f(z_j) \log\left[\alpha g(z_j)\right] \Bigg] dz_j \\
&= \int f(z_j) \log\left[\alpha g(z_j)\right] dz_j + \int \frac{\delta(z_j - c)}{\delta(0)} f(z_j) \log\left[\frac{\alpha g(z_j) + (1-\alpha)\delta(z_j - c)}{\alpha g(z_j)}\right] dz_j \, .
\end{aligned}
$$

Simplifying the argument of the second logarithm and solving the second integral we get

$$
\begin{aligned}
&\int f(z_j) \log\left[\alpha g(z_j) + (1-\alpha)\delta(z_j - c)\right] dz_j \\
&= \int f(z_j) \log\left[\alpha g(z_j)\right] dz_j + \lim_{u \to \infty} \frac{f(c)}{u} \log(1 + \frac{1-\alpha}{\alpha} \frac{u}{g(c)}),
\end{aligned}
$$

where the second term tends to zero, leaving the cross entropy between $f(z_j)$ and $\alpha g(z_j)$. Applying this result to $\textcircled{1}$ and $\textcircled{3}$ we obtain the following

$$
\begin{aligned}
\textcircled{3} - \textcircled{1} &= \gamma_{i,j} \int \Bigg[\mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \log\left[\gamma_{i,j} \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j})\right] \\
&\qquad - \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \log\left[\gamma_{u^*,j} \mathcal{N}(z_{i,j}; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j})\right] \Bigg] dz_j \\
&= \gamma_{i,j} \int \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \log\left[\frac{\gamma_{i,j} \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j})}{\gamma_{u^*,j} \mathcal{N}(z_{i,j}; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j})}\right] dz_j \\
&= \gamma_{i,j} D_{KL}\left(\mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \;||\; \mathcal{N}(z_{i,j}; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j})\right) + \gamma_{i,j} \log\left(\frac{\gamma_{i,j}}{\gamma_{u^*,j}}\right) \, .
\end{aligned}
\tag{8}
$$

The KL divergence $D_{KL}\left(\mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \;||\; \mathcal{N}(z_{i,j}; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j})\right)$ is the Gaussian-Gaussian KL divergence and has a simple analytic form:

$$
D_{KL}\left(\mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) \;||\; \mathcal{N}(z_{i,j}; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j})\right) = \log \frac{\sigma_{z,u^*,j}}{\sigma_{z,i,j}} + \frac{\sigma_{z,i,j} + (\mu_{z,i,j} - \mu_{z,u^*,j})^2}{2\sigma_{z,u^*,j}} - \frac{1}{2} \tag{9}
$$

$\textcircled{2}$ and $\textcircled{4}$ take the form of the cross entropy between a Dirac Delta function and a discrete mixture distribution. In this case, instead, the continuous density contributions vanish:

$$
\begin{aligned}
\textcircled{4} - \textcircled{2} &= (1-\gamma_{i,j}) \int \delta(z_{i,j}) \big(\log \left[\gamma_{i,j} \mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j}) + (1-\gamma_{i,j})\delta(z_{i,j}) \right] \\
&\qquad - \log \left[\gamma_{u^*,j} \mathcal{N}(z_{i,j}; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j}) + (1-\gamma_{u^*,j})\delta(z_{i,j}) \right] \big) dz_j \\
&= \lim_{u \to \infty} (1-\gamma_{i,j}) \log \left[\frac{\gamma_{i,j} \mathcal{N}(0; \mu_{z,i,j}, \sigma^2_{z,i,j}) + (1-\gamma_{i,j})u}{\gamma_{u^*,j} \mathcal{N}(0; \mu_{z,u^*,j}, \sigma^2_{z,u^*,j}) + (1-\gamma_{u^*,j})u} \right] \\
&= (1-\gamma_{i,j}) \log \left(\frac{1-\gamma_{i,j}}{1-\gamma_{u^*,j}} \right).
\end{aligned} \tag{10}
$$

Substituting the results of equations 8, 9 and 10 into equation 8, we obtain the KL divergence between two general Spike and Slab distributions

$$
\begin{aligned}
D_{KL}\left(q_\phi(z|x_i) || q_\phi(z|x_{u^*})\right) &= \sum_j^J \left[\textcircled{3} - \textcircled{1} + \textcircled{4} - \textcircled{2} \right] \\
&= \sum_j^J \Bigg[\gamma_{i,j} \underbrace{\log \frac{\sigma_{z,u^*,j}}{\sigma_{z,i,j}} + \frac{\sigma_{z,i,j} + (\mu_{z,i,j} - \mu_{z,u^*,j})^2}{2\sigma_{z,u^*,j}} - \frac{1}{2}}_{\text{Slab KL Divergence}} \\
&\qquad + \underbrace{(1-\gamma_{i,j}) \log \left(\frac{1-\gamma_{i,j}}{1-\gamma_{u^*,j}} \right) + \gamma_{i,j} \log \left(\frac{\gamma_{i,j}}{\gamma_{u^*,j}} \right)}_{\text{Spike KL Divergence}} \Bigg].
\end{aligned}
$$

This prior term presents two components. The first is the negative KL divergence between the distributions of the Slab variables, multiplied by the probability of $z_{i,j}$ being non-zero $\gamma_{i,j}$. The second term is the negative KL divergence between the distributions of the Spike variables. We find of particular interest that by computing the KL divergence analytically we recover a linear combination of the Spike and Slab components divergences.

