Probabilistic Graphical Models: Capturing Label Characteristics in VAEs.

Hippolyte Pilchen[*], Alexandre Cahill[†], Victor Barberteguy[‡]

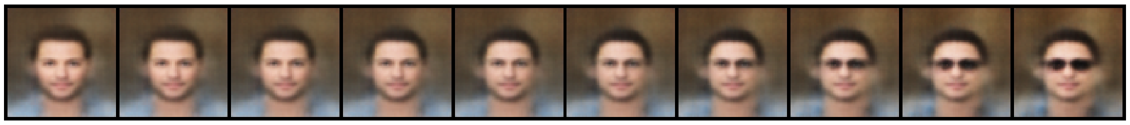
Figure 1: *Conditional generation using CCVAE: man with sun glasses.*

Keywords: Variational Autoencoders, Semi-supervised learning, Image generation, Multi-class attribute

1 Introduction

Learning human-understandable features from images is often a challenging task that is crucial in fields such as robotics or sentiment analysis. On top of that, computing meaningful representation of the data allow to generate new data, and even to modify certain features of a given datapoints, by manipulating the learned representation. In the 2D case, that means we could modify locally the image to change a human-understandable feature.

Using Variational Autoencoders have been a successful approach to tackle this kind of problem, and thus has been widely used. It consists in joint learning of an *encoder* that embeds a meaningful representation of the dataset in a latent space and of a *decoder* that reconstructs the data from the embedding. In a supervised fashion, it has been shown that we could effectively learn a latent space that contains separated areas for each label. It is then possible to operate on a given area of the latent space to change only one precise feature of a given datapoint. [Joy et al. 2022] noted that capturing the representation of the characteristics associated to a label is richer than just capturing the label itself. They introduce **Characteristic Capturing VAE (CCVAE)** that embeds labels characteristics thanks to a new formulation of the VAE framework that is successfully used to manipulate local features on the *CelebA* dataset [Liu et al. 2015]. Throughout this work, we delve into the theoretical details of CCVAE and highlight the divergences with other semi-supervised VAE frameworks such as M2 [Kingma et al. 2014] or DIVA [Ilse et al. 2019]. Then, we retrain the model and provide a clearer visualization of the latent space. We modify the CCVAE framework to work on multiclass labels. However, the performance of this model is not comparable to that of the CVAE paper model, which could be an area for future research. Finally, we assess the quality of the data manipulation using CCVAE by computing keypoints correspondences between original ad reconstructed images with *SuperGlue* model. All the code and necessary resources can be found here[1].

2 Background and related work

2.1 Variational Autoencoders

A latent variable model is a probabilistic model of observations x and unobserved latent variables z. It involves probability distributions to describe the relationships between latent and observed variables. Therefore, we define a probabilistic model $p(x, z)$ of $\mathcal{Z}$-valued latents z, $\mathcal{X}$-valued observes x. A key task in latent variable modeling is to perform inference, which means estimating or learning the values of the latent variables based on the observed data. This can involve techniques such as the expectation-maximization (EM) algorithm or as we do here variational inference. Latent variable models enable dimensionality reduction. By representing data in terms of latent variables, it can simplify complex datasets and capture meaningful patterns with fewer variables.

2.1.1 Variational inference

In situations where calculating or differentiating the integral of the marginal likelihood $p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz$ is intractable, and the exact posterior density $p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)}$ is then also intractable, it becomes impossible to apply the EM algorithm. The marginalization over $\mathbf{z}$ to calculate $p_\theta(x)$ in the denominator is typically intractable, because, for example, the search space of $\mathcal{Z}$ is combinatorially large. Therefore, we seek an approximation, using $q_x(z) \approx p_\theta(z|x)$. These challenges are quite common and often arise in scenarios involving moderately complex likelihood functions $p_\theta(x|z)$, such as in a neural network with a non-linear hidden layer.

Furthermore, in this model, amortized variational inference is used. To sample latent variables, we need parameters for the posterior distribution for each input. If $q_i(z) = N(\mu_i, \sigma_i)$ for every input x_i it requires then to compute a different distribution for each x_i which can be costly if we have a large dataset. The complexity is then $\mathbf{O}(\theta + N * (\mu_i + \sigma_i))$ with N the number of data points. So, we use an approximate distribution $q_\phi(z|x_i)$ with ϕ representing a neural net f_ϕ which takes x_i as an input an returns the parameters of the distribution (*amortized variational inference*). It enables to compute the marginal likelihood through importance sampling. Then, we aim at taking a distribution as close as possible from the ground-truth distribution. Variational Autoencoders (VAE) [Kingma and Welling 2022] are based on the previous observations . They have a strong representation-learning capability and enable to sample from this latent space thanks to the approximation on the posterior distribution.

2.1.2 Objective functions

The variational objective refers to the loss function used to train the VAE. The primary goal of a VAE is to learn a probabilistic generative model of the data. This is achieved by mapping the input data to a lower-dimensional latent space and then generating data from this latent space. To do so, the objective function aims at maximizing the marginal likelihood which can be rewritten as a sum of a Kullback-Leibler divergence and another term which is called the *variational lower bound* [Kingma and Welling 2022]. This bound which will be maximized can be split in the two following terms:

[*]hippolyte.pilchen@polytechnique.edu
Department of Applied Mathematics, ENS Paris-Saclay
[†]Alexandre.Cahill@polytechnique.edu
Department of Applied Mathematics, ENS Paris-Saclay
[‡]victor.barberteguy@polytechnique.edu
Department of Computer Science, Ecole Polytechnique

[1]`https://github.com/VictorBbt/ccvae_pytorch`

Reconstruction Loss : This term measures how well the VAE can reconstruct the input data from the latent representation. It measures the mismatch between an observation x and its reconstruction obtained following $p_\theta(x|z)$ with z sampled from $q_\phi(z|x)$. This term encourages the VAE to generate data that is similar to the input data.

Regularization (KL Divergence Term): This term is derived from the Kullback-Leibler (KL) Divergence and serves as a regularization term. It forces the distribution of the latent variables to be as close as possible to the chosen prior distribution. This encourages the latent space to have a certain structure and makes the model more interpretable. This quantity is computed between the approximat posterior and the prior.

$$D_{KL}(q_\phi(z|x)|p_\theta(z)) = -\sum_x q_\phi(z|x)log(\frac{q_\phi(z|x)}{p_\theta(z)}) \quad (1)$$

The combination of these two terms in the loss function creates the variational objective. By optimizing this objective during training, the VAE learns to encode data into a meaningful latent space, making it suitable for tasks like data generation, denoising, and data representation. The regularization term, in particular, helps in disentangling the representations in the latent space, making it a useful tool for unsupervised feature learning and generative modeling.

2.2 Semi-supervised VAE

However, these objective functions can be used in an unsupervised setting only. The paper of interest [Joy et al. 2022] focuses on a different setting: some data ($\mathcal{S}$) has labels y which the model should learn while learning the latent representation of the whole dataset ($\mathcal{D}$). Then, the log-likelihood becomes:

$$\sum_{(x,y)\in\mathcal{S}} \log p_\theta(y,x) + \sum_{(x)\in\mathcal{D}\setminus\mathcal{S}} \log p_\theta(x)$$

This framework enables to perform classification tasks but also as a foundation for acquiring meaningful representations while enabling to perform manipulations on generated data labels. This involves leveraging the decoder's capabilities to generate new datapoints by intervening on the labels through alterations of the latent representation of data with a specific label.

