RG-FLOW: A HIERARCHICAL AND EXPLAINABLE FLOW MODEL BASED ON RENORMALIZATION GROUP AND SPARSE PRIOR

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow-based generative models have become an important class of unsupervised learning approaches. In this work, we incorporate the key idea of renormalization group (RG) and sparse prior distribution to design a hierarchical flow-based generative model, called RG-Flow, which can separate different scale information of images with disentangle representations at each scale. We demonstrate our method mainly on the CelebA dataset and show that the disentangled representation at different scales enables semantic manipulation and style mixing of the images. To visualize the latent representation, we introduce the receptive fields for flow-based models and find receptive fields learned by RG-Flow are similar to convolutional neural networks. In addition, we replace the widely adopted Gaussian prior distribution by sparse prior distributions to further enhance the disentanglement of representations. From a theoretical perspective, the proposed method has $O(\log L)$ complexity for image inpainting compared to previous flow-based models with $O(L^2)$ complexity.

1 INTRODUCTION

One of the most important unsupervised learning tasks is to learn the data distribution and build generative models. Over the past few years, various types of generative models have been proposed. Flow-based generative models are a particular family of generative models with tractable distributions (Dinh et al., 2017; Kingma & Dhariwal, 2018; Chen et al., 2018b; 2019; Behrmann et al., 2019; Hoogeboom et al., 2019; Brehmer & Cranmer, 2020; Rezende et al., 2020; Karami et al., 2019). Yet the latent variables are on equal footing and mixed globally. Here, we propose a new flow-based model, *RG-Flow*, which is inspired by the idea of *renormalization group* in statistical physics. RG-Flow imposes locality and hierarchical structure in bijective transformations. It allows us to access different scale information in original images by latent variables at different locations, which offers better explainability. Combined with sparse prior (Olshausen & Field, 1996; 1997; Hyvärinen & Oja, 2000), we show that RG-Flow achieves hierarchical disentangled representations.

Renormalization group (RG) is a powerful tool to analyze statistical mechanics models and quantum field theories in physics (Kadanoff, 1966; Wilson, 1971). It progressively extracts more coarse-scale statistical features of the physical system and decimates irrelevant fine-grained statistics at every scale. Typically, the local transformations used in RG are designed by human physicists and they are not bijective. On the other hand, the flow-based models use cascaded invertible global transformations to progressively turn a complicated data distribution into Gaussian distribution. Here, we would like to combine the key ideas from the RG and the flow-based models. The proposed RG-flow enables machine to learn the optimal RG transformation from data, by constructing local invertible transformations to build a hierarchical generative model for the data distribution. Latent representations are introduced at different hierarchies, which capture the statistical features at the corresponding scales. Together, the latent representation of all hierarchies can be jointly inverted to generate the data. This method was recently proposed in the physics community as NeuralRG (Li & Wang, 2018; Hu et al., 2020).

Our main contributions are two-fold: First, RG-Flow can separate signal statistics of different scales in the input distribution naturally, and represent each scale information in its latent variables $\boldsymbol{z}$. The

hierarchical latent variables live on a hyperbolic tree. Taking CelebA dataset (Liu et al., 2015) as an example, the network will not only find high-level representations, such as gender factor and emotion factor for human faces, but also mid-level and low-level representations. To visualize representations of different scales, we adopt the concept of *receptive field* from convolutional neural networks (CNN) (LeCun, 1988; LeCun et al., 1989) and visualize the hidden units in RG-flow. In addition, since the statistics are separated into a hierarchical fashion, we show that the representations can be mixed at different scales. This achieves an effect similar to style mixing. Second, we introduce the *sparse prior distribution* for the latent variables. We find the sparse prior distribution is helpful to further disentangle representations and make them more interpretable. The widely adopted Gaussian prior is rotationally symmetric. As a result, each of the hidden dimensions in a flow model usually does not have a clear semantic meaning. By using a sparse prior, we demonstrate the clear semantic meaning in the hidden latent space.