D DETAILS OF THE EXPERIMENTS

D.1 ELBO EVALUATION EXPERIMENTAL DETAILS

For the ELBO evaluation experiments, we train identical VSC models for the Fashion-MNIST and UCI-HAR datasets, with the exception of the first layer in the recognition model and the last layer in the likelihood function, as the two data sets have different dimensionality (784 and 561 respectively). The likelihood function takes as input of a fully connected network a latent variable z_i and maps it to a first deterministic layer of $3,000$ dimensions. Two separate fully connected network then map this layer to the observation space mean μ and log variance $\log(\sigma^2)$ respectively. The recognition model takes as input of a fully connected network an observation x_i and maps it to a first deterministic layer of $3,000$ dimensions. Three separate fully connected networks then map this layer to the latent space mean $\mu_{z,i}$, log variance $\log(\sigma^2_{z,i})$ and spike probabilities γ_i respectively. The selection function $u^* = C_\omega(x)$ is composed of a single fully connected layer (linear matrix and ReLu non-linearity) taking as input observations x_i and outputting a selection vector, as described in supplementary A.3. The total number of pseudo-inputs was set to 20.

The models were then trained with the ADAM optimiser in Tensorflow, with a batch size of 500. The spike pre-training, carried out as described in section 3.4, was performed over $15,000$ iterations with $\lambda = 0$. λ was then linearly increased between 0 and 1 over $5,000$ iterations. During this phase, the initial training rate was set to 10^{-3}. The model is then trained further for $50,000$ iterations and an initial training rate of 10^{-4}.

The β-VAE was trained with as an identical structure as possible; The likelihood function was identical to that of the VSC and the recognition model was given the same structure, a side of the fact that there is no mapping to a Spike variable. The β-VAE was trained for $70,000$ iterations with the same batch size and an initial training rate of

10^-4. Each data point in figure 1 is obtained by performing the same experiment five times with different random initialisation and seeds. the points are obtained as the means and the error bars as the standard deviations of the results.

D.2 FEATURES DISENTANGLEMENT EXPERIMENTAL DETAILS

For the feature disentanglement experiments using the Smiley sparse data set we use a VSC and β-VAE identical to those used in the Fashion-MNIST ELBO evaluation, as the two types of data have the same dimensionality. The VSC model is trained with the ADAM optimiser over a total of $200,000$ iterations with a batch size of 500. $50,000$ iterations are dedicated to pre-training, with $\lambda = 0$, then λ is linearly increased to 1 over $10,000$ iterations and the model is then trained with $\lambda = 1$ for the remaining training duration. During the first pre-training $60,000$ iterations, the optimiser is given a step size of 5×10^{-4}, which is then decreased to 5×10^{-5} for the rest of training. The β-VAE was given analogous structure and was trained over $200,000$ iterations and a step size of 5×10^{-5}. The value of β was cross validated between 1 and 50 at increasing steps between 1 and 8 and the model giving the best correlation contrast in the matrices shown was chosen.

The results obtained with the CelebA data set were obtained with the same VSC architecture described above, with the only differences that the observation space is of 3072 dimensions (the CelebA examples were down-sampled to 32×32) and, due to this higher dimensionality, the latent space was given 300 dimensions. The model was trained with the same training parameters described above. However, the total number of iterations was extended to $500,000$, maintaining the same Spike pre-training procedure.

D.3 FEATURES ACTIVATION EXPERIMENTAL DETAIL

The matrices of figure 6 were obtained from the exact models trained for the ELBO evaluations at a prior sparsity of $\alpha = 0.01$. The images shown for the examples of conditional sampling in figure 6 and controlled continuous and discrete interpolation of figure 7 are also generated from the same VSC model, trained with the Fashion-MNIST data set.

E SMILEY DATA SET DETAILS

The Smiley sparse data set is composed of 32×32 binary images of automatically generated smileys. the base of every example is a centered filled circle 10 pixels in radius. Each example is then generated with a sparse superposition of 4 attributes, each defined by a variable number of features. The attribute are assigned a fixed probability of being present or absent in each example. The attributes are the following:

- *Eyes* - The eyes are added as two symmetric circular holes in the circular head and are determined by 3 variables: vertical position, horizontal separation and radius. If eyes are active, each of these features is drawn from a normal distribution with standard deviation of 0.5 pixels.
- *Mouth* - The mouth is added as a central horizontal rectangular hole and two smaller horizontal rectangular holes at its side at the bottom of the head. It is determined by 5 variables: vertical position, horizontal position, vertical width, horizontal length and vertical position of side holes. If mouth is active, each of these features is drawn from a normal distribution with standard deviation of 0.5 pixels.
- *Hat* - The hat is added as a larger rectangle above the smiley's head and a smaller rectangle above it. It is determined by 6 variables: vertical and horizontal position, height and width of larger rectangle, height and width of smaller rectangle. If hat is active, features are drawn from normal distributions with the following standard deviations: 0.5 pixels for the two position variables, 1 pixels for the vertical position of the larger rectangle, 1.5 pixels for the horizontal position of the larger rectangle, 0.5 pixels for the vertical position of the smaller rectangle and 1 pixels for the horizontal position of the smaller rectangle.
- *Bowtie* - The Bowtie is inserted by adding an image of two triangles connected at a corner at the bottom of the smiley's head. It is determined by 4 variables: vertical position, horizontal position, vertical width and horizontal length. If mouth is active, these features are drawn from normal distributions with the following standard deviations: 0.5 pixels for vertical position, 0.25 pixels for horizontal position, 1 pixels for height, and 1.5 pixels for length.

All of the aforementioned standard deviations and other parameters can be altered in the generating code to make different variations of the smiley sparse data set. THe dataset used in our experiments was generated with the parameters detailed above and an attribute presence probability of 0.5.

F ADDITIONAL FEATURE DISENTANGLEMENT RESULTS

We show in figure 9 absolute value of correlation matrices analogous to those shown in figure 3, but for β-VAE, β-TCVAE, and VSC for different choices of latent space dimensionality d_z. As the number of latent dimensions increases, the correlation between ground-truth features and latent variables recovered with the β-VAE and the β-TCVAE decreases, as these unsupervised disentanglement models force to disperse the 18 original generating variables into an increasingly larger number of factors of variation. Conversely, the VSC model maintains good feature disentanglement regardless of latent dimensionality, as the correlation contrast remails strong in all experiments. Furthermore, VSC consistently activates a number of variables which is close to the true number of sources of variation, both in total and for each attribute (zero column indicate latent variables that were never used), as the correlation matrices all present a close to square matrix with block diagonal structure.

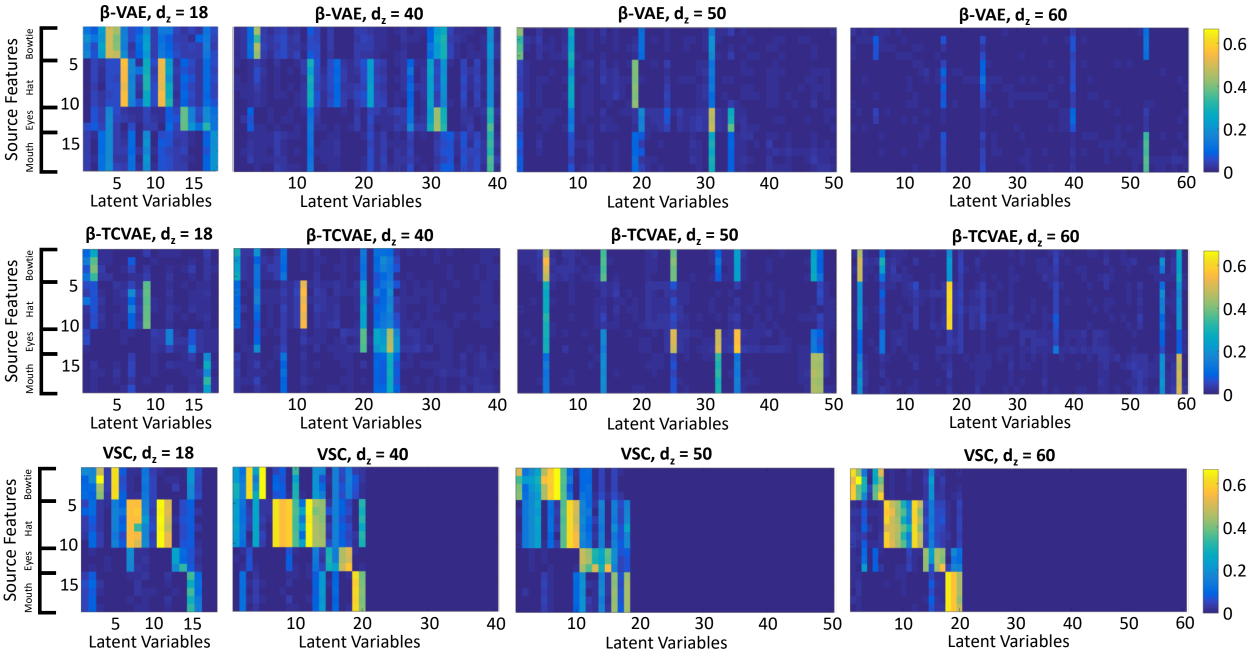

Figure 9: Absolute value of correlation between source features and recovered latent variables with the Smiley sparse data set for multiple choices of latent space dimensionality. While the β-VAE and the β-TCVAE gradually loses their feature disentanglement properties as the number of dimensions is made increasingly different from the number of true sources of variation, the VSC maintains strong disentanglement properties, independently of the choice of dimensionality.