In the paper [Kingma et al. 2014] the M2 model is the first to include supervision in VAE's. It adds the label to the latent representation as a categorical label. For unlabeled data, an auxiliary classifier learns which categorical variable to put at the end of the latent representation for each data. However, the data associated with a label contains richer information than what is represented by the categorical label itself. Characteristics specific to the label and information about the rest of the image are *entangled* in the representation vector z. Therefore, the model struggles to learn features that belong to the label and those that belong to the rest of the image with a latent representation of shape (z, y).

DIVA model [Ilse et al. 2019] tackles domain generalization and partially solves the previous problem. The latent space is now based on three representation vectors z_y, z_d and z_x which respectively represents features associated with the label, the domain and the rest of the input image x. Therefore, labels are not treated as categorical data in the latent space anymore but latent vectors are learned from them. This enables to capture information specific to the label in z_y while allowing to perform characteristics manipulation. By traversing the distribution of z_y for one label, different images with the same label can be generated. Intervention on labels no longer boils down to binary manipulations, but enables smooth changes from one label to another (for example, you can switch from a smiling face to one that isn't smiling through intermediate shapes. Nevertheless, the main goal being to learn a domain invariant representation, conditional inference with this model remains complex.

The paper of interest [Joy et al. 2022] proposes a similar approach to the previous one while solving inference issues. CCVAE model facilitates classification, conditional generation and intervention tasks

3 Characteristic capturing Variational Autoencoders

3.1 The method of interest: CCVAE

The Characteristic Capturing VAE maps the input in a latent space where representation vectors are designed as follows: $\{z_c, z_c\}$. z_c is the *characteristic latent* which encapsulates features strictly related to a precise label and z_c captures all the rest (information about the image not related to any label characteristic). Furthermore, different labels are disentangled by taking a factorized label predictive distribution: $q_\varphi(y|z_c) = \Pi_i q_{\varphi^i}(y^i|z_c^i)$, so the latent vectors associated to labels are partitioned because they depend on different distributions according to their label.

To facilitate inference of data with a particular label, a conditional prior is introduced $p_\psi(z_c|y)$. As for the approximate posterior, the conditional prior is designed as a factorized set of generative models $\Pi_i p_{\psi^i}(z_c^i|y^i)$. The graphical model is described in Figure . By incorporating these particularities into the *variational lower bound* 2 a natural classifier appears in the supervised setting unlike DIVA [Ilse et al. 2019] and M2 [Kingma et al. 2014] models where a classifier is added to the objective function.

– Unsupervised lower bound:

$$\mathbb{E}_{q_\phi(z|x)}\left[\frac{q_\varphi(y|z_c)}{q_{\varphi,\phi}(y|x)}\log\frac{p_\theta(x|z) * p_\psi(z|y)}{q_\varphi(y|z_c)q_\phi(z|x)}\right]$$
$$+\log q_{\varphi,\phi}(y|x) + \log p(y) \quad (2)$$

– Supervised lower bound:

$$\mathbb{E}_{q_\phi(z|x),q_\psi(y|z_c)}\left[\log\left(\frac{p_\theta(x|z)p_\psi(z|y)p(y)}{q_\varphi(y|z_c)q_\phi(z|x)}\right)\right] \quad (3)$$

In the unsupervised setting, the evidence lower bound (ELBO) is derived using Jensen inequality another time to avoid marginalizing over labels for the prior distribution. It comes down to the unsupervised lower bound as defined in 3. All the calculations to derive the final lower bound on the log-likelihood has not been written in this report since the calculations appear in the studied paper [Joy et al. 2022].

Finally, the sum of these two quantities for all data points (with or without labels) is maximized during learning in order to maximize the marginal log-likelihood of this probabilistic model.

In the paper, several distributions are used to pass from the latent to the input space but also from the latent to the label space and vice-versa. We define and summarize all the distributions here and underline the distributions chosen in the implementation.

- Generative model: $p_\theta(x|z)$, Laplace distribution with the output of the decoder as the mean and a matrice full of one as the scale [Higgins et al. 2017];

- Approximate posterior: $q_\phi(z|x) = \mathcal{N}(z_c, z_{\setminus c}|\mu_\phi(x), \text{diag}(\sigma^2_\phi(x)))$, with $\mu_\phi(x)$ and $\text{diag}(\sigma^2_\phi(x))$ learned with CNN's, it enables image reconstruction;
- Label predictive distribution: $q_\varphi(y|z_c)$ following a $Ber(y|\pi_\varphi(z_c))$ law, with $\pi_\varphi(z_c)$ a diagonal transformation to impose factorization $q_\varphi(y|z_c) = \prod_{i=1}^n q_{\varphi_i}(y_i|z_{c_i})$;
- Conditional prior : $p_\psi(z_c|y) = \mathcal{N}(z_c|\mu_\psi(y), \text{diag}(\sigma^2_\psi(y)))$, which induces the same factorization as above and which parameters are represented by an MLP. The continuity of this distribution is of fundamental importance for the intervention task;
- Prior placed on unlabeled data: $p(z_{\setminus c}) = \mathcal{N}(z_{\setminus c}|0, I)$.

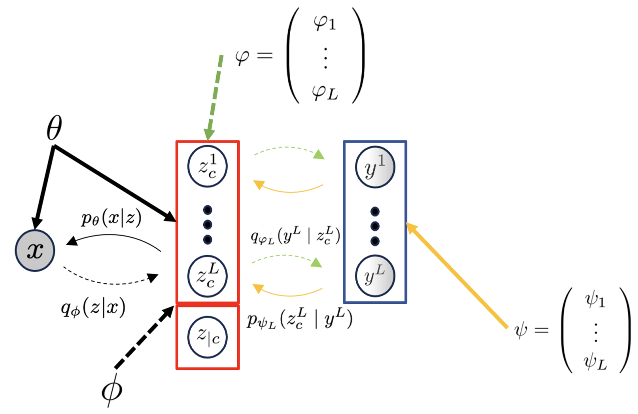
$\varphi = \begin{pmatrix}\varphi_1\\ \vdots \\ \varphi_L\end{pmatrix}$
θ
$p_\theta(x|z)$
x
$q_\phi(z|x)$
z_c^1
z_c^L
$z_{\setminus c}$
$q_{\varphi_L}(y^L \mid z_c^L)$
$p_{\psi_L}(z_c^L \mid y^L)$
y^1
y^L
$\psi = \begin{pmatrix}\psi_1\\ \vdots \\ \psi_L\end{pmatrix}$
ϕ

Figure 2: *The type of directed graphical model under consideration for the CCVAE. Solid black lines denote the generative model $p_\theta(z)p_\theta(x|z)$, dashed black lines denote the variational approximation $q_\phi(z|x)$ to the intractable posterior $p_\theta(z|x)$. Orange solid lines denote factorized conditional priors $p_\psi(z_c|y)$ (along with a prior $p(y)$ to perform inference on unlabeled data), which are generative models useful to train unlabeled data points. Green dashed lines define factorized label predictive distributions. Symbols outside boxes and bubbles define learned parameters for each distribution. Solid lines define distributions used during the generation process and dashed lines during inference.*

3.2 Comparison with previous Semi-supervised VAE

3.2.1 M2 [Kingma et al. 2014]

The M2 Model is the first VAE model to perform semi-supervision. However, the major drawback is that the class labels are directly used as a latent class variable. Therefore, there is no latent variable which encapsulates the characteristics associated to a certain label only a categorical latent class variable. This limits the possibility to perform conditional generation, it also reduces the possible interventions to binary cases (switching the label on and off) in the binary label case.