2 Related work

Some flow-based generative models also possess multi-scale latent space (Dinh et al., 2017; Kingma & Dhariwal, 2018), and recently hierarchies of features have been utilized in Schirrmeister et al. (2020), where the top-level feature is shown to perform strongly in out-of-distribution (OOD) detection task. Yet, previous models do not impose hard locality constraint in the multi-scale structure. In Appendix F, difference between globally connected multi-scale flow and RG-Flow is discussed, and semantic, meaningful receptive fields do not show up in the globally connected cases. Recently, other more expressive bijective maps have been developed (Hoogeboom et al., 2019; Karami et al., 2019; Durkan et al., 2019), and those methods can be incorporated into the proposed structure to further improve expressiveness of RG-Flow.

Some other classes of generative models rely on a separate inference model to obtain the latent representation. Examples include *variational autoencoders* (Kingma & Welling, 2014), adversarial autoencoders (Makhzani et al., 2015), InfoGAN (Chen et al., 2016), and BiGAN (Donahue et al., 2017; Dumoulin et al., 2017). Those techniques typically do not use hierarchical latent variables, and the inference of latent variables is approximate. Notably, recent advances suggest that having hierarchical latent variables may be beneficial (Vahdat & Kautz, 2020).

Disentangled representations (Tenenbaum & Freeman, 2000; DiCarlo & Cox, 2007; Bengio et al., 2013) during learning is another important aspect in understanding how a model generates images (Higgins et al., 2018). Especially, disentangled high-level representations have been discussed and improved from information theoretical principles (Cheung et al., 2015; Chen et al., 2016; 2018a; Higgins et al., 2017; Kipf et al., 2020; Kim & Mnih, 2018; Locatello et al., 2019; Ramesh et al., 2018). Apart from the high-level representations, the multi-scale structure also lies at the heart of natural images. If a model can separate information of different scales, then the multi-scale representations can be used to perform other tasks, such as style transfer (Gatys et al., 2016; Zhu et al., 2017), face mixing (Karras et al., 2019; Gambardella et al., 2019; Karras et al., 2020), and texture synthesis (Bergmann et al., 2017; Jetchev et al., 2016; Gatys et al., 2015; Johnson et al., 2016; Ulyanov et al., 2016).

Typically, in flow-based generative models, Gaussian distribution is used as the prior for the latent space. Due to the rotational symmetry of Gaussian prior, an arbitrary rotation of the latent space would lead to the same likelihood if the bijective transformation is rich enough. Sparse prior (Olshausen & Field, 1996; 1997; Hyvärinen & Oja, 2000) was proposed as an important tool for unsupervised learning and it leads to better explainability in various domains (Ainsworth et al., 2018; Arora et al., 2018; Zhang et al., 2019). To break the symmetry of Gaussian prior and further improve the interpret-ability, we introduce sparse priors to flow-based models. Please refer to Figure 9 for a quick illustration on the difference between Gaussian prior and a sparse prior, where the sparse prior leads to better disentanglement.

3 Methods

Flow-based generative models. Flow-based generative models are a family of generative models with tractable distributions, which allows efficient sampling and exact evaluation of the probability density (Dinh et al., 2015; 2017; Kingma & Dhariwal, 2018; Chen et al., 2019). The key idea is

to build a bijective mapping $G(\boldsymbol{z}) = \boldsymbol{x}$ between visible variables $\boldsymbol{x}$ and latent variables $\boldsymbol{z}$. Visible variables $\boldsymbol{x}$ are the data that we want to generate, which may follow a complicated probability distribution. And latent variables $\boldsymbol{z}$ usually have simple distribution that can be easily sampled, for example the i.i.d. Gaussian distribution. In this way, the data can be efficiently generated by first sampling $\boldsymbol{z}$ and mapping them to $\boldsymbol{x}$ through $\boldsymbol{x} = G(\boldsymbol{z})$. In addition, we can get the probability associated with each data sample $\boldsymbol{x}$,

$$\log p_X(\boldsymbol{x}) = \log p_Z(\boldsymbol{z}) - \log \left| \frac{\partial G(\boldsymbol{z})}{\partial \boldsymbol{z}} \right|. \tag{1}$$

The bijective mapping $G(\boldsymbol{z}) = \boldsymbol{x}$ is usually composed as a series of bijectors, $G(\boldsymbol{z}) = G_1 \circ G_2 \circ \cdots \circ G_n(\boldsymbol{z})$, such that each bijector layer G_i has a tractable Jacobian determinant and can be inverted efficiently. The two key ingredients in flow-based models are the design of the bijective mapping G and the choice of the prior distribution $p_Z(\boldsymbol{z})$.

