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.

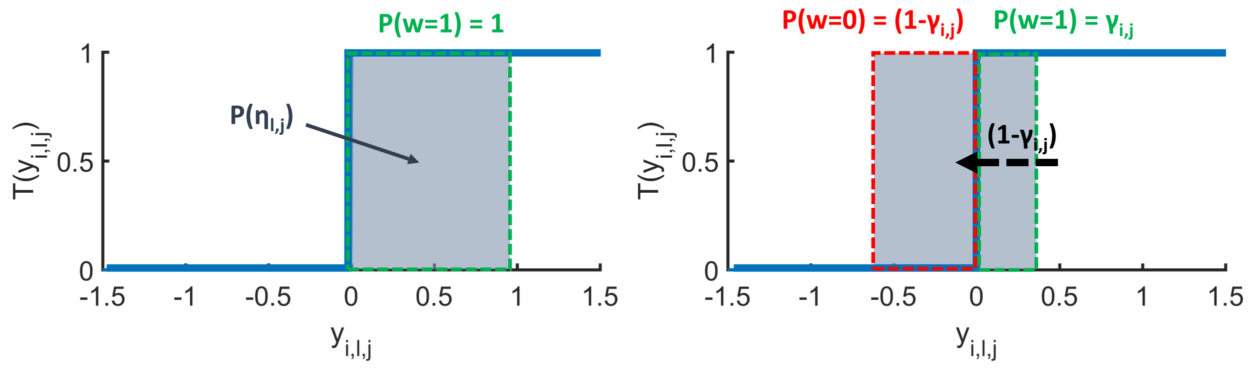

Figure 8: Schematic representation of the reparametrisation of the Spike variable. The variable $y_{i,l,j}$ is drawn in the range covered by the grey square with probability proportional to its height. On the left, for a spike probability $\gamma_{i,j} = 1$, the variable $y_{i,l,j}$ is drawn to always 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

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

① and 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

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

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:

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

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

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

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.

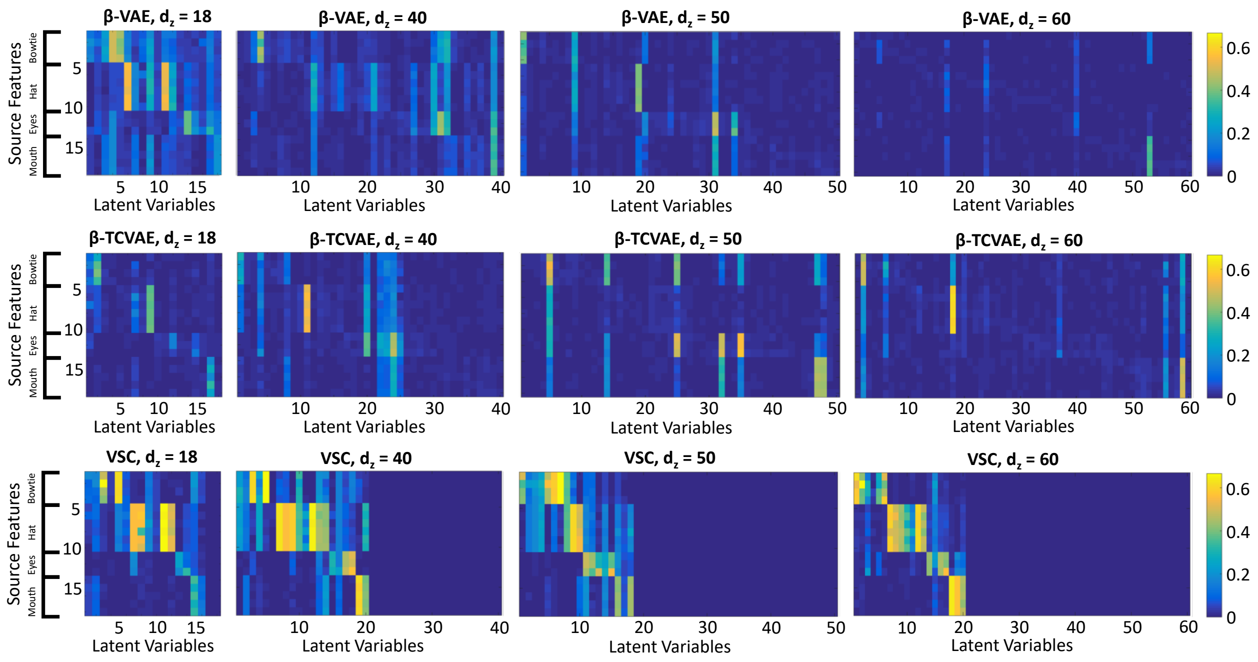

Figure 9: Absolute value of correlation between source features and recovered latent variables with the Smiley sparse data set for multiple choices of latent space dimensionality. While the β-VAE and the β-TCVAE gradually loses their feature disentanglement properties as the number of dimensions is made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