Generative process:

- Likelihood function, with f a Gaussian or Bernouilli distribution: $p_\theta(x|y,z) = f(x; y, z, \theta)$;
- $p(z) = \mathcal{N}(z|0, I)$;
- For unlabeled data, y is treated as latent variables: $p(y) = \text{Cat}(y|\pi)$.

Inference model for the latent variable z and y with $z \perp y \mid x$:

- Approximate posterior distribution: $q_\phi(z|y,x) = \mathcal{N}(z|\mu_\phi(y,x), \text{diag}(\sigma^2_\phi(x)))$;
- $q_\phi(y|x) = \text{Cat}(y|\pi_\phi(x))$

All the laws parameters are represented as MLPs.

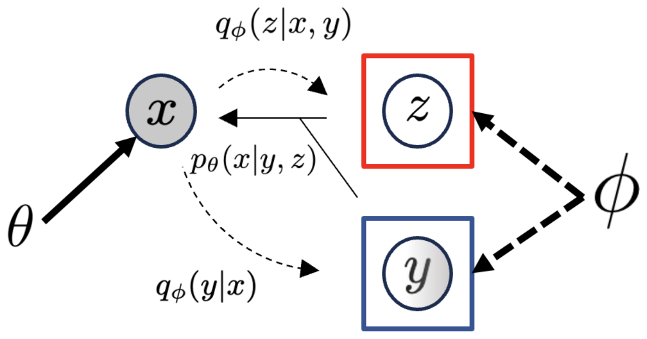

Figure 3: *The type of directed graphical model under consideration for the M2 model. Solid black lines denote the generative model $p_\theta(z)p_\theta(y)p_\theta(x|y,z)$, dashed black lines denote the variational approximation $q_\phi(z|x,y)$ to the intractable posterior $p_\theta(z|x,y)$ and $q_\phi(y|x)$ for y. Symbols outside boxes and bubbles define learned parameters for each distribution. Solid lines define distributions used during the generation process and dashed lines during inference. During generation, y and z are sampled from categorical and normal distributions.*

In the appendix A.1, we have derived the objective function in detail from the variational inference principles. It is given by the supervised and unsupervised lower bounds :

$$\mathcal{L}_s(x,y) = \mathbb{E}_{q_\phi(z|x,y)}[\log \frac{p(x|y,z)p(y)p(z)}{q(z|x,y)}]$$
$$\mathcal{L}_u(x) = \sum_y q_\phi(y|x)(\mathcal{L}(x,y)) - H(q_\phi(z|x))$$
$$\mathcal{U}(x) = \mathcal{L}_s(x,y) + \mathcal{L}_u(x) + \alpha\mathbb{E}_{p_\theta(x,y)}[-\log q_\phi(y|x)]$$

Beyond the limits of the model exposed above, this construction of the objective function also implies that 1. The latent space is not partitioned according to each label making it *entangled* and 2. In the supervised case, x and y are given so $q_\phi(x|y)$ does not update ϕ and is not learned. Hence an additional term $\alpha\mathbb{E}_{p_\theta(x,y)}[\log q_\phi(y|x)]$. This adds potential hyperparameters.

3.2.2 Domain Invariant Variational Autoencoders [Ilse et al. 2019]

Here, the task is slightly different since the model is designed to map the input into domain invariant latent variables. The data is divided into different sets with similar distributions called domains. Inside each domain data points have class labels. Therefore, there are three latent variables z_d which captures domain-specific information, z_c which encapsulates the characteristics related to the labels, z_x which captures any residual variations left in x the equivalent of $z_{\setminus c}$.

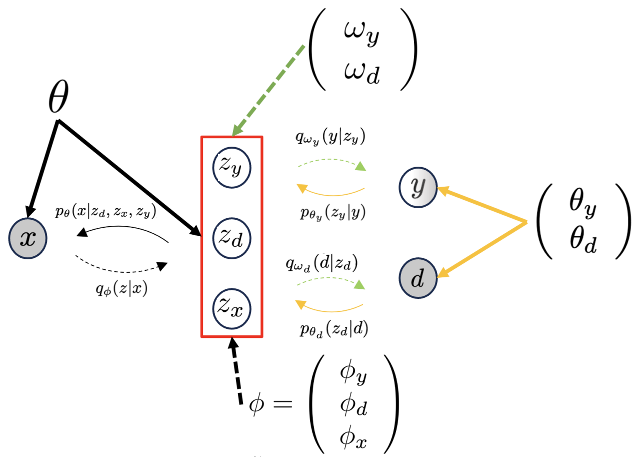
θ
$\begin{pmatrix}\omega_y\\ \omega_d\end{pmatrix}$
$q_{\omega_y}(y|z_y)$
$p_{\theta_y}(z_y|y)$
$q_{\omega_d}(d|z_d)$
$p_{\theta_d}(z_d|d)$
$p_\theta(x|z_d,z_x,z_y)$
$q_\phi(z|x)$
x
z_y
z_d
z_x
y
d
$\begin{pmatrix}\theta_y\\ \theta_d\end{pmatrix}$
$\phi = \begin{pmatrix}\phi_y\\ \phi_d\\ \phi_x\end{pmatrix}$

Figure 4: *The type of directed graphical model under consideration for the DIVA model. Solid black lines denote the generative model $p_\theta(z)p_\theta(x|z)$, dashed black lines denote the variational approximation $q_\phi(z|x) = q_{\phi_y}(z_y|x)q_{\phi_d}(z_d|x)q_{\phi_x}(z_x|x)$ to the intractable posterior $p_\theta(z|x)$, where z is (z_y, z_x, z_d). Orange solid lines denote auxiliary classifiers (one for labels and one for domains). Green solid lines define factorized conditional prior distributions. Symbols outside boxes and bubbles define learned parameters for each distribution. Solid lines define distributions used during the generation process and dashed lines during inference.*

Solid lines: Generative model. According to the graphical model, we obtain $p(d, x, y, z_d, z_x, z_y) = p_\theta(x|z_d, z_x, z_y)p_{\theta_d}(z_d|d)p(z_x)p_{\theta_y}(z_y|y)p(d)p(y)$.

Dashed lines: Inference model. We propose to factorize the variational posterior as $q_{\phi_d}(z_d|x)q_{\phi_x}(z_x|x)q_{\phi_y}(z_y|x)$. Dashed arrows represent the two auxiliary classifiers $q_{\omega_d}(d|z_d)$ and $q_{\omega_y}(y|z_y)$.

· Conditionnal priors with learnable parameters: $p_{\theta_d}(z_d|d)$ and $p_{\theta_y}(z_y|y)$;

· Gaussian prior: $p(z_x)$;

· Three separate variational posterior distributions (encoders), fully factorized Gaussians with parameters given by $MLP(x)$: $q_{\phi_d}(z_d|x)$, $q_{\phi_x}(z_x|x)$ and $q_{\phi_y}(z_y|x)$.

· For unlabelled data: $q_{\omega_y}(y|z_y)$

The authors have adapted the DIVA framework to domain agnostic spaces by removing the domain characteristic in labels and from the latent space. The latent space for the labels is also partitioned to match the approach in CC-VAE. Both DIVA and Adapted DIVA are fully derived in the Appendix A.2. The final objective function for Adapted DIVA exhibits two major differences with the CCVAE model :

1. In the supervised case, similarly to the M2 model, an external classifier $\alpha_y \mathbb{E}_{q_\phi(z_y|x_i)}(log(q_{\omega_y}(y_i|z_y)))$ has to be added to the objective function to learn from the supervised cases as well. This is handled with in CC-VAE with the natural classifier

2. In the unsupervised case, y is imputed through the $q_{\omega_y}(y|z_y)$

In both cases having this external classifier increases the number of hyperparameters to the model.