Structure of RG-Flow networks. Much of the prior research has focused on designing more powerful bijective blocks for the generator G to improve its expressive power and to achieve better approximations of complicated probability distributions. Here, we focus on designing the architecture that arranges the bijective blocks in a hierarchical structure to separate features of different scales in the data and to disentangle latent representations.

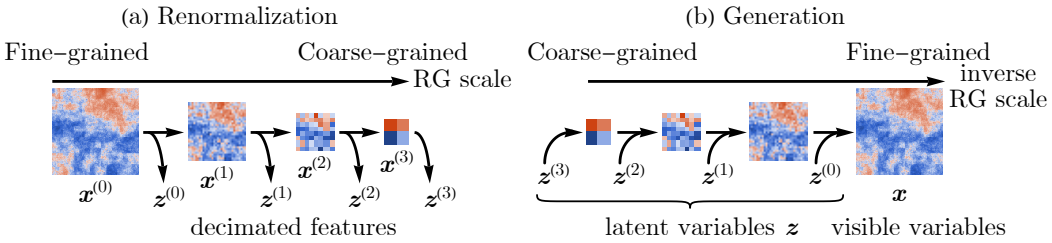

Figure 1: (a) The forward RG transformation splits out decimated features at different scales. (b) The inverse RG transformation generates the fine-grained image from latent variables.

Our design is motivated by the idea of RG in physics, which progressively separates the coarse-grained data statistics from fine-grained statistics by local transformations at different scales. Let $\boldsymbol{x}$ be the visible variables, or the input image (level-0), denoted as $\boldsymbol{x}^{(0)} \equiv \boldsymbol{x}$. One step of the RG transformation extracts the coarse-grained information to send to the next layer $\boldsymbol{x}^{(1)}$ (level-1) and splits out the rest of fine-grained information to variables $\boldsymbol{z}^{(0)}$. The procedure can be described by the following recursive equation (at level-h for example),

$$\boldsymbol{x}^{(h+1)}, \boldsymbol{z}^{(h)} = R_h(\boldsymbol{x}^{(h)}), \tag{2}$$

which is illustrated in Fig. 1(a). $\dim(\boldsymbol{x}^{(h+1)}) + \dim(\boldsymbol{z}^{(h)}) = \dim(\boldsymbol{x}^{(h)})$ and the RG transformation R_h can be made invertible. At each level, the transformation R_h is a local bijective map, which will be constructed by stacking trainable bijective blocks. Its details will be specified soon. The split-out features $\boldsymbol{z}^{(h)}$ can be viewed as latent variables arranged at different scales. Then the inverse RG transformation $G_h \equiv R_h^{-1}$ simply generates the fine-grained image,

$$\boldsymbol{x}^{(h)} = R_h^{-1}(\boldsymbol{x}^{(h+1)}, \boldsymbol{z}^{(h)}) = G_h(\boldsymbol{x}^{(h+1)}, \boldsymbol{z}^{(h)}). \tag{3}$$

The highest-level image $\boldsymbol{x}^{(h_L)} = G_{h_L}(\boldsymbol{z}^{(h_L)})$ can be considered as generated directly from latent variables $\boldsymbol{z}^{(h_L)}$ without referring to any higher-level coarse-grained image, where $h_L = \log_2 L - \log_2 m$ for the original image of size $L \times L$ with local transformations acting on kernel size $m \times m$. Therefore, given the latent variables $\boldsymbol{z} = \{\boldsymbol{z}^{(h)}\}$ at all levels h, the original image can be restored by the following nested maps, as illustrated in Fig. 1(b),

$$\boldsymbol{x} \equiv \boldsymbol{x}^{(0)} = G_0(G_1(G_2(\cdots, \boldsymbol{z}^{(2)}), \boldsymbol{z}^{(1)}), \boldsymbol{z}^{(0)}) \equiv G(\boldsymbol{z}), \tag{4}$$

where $\boldsymbol{z} = \{\boldsymbol{z}^0, \cdots, \boldsymbol{z}^{h_L}\}$. RG-Flow is a flow-based generative model that uses the above composite bijective map G as the generator.

To model the RG transformation, we arrange the bijective blocks in a hierarchical network architecture. Fig. 2(a) shows the side view of the network, where each green or yellow block is a local bijective map. We name the green blocks as the *disentanglers*, which reparametrize local variables to reduce their correlations, and the yellow blocks as the *decimators*, which separate the decimated features out as latent variables. The blue dots on the bottom are the visible variables $\boldsymbol{x}$ from the data, and the red crosses are the latent variables $\boldsymbol{z}$. We omit color channels of the image in the illustration, since we keep the number of color channels unchanged through the transformation.

Fig. 2(b) shows the top-down view of one step of the RG transformation. The green/yellow blocks (disentanglers/decimators) are interwoven on top of each other. The covering area of a disentangler or decimator is defined as the kernel size $m \times m$ of the bijector. For example, in Fig. 2(b), the kernel size is 4×4. After the decimator, three fourth of the degrees of freedom are decimated into latent variables (red crosses in Fig. 2(a)), so the edge length of the image is halved. In terms of formula, for the single-step RG transformation R_h, in each block (p, q) labeled by $p, q = 0, 1, \ldots, \frac{L}{2^h m} - 1$, the mapping from $\boldsymbol{x}^{(h)}$ to $(\boldsymbol{x}^{(h+1)}, \boldsymbol{z}^{(h)})$ is given by

$$\begin{aligned}
&\left\{\boldsymbol{y}^{(h)}_{2^h(mp+\frac{m}{2}+a, mq+\frac{m}{2}+b)}\right\}_{(a,b)\in\square^1_m} = R_h^{\text{dis}}\left(\left\{\boldsymbol{x}^{(h)}_{2^h(mp+\frac{m}{2}+a, mq+\frac{m}{2}+b)}\right\}_{(a,b)\in\square^1_m}\right)\\
&\left\{\boldsymbol{x}^{(h+1)}_{2^h(mp+a, mq+b)}\right\}_{(a,b)\in\square^2_m}, \left\{\boldsymbol{z}^{(h)}_{2^h(mp+a, mq+b)}\right\}_{(a,b)\in\square^1_m/\square^2_m} = R_h^{\text{dec}}\left(\left\{\boldsymbol{y}^{(h)}_{2^h(mp+a, mq+b)}\right\}_{(a,b)\in\square^1_m}\right),
\end{aligned} \tag{5}$$

where $\square^k_m = \{(ka, kb) \mid a, b = 0, 1, \ldots, \frac{m}{k} - 1\}$ denotes the set of pixels in a $m \times m$ square with stride k. The notation $\boldsymbol{x}^{(h)}_{(i,j)}$ stands for the variable (a vector of all channels) at the pixel (i, j) and at the RG level h (similarly for $\boldsymbol{y}$ and $\boldsymbol{z}$ variables). The disentanglers R_h^{dis} and decimators R_h^{dec} can be any bijective neural network. Practically, We use the coupling layer proposed in the Real NVP networks (Dinh et al., 2017) to build them, with a detailed description in Appendix A. By specifying the RG transformation $R_h = R_h^{\text{dec}} \circ R_h^{\text{dis}}$ above, the generator $G_h \equiv R_h^{-1}$ is automatically specified as the inverse transformation.

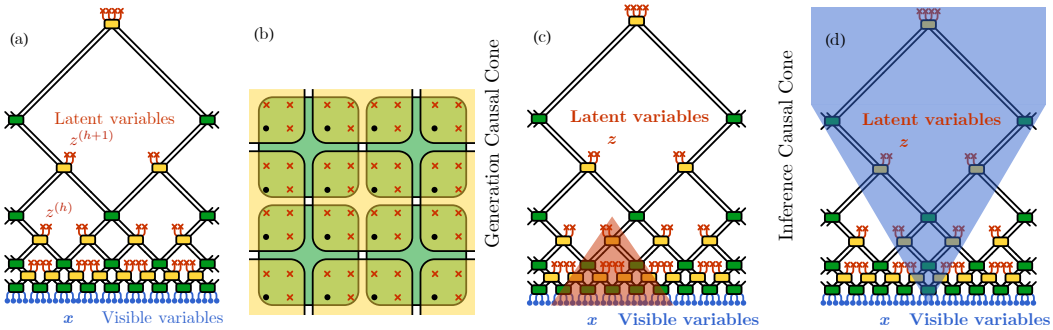

Figure 2: Subplot (a) shows the side view of the network. Green/Yellow blocks denote the disentanglers/decimators, which are bijective maps. Subplot (b) shows the top-down view of the network. The red area in the subplot (c) illustrates the generation causal cone for a latent variable. The blue area in the subplot (d) illustrates the inference causal cone for a visible variable.