4 Experiments

In this part, we carry out several experiments in order to underline the various advantages of Characteristic Capturing VAE described in the studied paper, but above all to test its limitations. We use Tom Joy's code, one of the author of the paper, to carry out our various experiments. However, due to dataloader issues we had to modify several lines of code and fix a few problems. Then, we wrote additional scripts and notebooks to carry out our experiments. We mainly used *PyTorch*[2] library. We add features to the existing dataloader such as pruning which enables to reduce the size of the training dataset. First, CCVAE model has been trained on the entire *CelebA*[3] dataset for 80 epochs with 10% of supervision (which means that 10% of the images are labelled).

4.1 Latent representation

For this experiment, we seek to visualize the learned embeddings in the latent space. Our latent space has total dimension 45 but we kept the latent space corresponding to each label to be 1-dimensional. Therefore we can simply plot the z_{c_i} corresponding to label i for a batch of images. If the latent space would have been larger, we could have also used PCA to reduce the space to 2D for visualization purposes.

We have also conducted latent traversal experiments. The simplified approach adopted is to take for a given characteristic, the mean of the distribution for label 0 and label 1. Construct the straight line between these two points and interpolate it by sampling it with fixed-length steps between the two points. Then generate the images corresponding to the latent representation of the original image but locally perturbed for the label corresponding to the characteristic.

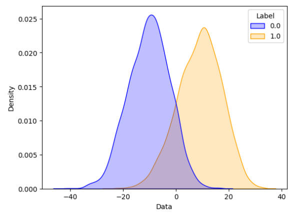

Figure 5: *Latent space embedding for smiling label*

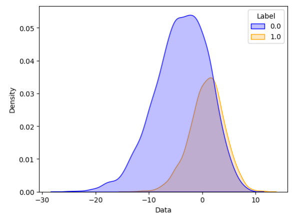

Figure 6: *Latent space embedding for arched eyebrows label*

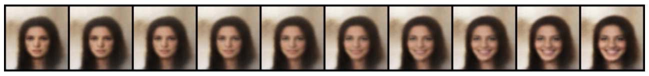

Figure 7: *Latent traversal for smiling label*

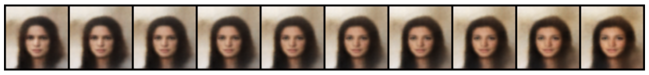

Figure 8: *Latent traversal for arched eyebrows label*

Figure 9: *Figures 5 and Figures 6 illustrate the distributions of the embedding corresponding to "smiling" and "arched eyebrows" respectively for 10000 labeled images. We can observe that the model has learned a much clearer representation for the 'smiling' attribute where the densities for label 0 and 1 are distinguishable and clustered. This is not the case for the 'arched eyebrows' label. This implies that the latent traversal for the smiling attribute is much clearer and significant than for the 'arched eyebrow' where the characteristic has been poorly captured.*

[2]`https://pytorch.org`
[3]`https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html`

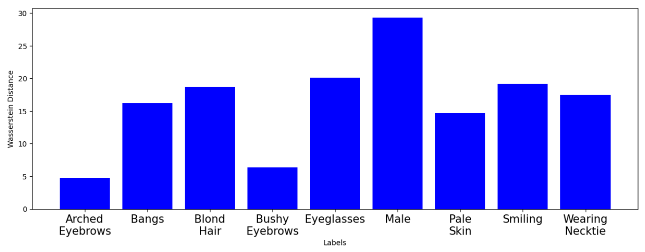

Figure 10: *Wasserstein distance between the distribution for labels 0 and labels 1. The higher the distance, the better the two labels are disentangled and the better conditional generation results we will get*

We can observe that the 1D latent embeddings perform particularly poorly on labels that have been binarized but should be multi-label. For instance the model learns poorly to classify brown hair and black hair. This gave us the idea to implement a multiclass attribute for hair (containing the attributes 'Bald','Black Hair','Blond Hair','Brown Hair','Gray Hair'). Our intuition here is that explicitely creating binary labels was an intention of the authors to show that it is possible to learn embeddings for each label and manipulate them. However it might come at the cost of poorer performance.

4.2 Multi-class attributes

Thanks to the various insights we have gained from our previous analysis of latent variables and our experience with the dataset (human faces), we now modify the model proposed in the paper so that it adapts to non-binary labels, i.e. multi-class labels. The different modifications and the new architecture is described in the Appendix C.1

Since the task is now more complex we set the supervision rate to 90%. We have departed from the setting of the previous experiment. As the task is more complex, we obtained poorer results with a supervision rate of only 10%. This can be explained by the need for more observations if we are to have enough examples for each of the classes of the multi-class label.

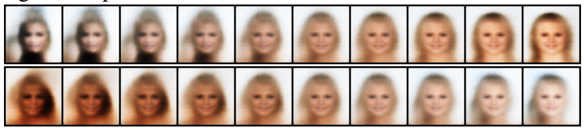

Figure 11: *Latent traversal for binary attributes with the modified CCVAE trained with a multi-class label ('Hair_MULTI'). First row: traversal of the latent space for smiling attribute. Second row: traversal of the latent space for pale skin attribute.*

Firstly, with multi-class labels, CCVAE generates fuzzy reconstructions, but their quality is not that different from that of standard CCVAE trained for the same number of epochs (see C.2 for the comparison). The modified CCVAE partitions well the latent space for binary attributes. However, interventions on these labels also affect other parts of the image, the feature associated to the label can be modified but the entire reconstruction is then affected, as in Figure 11. This multi-class latent representation breaks the partitioning between z_c and z_c. In addition, the coordinate of the latent which should encapsulate features associated with hair attributes fails to partition its space into 5 parts (one per class). Indeed, the five gaussians generated and learned by the conditional prior multi-class module have close means (between 0 and 1) and a high variance (≈ 1). Therefore, the conditional distributions are overlapping. This gathering of the distributions leads to a difficultly interpretable latent space. The latent traversal is not able to perform label intervention properly. This latent coordinate does not even seem to be linked to hair features as shown in Figure 12.

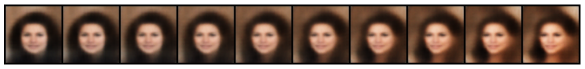

Figure 12: *Latent traversal in the space associated with multi-class hair label.*

Even though adding the possibility to learn from multi-class attributes has not changed much how the model performs on binary attributes, the new latent coordinate which represents the multi-class label fails to encapsulate the features associated with this label. Although, this modified version may be flawed because of our implementation or our choices of architecture for this new model, we have tried several methods and we have made sure that the modules we have added learn the same way as the original CCVAE from the paper. Therefore, enabling CCVAE to deal with multi-class labels and binary class labels remains a tough unsolved problem (different from the multi-class setting in the paper where there is no other attribute than the class) which may require more complex structure.

We have also tried an hybrid training by training for 40 epochs with different supervision rates (first 40 epochs with $f = 0.9$ then $f = 0.5$ and finally $f = 0.1$). After 120 epochs, the model does not seem to partition the latent space anymore. The results are display in the appendix C.3.