Training objectives. After statistics has been decomposed into multiple scales, we would make the latent features to be decoupled. So we assume that the latent variables $\boldsymbol{z}$ are independent random variables, described by a factorized prior distribution

$$p_Z(\boldsymbol{z}) = \prod_l p(z_l), \tag{6}$$

where l labels every element in $\boldsymbol{z}$, including the RG level, the pixel position and the channel. This prior gives the network the incentive to minimize the mutual information between latent variables. This *minimal bulk mutual information* (minBMI) principle was previously proposed to be the information theoretic principle that defines the RG transformation (Li & Wang (2018); Hu et al. (2020)).

Starting from a set of independent latent variables $\boldsymbol{z}$, the generator G should build up correlations locally at different scales, such that the multi-scale correlation structure can emerge in the resulting image $\boldsymbol{x}$ to model the correlated probability distribution of the data. To achieve this goal, we should maximize the log likelihood for $\boldsymbol{x}$ drawn from the data set. The loss function to minimize reads

$$\mathcal{L} = -\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})} \log p_X(\boldsymbol{x}) = -\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})} \left(\log p_Z(R(\boldsymbol{x})) + \log \left| \frac{\partial R(\boldsymbol{x})}{\partial \boldsymbol{x}} \right| \right), \tag{7}$$

where $R(\boldsymbol{x}) \equiv G^{-1}(\boldsymbol{x}) = \boldsymbol{z}$ denotes the RG transformation, which contains trainable parameters. By optimizing the parameters, the network learns the optimal RG transformation from the data.

Receptive fields of latent variables. Due to the nature of local transformations in our hierarchical network, we can define the generation causal cone for one latent variable to be the affected area when that latent variable is changed. This is illustrated as the red cone in Fig. 2(c).

To visualize the latent space representation, we define the *receptive field* for a latent variable z_l as

$$\text{RF}_l = \mathbb{E}_{\boldsymbol{z}\sim p_Z(\boldsymbol{z})} \frac{\partial G(\boldsymbol{z})}{\partial z_l}. \tag{8}$$

The receptive field reflects the response of the generated image to an infinitesimal change of the latent variable z_l, averaged over $p_Z(\boldsymbol{z})$. Therefore, the receptive field of a latent variable is always contained in its generation causal cone. Higher-level latent variables have larger receptive fields than those of the lower-level ones. Especially, if the receptive fields of two latent variables do not overlap, which is often the case for lower-level latent variables, they automatically become disentangled in the representation.

Image inpainting and error correction. Another advantage of the network locality can be demonstrated in the inpainting task. Similar to the generation causal cone, we can define the *inference causal cone* shown as the blue cone in Fig. 2(d). If we perturb a pixel at the bottom of the blue cone, all the latent variables within the blue cone will be affected, whereas the latent variables outside the cone cannot be affected. One important property of the hyperbolic tree-like network is that the higher level contains exponentially fewer latent variables. Even though the inference causal cone is expanding as we go into higher levels, the number of latent variables dilutes exponentially as well, resulting in a constant number of latent variables covered by the inference causal cone on each level. Therefore, if a small local region on an image is corrupted, only $O(\log L)$ latent variables need to be modified, where L is the edge length of the entire image. While for globally connected networks, all $O(L^2)$ latent variables have to be varied.

Sparse prior distribution. We have chosen to hard-code the RG information principle by using a factorized prior distribution, i.e. $p_Z(\boldsymbol{z}) = \prod_l p(z_l)$. The common practice is to choose $p(z_l)$ to be the standard Gaussian distribution, which is spherical symmetric. If we apply any rotation to $\boldsymbol{z}$, the distribution will remain the same. Therefore, one cannot avoid different features from being mixed under the arbitrary rotation.

To overcome this issue, we use an anisotropic sparse prior distribution for $p_Z(\boldsymbol{z})$. In our implementation, we choose the Laplacian distribution $p(z_l) = \frac{1}{2b}\exp(-|z_l|/b)$, which is sparser compared to Gaussian distribution and breaks the spherical symmetry of the latent space. In Appendix C, we show a two-dimensional pinwheel example to illustrate this intuition. This heuristic method will encourage the model to find more semantically meaningful representations by breaking the spherical symmetry.