4.3 Feature matching

CCVAE model learns a latent representation on which we can operate to achieve several tasks such as reconstructing the input image, generating new ones or modifying one label on the input image. Indeed, as all the labels on which we have trained can be modified independently by modifying its latent representation, we can achieve local transformations on the input image. However, as highlighted in the original article, measuring the diversity of the generation, as well as the quality and the locality of the deformation is complex. Such results are most often left to appreciation of the user or using variance on pixels. Having changed some labels for a batch of images, we use a *feature matcher* to evaluate the quality of the generation.

We use the *SuperGlue* model [Sarlin et al. 2020], a Graph Neural Network combined with Optimal Matching layer that computes keypoints correspondences between two images. It first detects keypoints (distinctive points in an image) in both images and extracts descriptors for each of them. Then, a similarity score is computed for each pair of keypoints between images before being filtered (with geometric insights such as spatial distribution). In our experiments, we apply the pretrained model to the "indoor" dataset with the default parameters. Moreover, we do not resize the input as image, as there are already of the same shape. Our experiment is the following:

- Take images that have a local, easy-to-recognize label (namely one in *Smiling, Wearing Glasses, Wearing Necktie*), whether it is positive or negative (smiling or not, ...).
- Generate two images. The reconstruction of the input image, and we generate another image with the same features, but change the studied label to the opposite. To do so, we use the same strategy as in 4.1. We compute the direction between the gaussians that embed the positive and negative labels, and then move away from the gaussian of the label we want to change.
- We use our feature matcher to compute the pairwise correspondences between the original image and the generated images.

The main difficulty of this approach is that we can not impose the keypoints that we want to match. Enabling this would require substantial amount of work on the core algorithm of *SuperGlue*. Instead, we chose labels that are - from a human point-of-view - easy to recognize, and expect that the algorithm finds keypoints on the feature we want to match. The Fig 13 provides some interesting insights about the image generation. When comparing the reconstructed image to the original one, there is a matching between features on labels that are really binary (wearing glasses or not on the figure), but other labels, harder to distinguish (smiling, as on the third row) are not matched correctly. However, we can underline here, seeing the second row of Figure 13, that reconstructed and then modified features are far from matching the original label. The keypoints are on the edges of the face rather than on the details of the face, suggesting that modifying a local feature could impact the entirety of the face, therefore changing the image globally whereas the authors advocated for locality with their variance test.

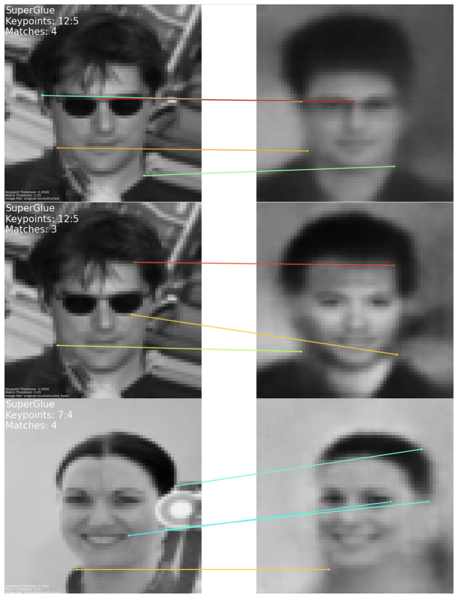

Figure 13: *Correspondences between pairs of image. First row: the reconstructed image correctly matches the glasses of the original image, although not perfectly reconstructed. Second row: Non-matching in the zone of the glasses for a generated image where the label Wearing Glasses has been changed. Third row: Non-matching between the smile of the original image and the reconstructed image without any feature changed. The color of the correspondence represents the confidence of the match, going from 0 (blue, low confidence) to 1 (red, strong confidence).*

To assess this, we compute correspondences for a batch of 50 images following the method described before, we calculate the mean of keypoints matches and of matching confidence between the original and reconstructed image ("**true reconstruction**"). We also compute these means between the original and the image generated with a label changed ("**label reconstruction**"). To us, this is a way to measure the locality of the deformation, as keypoints are computed on all the image (a global measure then). If we have less keypoints and less confidence in average, we can conclude that differences span globally on the image, and are not limited to the local zone of the changed label. We obtained the following results:

- Mean Keypoints Match (True Reconstruction): $M_{\text{keypoints}} = 8.88$
- Confidence (True Reconstruction): $c = 0.76$

In comparison, for the label reconstruction:

- Mean Keypoints Match (Label Reconstruction): $M_{\text{keypoints}} = 7.44$
- Confidence (Label Reconstruction): $c = 0.62$

Hence, that highlights the non-locality of the transformation when modifying one label on a given image. An example of such correspondences is given in Figure 14 Nonetheless, this needs to be mitigated by the low quality of the images, and validated by a thorougher experiment. The code related to this experiment does not appear in our Git link but *SuperGlue* implementation we used comes from this GitHub [4].

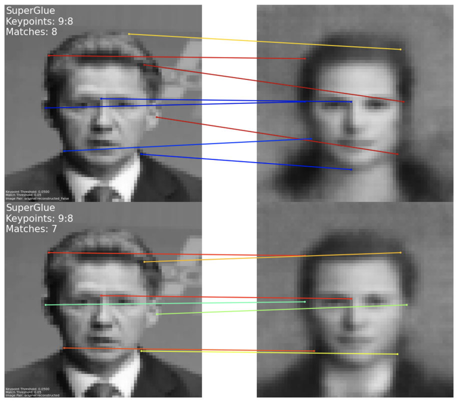

Figure 14: *Example of the impact of changing a feature when reconstructing a image. Although we only changed the Wearing Necktie label, the correspondences are a lot weaker than that of the original reconstructed image.*

5 Conclusions and limitations

Building on previous work on VAE's applied to semi-supervision tasks, the authors designed a novel variational inference framework: CC-VAE. By adapting the utility function and partitioning the latent space, the aim was to disentangle the latent space to make intervention and conditional generation more modular and adaptable to precise characteristics. However, our experiments showed that this was only achieved partially and came at a cost.

- **Label Selection** : We showed that the algorithm only performs well on a small selection of labels : the *CELEBA EASY LABELS*. Even on this subset the disparities are visible. The experiment on the latent space showed that the embeddings are clear for binary labels such as Male or Female but the clustering is less clear for multi-class labels such as hair or eyebrows. If the learned embedding is poor, it is difficult to apply intervention, conditional intervention or latent traversal. This pushed us to implement the algorithm in the multi-class case but the latent embeddings didn't achieve a better clustering performance.

- **Reconstruction** : The partitioning of the latent space came at a high reconstruction cost. Even though this was not the initial aim of this paper we can observe through our multi class experiment or feature matching experiments that the reconstructed images achieve poor performances far from what is demonstrated with diffusion models for example ([Ho et al. 2020]).

[4] `https://github.com/magicleap/SuperGluePretrainedNetwork`

- **Feature Intervention** : Our experiment on feature mapping shows that a local intervention on a local feature applies a global transformation to the image. This shows that the latent space is not fully disentangled between the label latent space and the image reconstruction latent space.

References

HIGGINS, I., MATTHEY, L., PAL, A., BURGESS, C., GLOROT, X., BOTVINICK, M., MOHAMED, S., AND LERCHNER, A. 2017. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*.

HO, J., JAIN, A., AND ABBEEL, P., 2020. Denoising diffusion probabilistic models.

ILSE, M., TOMCZAK, J. M., LOUIZOS, C., AND WELLING, M., 2019. Diva: Domain invariant variational autoencoders.

JOY, T., SCHMON, S. M., TORR, P. H. S., SIDDHARTH, N., AND RAINFORTH, T., 2022. Capturing label characteristics in vaes.

KINGMA, D. P., AND WELLING, M., 2022. Auto-encoding variational bayes.

KINGMA, D. P., REZENDE, D. J., MOHAMED, S., AND WELLING, M., 2014. Semi-supervised learning with deep generative models.

LIU, Z., LUO, P., WANG, X., AND TANG, X. 2015. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

REZENDE, D. J., MOHAMED, S., AND WIERSTRA, D., 2014. Stochastic backpropagation and approximate inference in deep generative models.

SARLIN, P.-E., DETONE, D., MALISIEWICZ, T., AND RABINOVICH, A. 2020. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*.

A Utility function Derivation

This section's goal is to understand the differences in the evidence lower bound functions used in each model and how they are derived. We will start from the Lower Bound derived in the variational inference framework and the M2 model to incrementally build up to the latest models. We will try to explicit the steps that were unclear to us or to highlight some implicit steps in the original papers.

Our paper of interest also proposes a modified version of DIVA with a new loss function.

We start off by restating the definitions :

– The *Kullback-Leibler divergence*:

$$D_{KL}(p(x)|q(x)) = -\sum_x p(x)log(\frac{q(x)}{p(x)})$$

– The *entropy*:

$$H(p(x)) = -\sum p(x)log(x) = \mathbb{E}(-log(p(x))$$

– The *Evidence Lower Bound* (ELBO):

$$ELBO = \mathcal{L}(x) = \mathbb{E}_{q_\phi(z|x)}(log(\frac{p(x,y)}{q_\phi(z|x)}))$$

The ELBO is derived from the loglikelihood as follows, which in the case of semi supervision is given by :

$$log(p_\theta(\tilde{x})) = log\prod_i^n p_\theta(\tilde{x}) = log(\prod_i^n p_\theta(x,y)\prod_i^n p_\theta(x)) \tag{4}$$

A.1 M2 Derivation

Model assumptions:

1. $Z \perp\!\!\!\perp Y$
2. Latent space $\tilde{z} = (z, y)$ is partitioned. So we can rewrite $q_\phi(z, y|x) = q_\phi(z|x)q_\phi(y|x)$

Supervised Case:

– Just take the usual ELBO over (x, y):

$$\begin{aligned}\mathcal{L}_s(x,y) &= \mathbb{E}_{q_\phi(z|x,y)}[\log(\frac{p(x,y,z)}{q(z|x,y)})] &(5)\\ &= \mathbb{E}_{q_\phi(z|x,y)}[\log(\frac{p(x|y,z)p(y,z)}{q(z|x,y)})] &(6)\\ &= \mathbb{E}_{q_\phi(z|x,y)}[\log\frac{p(x|y,z)p(y)p(z)}{q(z|x,y)}] &(7)\end{aligned}$$

Unsupervised Case:

$$
\begin{aligned}
\mathcal{L}_u(x) &= \mathbb{E}_{q_\phi(\tilde{z}|x)}\left[\log \frac{p(x,\tilde{z})}{q_\phi(\tilde{z}|x)}\right] \\
&= \mathbb{E}_{q_\phi(z,y|x)}\left[\log \frac{p(x,z,y)}{q_\phi(z,y|x)}\right] \\
&= \mathbb{E}_{q_\phi(z|x)q_\phi(y|x)}\left[\log \frac{p(x,z,y)}{q_\phi(z|x)q_\phi(y|x)}\right] \\
&= \sum_y q_\phi(y|x)\mathbb{E}_{q_\phi(z|x)}[\log(\frac{p(z|x,y)}{q_\phi(z|x)q_\phi(y|x)})] \\
&= \sum_y q_\phi(y|x)\left[\mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{p(z|x,y)}{q_\phi(z|x)q_\phi(y|x)}\right)\right]\right. \\
&\quad \left. -\ \mathbb{E}_{q_\phi(z|x)}[\log(q_\phi(y|x))]\right] \\
&= \sum_y q_\phi(y|x)\left[\mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{p(z|x,y)}{q_\phi(z|x)q_\phi(y|x)}\right)\right] - \log(q_\phi(y|x))\right] \\
&= \sum_y q_\phi(y|x)(\mathcal{L}(x,y)) - H(q_\phi(z|x))
\end{aligned}
$$

and so the global utlility function is given by

$$
\mathcal{U}(x) = \mathcal{L}_s(x,y) + \mathcal{L}_u(x) + \alpha\mathbb{E}_{p_\theta(x,y)}[-\log q_\phi(y|x)]
$$

Where the last element is added so that the classifier also learns in the supervised case.

A.2 DIVA Derivation

Model assumptions:

1. $Z \perp\!\!\!\perp Y \perp\!\!\!\perp D$

2. Latent space $\tilde{z} = (d, z, y)$ is partitioned. So we can rewrite $q_\phi(z,y,d|x) = q_\phi(z|x)q_\phi(y|x)q_\phi(d|x)$

Supervised case :

$$
\begin{aligned}
\mathcal{L}(d,x,y) &= \mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\right)\right] \\
&= \mathbb{E}_{q_\phi(z_d|x)q_\phi(z_y|x)q_\phi(z_x|x)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z_d|x)q_\phi(z_x|x)q_\phi(z_y|x)}\right)\right] \\
&= \mathbb{E}_{q_\phi(z_d|x)q_\phi(z_y|x)q_\phi(z_x|x)}\left[\log\left(\frac{p_\theta(x|z_d,z_y,z_x)p_\theta(z_d)p_\theta(z_y)p_\theta(z_x)}{q_\phi(z_d|x)q_\phi(z_x|x)q_\phi(z_y|x)}\right)\right] \\
&= \mathbb{E}_{q_\phi(z_d|x)q_\phi(z_y|x)q_\phi(z_x|x)}\left[\log p_\theta(x|z)\right] \\
&\quad + \mathbb{E}_{q_\phi(z_y|x)q_\phi(z_d|x)q_\phi(z_x|x)}\left[\log\frac{p_\theta(z_y)}{q_\phi(z_y|x)}\right] \\
&\quad + \mathbb{E}_{q_\phi(z_y|x)q_\phi(z_d|x)q_\phi(z_x|x)}\left[\log\frac{p_\theta(z_x)}{q_\phi(z_x|x)}\right] \\
&\quad + \mathbb{E}_{q_\phi(z_y|x)q_\phi(z_d|x)q_\phi(z_x|x)}\left[\log\frac{p_\theta(z_d)}{q_\phi(z_d|x)}\right] \\
&= \mathbb{E}_{q_\phi(z_y|x)q_\phi(z_d|x)q_\phi(z_x|x)}\left[\log p_\theta(x|z)\right] \\
&\quad + \mathbb{E}_{q_\phi(z_y|x)}\left[\log\frac{p_\theta(z_y)}{q_\phi(z_y|x)}\right] \\
&\quad + \mathbb{E}_{q_\phi(z_x|x)}\left[\log\frac{p_\theta(z_x)}{q_\phi(z_x|x)}\right] \\
&\quad + \mathbb{E}_{q_\phi(z_d|x)}\left[\log\frac{p_\theta(z_d)}{q_\phi(z_d|x)}\right] \\
&= \mathbb{E}_{q_{\phi_d}(z_d|x)q_{\phi_x}(z_x|x)q_{\phi_y}(z_y|x)}\left[\log p_\theta(x|z)\right] \\
&\quad - \mathrm{KL}\left(q_{\phi_d}(z_d|x)||p_{\theta_d}(z_d|d)\right) \\
&\quad - \mathrm{KL}\left(q_{\phi_x}(z_x|x)||p_\theta(z_x)\right) \\
&\quad - \mathrm{KL}\left(q_{\phi_y}(z_y|x)||p_{\theta_y}(z_y|y)\right)
\end{aligned}
$$

Unsupervised Case :

$$
\begin{aligned}
\mathcal{L}(d,x) = &= \mathbb{E}_{q_\phi(z|x)q_\omega(y|z_y)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z|x)q_\omega(y|z_y)}\right)\right]\\
&= \mathbb{E}_{q_\phi(z_d|x)q_\phi(z_y|x)q_\phi(z_x|x)q_\omega(y|z_y)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z_d|x)q_\phi(z_x|x)q_\phi(z_y|x)q_\omega(y|z_y)}\right)\right]\\
&= \mathbb{E}_{q_\phi(z_d|x)q_\phi(z_y|x)q_\phi(z_x|x)q_\omega(y|z_y)}\left[\log\left(\frac{p_\theta(x|z_d,z_y,z_x)p_\theta(z_d)p_\theta(z_y|y)p_\theta(y)p_\theta(z_x)}{q_\phi(z_d|x)q_\phi(z_x|x)q_\phi(z_y|x)q_\omega(y|z_y)}\right)\right]\\
&= \mathbb{E}_{q_\phi(z_d|x)q_\phi(z_y|x)q_\phi(z_x|x)q_\omega(y|z_y)}\left[\log p_\theta(x|z)\right]\\
&\quad + \mathbb{E}_{q_\phi(z_y|x)q_\phi(z_d|x)q_\phi(z_x|x)q_\omega(y|z_y)}\left[\log\frac{p_\theta(z_y|y)p_\theta(y)}{q_\phi(z_y|x)q_\omega(y|z_y)}\right]\\
&\quad + \mathbb{E}_{q_\phi(z_y|x)q_\phi(z_d|x)q_\phi(z_x|x)q_\omega(y|z_y)}\left[\log\frac{p_\theta(z_x)}{q_\phi(z_x|x)}\right]\\
&\quad + \mathbb{E}_{q_\phi(z_y|x)q_\phi(z_d|x)q_\phi(z_x|x)q_\omega(y|z_y)}\left[\log\frac{p_\theta(z_d)}{q_\phi(z_d|x)}\right]\\
&= \mathbb{E}_{q_\phi(z_y|x)q_\phi(z_d|x)q_\phi(z_x|x)}\left[\log p_\theta(x|z)\right]\\
&\quad + \mathbb{E}_{q_\phi(z_y|x)q_\omega(y|z_y)}\left[\log\frac{p_\theta(z_y|y)p(y)}{q_\phi(z_y|x)q_\omega(y|z_y)}\right]\\
&\quad + \mathbb{E}_{q_\phi(z_x|x)}\left[\log\frac{p_\theta(z_x)}{q_\phi(z_x|x)}\right]\\
&\quad + \mathbb{E}_{q_\phi(z_d|x)}\left[\log\frac{p_\theta(z_d)}{q_\phi(z_d|x)}\right]\\
&= \mathbb{E}_{q_{\phi_d}(z_d|x)q_{\phi_x}(z_x|x)q_{\phi_y}(z_y|x)}\left[\log p_\theta(x|z)\right]\\
&\quad - \mathrm{KL}\left(q_{\phi_d}(z_d|x)||p_{\theta_d}(z_d|d)\right)\\
&\quad - \mathrm{KL}\left(q_{\phi_x}(z_x|x)||p_\theta(z_x)\right)\\
&\quad + \mathbb{E}_{q_\phi(z_y|x)q_\omega(y|z_y)}\left[\log\frac{p_\theta(z_y|y)}{q_\phi(z_y|x)}\right] + \mathbb{E}_{q_\phi(z_y|x)q_\omega(y|z_y)}\left[\log\frac{p(y)}{q_\omega(y|z_y)}\right]
\end{aligned}
$$

Which is the form given in [Ilse et al. 2019].

So the total utility function is given by :

$$
\mathcal{F} = \sum_{i=1}^{N}\mathcal{L}_s(x_i,d_i)+\alpha_d\mathbb{E}_{q_\phi(z_d|x_i)}(log(q_{\omega_d}(d_m|z_d)))+\alpha_y\mathbb{E}_{q_\phi(z_y|x_i)}(log(q_{\omega_y}(y_i|z_y)))+\sum_{j=1}^{N}\mathcal{L}_u(x_j,y_j,d_j)+\alpha_d\mathbb{E}_{q_\phi(z_d|x_j)}(log(q_{\omega_d}(d_j|z_d)))
$$

where additional classifiers are added in the objective function to enforce a better domain and label prediction during training.

A.3 Adapted DIVA

[Joy et al. 2022] proposes to adapt the DIVA framework to a domain agnostic case. He modifies the initial DIVA objective function by removing z_d and the corresponding functions and by partitioning the latent space of the labels. So the final objective function is given by the supervised case :

$$
\begin{aligned}
\mathcal{L}(d,x,y) &= \mathbb{E}_{q_{\phi_d}(z_d|x)q_{\phi_x}(z_x|x)q_{\phi_y}(z_y|x)}\left[\log p_\theta(x|z)\right]\\
&\quad - \mathrm{KL}\left(q_{\phi_x}(z_x|x)||p_\theta(z_x)\right)\\
&\quad - \mathrm{KL}\left(q_{\phi_y}(z_y|x)||p_{\theta_y}(z_y|y)\right)
\end{aligned}
$$

And the unsupervised case given by :

$$
\begin{aligned}
\mathcal{L}(d,x) &= \mathbb{E}_{q_{\phi_d}(z_d|x)q_{\phi_x}(z_x|x)q_{\phi_y}(z_y|x)}\left[\log p_\theta(x|z)\right]\\
&\quad - \mathrm{KL}\left(q_{\phi_x}(z_x|x)||p_\theta(z_x)\right)\\
&\quad + \mathbb{E}_{q_\phi(z_y|x)q_\omega(y|z_y)}\left[\log\frac{p_\theta(z_y|y)}{q_\phi(z_y|x)}\right] + \mathbb{E}_{q_\phi(z_y|x)q_\omega(y|z_y)}\left[\log\frac{p(y)}{q_\omega(y|z_y)}\right]
\end{aligned}
$$

B Training Procedure

We can then add beta parameters as introduced in the Beta-VAE paper [Higgins et al. 2017] to get the required Loss.

C Multi-class CCVAE

C.1 Architecture

Firstly, we modify the dataloader, which treated each attribute as binary. Consequently, adjustments are made to the text files (*.txt* files) to introduce a new label: 'Hair_MULTI'. This newly created attribute serves as a multi-class label encompassing various attributes related to hair color, namely: 'Bald', 'Black_Hair', 'Blond_Hair', 'Brown_Hair', and 'Gray_Hair'. As a result, a new label is introduced, and y denotes observations on this attribute with values in $\{0, 1, 2, 3, 4\}$. We consider only one multi-class label ('Hair_MULTI') during all this experiment.

Secondly, we adapt prior distributions from uniform distributions on $\{0, 1\}$ to uniform distributions over the number of classes per attribute.Then, we design two new neural network blocks: *CondPrior_mc()* and *Classifier_mc()*. The first one enables to compute the parameters of the Gaussian distribution which generates z_i (with i an index where z represents a multi-class label). This module returns different learned values of parameters depending on the input observation (in $\{0, 1, 2, 3, 4\}$ for the 'Hair_MULTI' label). Particular attention has been given to the implementation of this block because several of the methods we tested did not allow for the proper calculation and propagation of gradients. The second one is more straightforward. It calculates the logits of the Categorical distribution (instead of a Bernoulli ditribution for binary attributes) which classifies the latent z for multi-class labels. We just implement an MLP layer from the dimension of z for multi-class labels to the number of classes per labels. We can no longer use the *Diagonal()* module since we no longer have one parameter (Bernoulli parameter) per attribute to compute. The classifier loss is modified as well as other terms in the ELBO for both the unsupervised and supervised lower bound to maximize.

C.2 Results

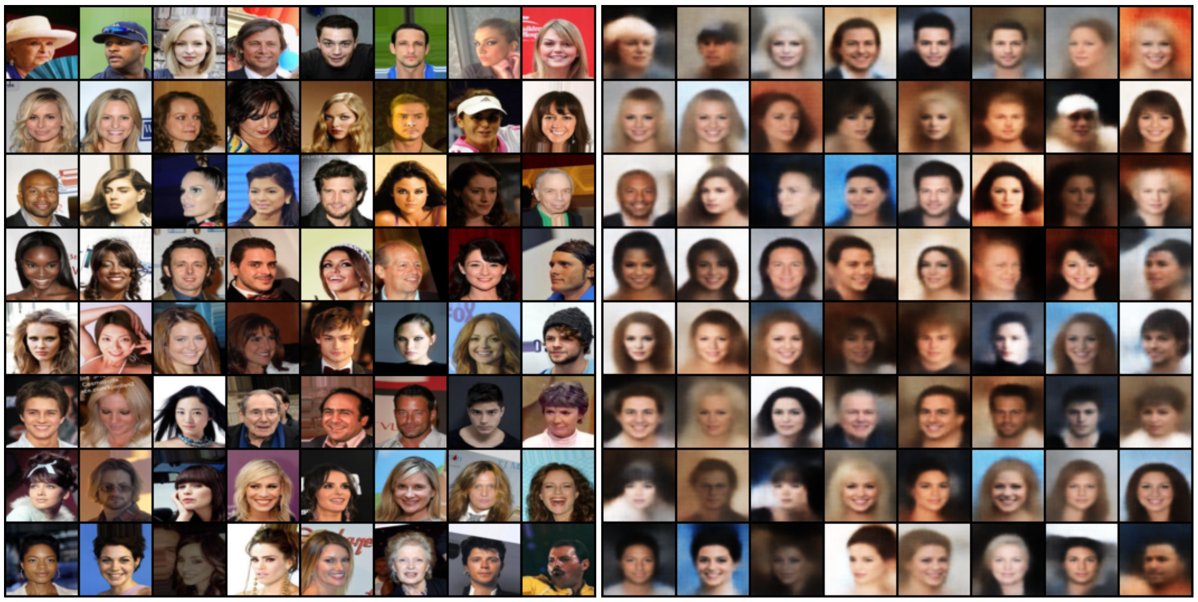

Figure 15: *Reconstructions (right images) of the images on the left with the CCVAE model from the paper trained with binary labels for 80 epochs with a supervision rate of* 10%.

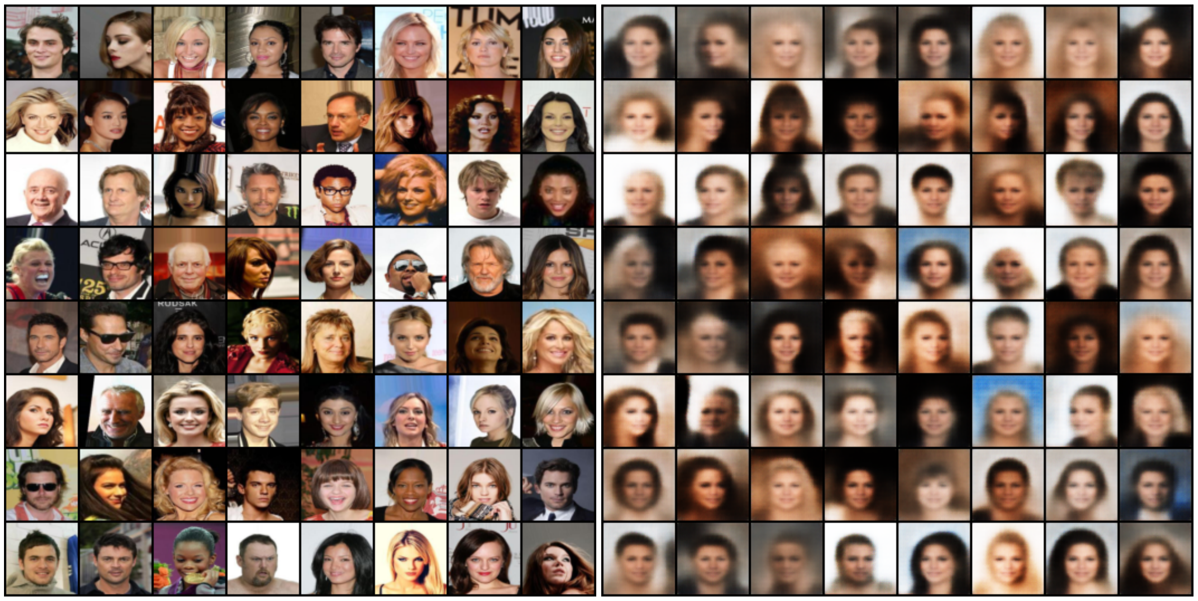

Figure 16: *Reconstructions (right images) of the images on the left with the multi-class CCVAE model trained with 'Hair_MULTI' as the multi-class label for 80 epochs with a supervision rate of* 90%.

C.3 Results with hybrid training

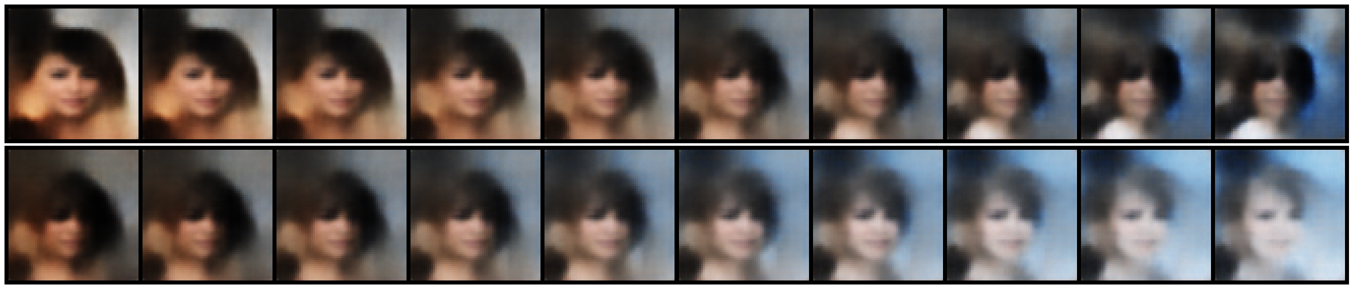

Figure 17: *Latent walk for two binary attributes using the hybrid-trained multi-class friendly CCVAE model. On the left the reconstruction without the attribute and on the right with. On the first row: latent walk across eyeglasses attribute. On the second row: latent walk across pale skin attribute. Hybrid training with the modified version of CCVAE which accepts multi-class label seems to reduce partitioning. Therefore, intervention on a label affects the entire image and especially the hair area (which corresponds to the multi-class label).*