4 Experiments

Experiment setup. We implement RG-Flow as shown in Fig. 2. In each of the disentangler/decimator (green/yellow block in Fig. 2(a)), we use Real NVP coupling layers. Most of our experiments use the human face dataset CelebA (Liu et al., 2015), and other experiments, such as CIFAR-10 (Krizhevsky et al.), can be found in Appendix B. For CelebA dataset, we crop and scale the images to 32×32. Details of the network and the training procedure can be found in Appendix A.

After training, the network learns to progressively generate finer-grained images, as shown in Fig. 3(a). The colors in the coarse-grained images are not necessarily the same as those at the same

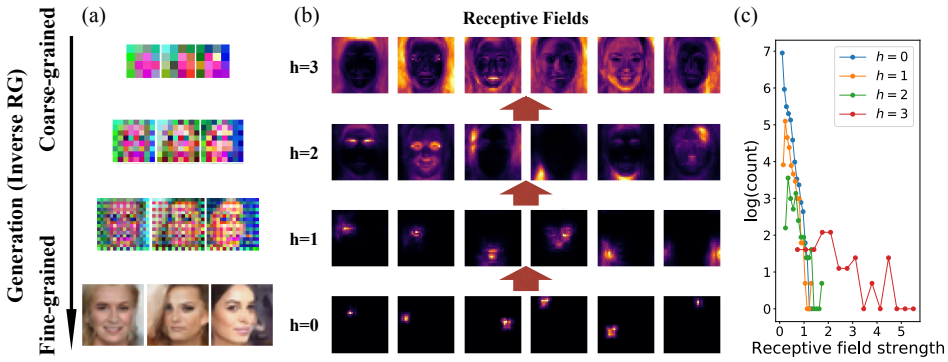

Figure 3: Subplot (a) shows the progressive generation of images during the inverse RG. Subplot (b) shows some receptive fields of latent variables from low level to high level. The strength of each receptive field has been rescaled to unity for better visualization. Subplot (c) shows the statistics of receptive field strength.

positions in the fine-grained images, because there is no constraint to prevent the RG transformation from mixing color channels.

Receptive fields. To visualize the latent space representation, we calculate the receptive field for each latent variable, and list some of them in Fig. 3(b). We can see the receptive size is small for low-level variables and large for high-level ones, as indicated from the generation causal cone. In the lowest level ($h = 0$), the receptive fields are merely small dots. In the second lowest level ($h = 1$), small structures emerge, such as an eyebrow, an eye, a part of hair, etc. In the middle level ($h = 2$), we can see eyebrows, eyes, forehead bang structure emerge. In the highest level ($h = 3$), each receptive field grows to the whole image. We will investigate those explainable latent representations in the next section. For comparison, we show receptive fields of Real NVP in Appendix F. Even though Real NVP has multi-scale structure, which is not locally constrained, semantic representations at different scales do not emerge.

Learned features on different scales. In this section, we show that some of these emergent structures correspond to explainable latent features. Flow-based generative model is the maximal encoding procedure, because the core of flow-based generative model is bijective mapping, and it preserves the dimensionality before and after the encoding. Usually, the images in the dataset live on a low dimensional manifold, and we do not need to use all the dimensions to encode such data. In Fig. 3(c) we show the statistics of the strength of receptive fields. We can see most of the latent variables have receptive fields with relatively small strength, meaning that if we change the value of those latent variables, the generated images will not be affected much. We focus on those latent variables with receptive field strength greater than one, which have visible effects on the generated images. We use h to label the RG level of latent variables, for example, the lowest-level latent variables have $h = 0$, whereas the highest-level latent variables have $h = 4$. In addition, we will focus on $h = 1$ (low level), $h = 2$ (mid level), $h = 3$ (high level) latent variables. There are a few latent variables with $h = 0$ that have visible effects, but their receptive fields are only small dots with no emergent structures.

For high-level latent representations, we found in total 30 latent variables that have visible effects, and six of them have been identified with disentangled and explainable meanings. Those factors are gender, emotion, light angle, azimuth, hair color, and skin color. In Fig. 4(a), we plot the effect of varying those six high-level variables, together with their receptive fields. For the mid-level latent representations, we plot the four leading variables together with their receptive fields in Fig. 4(b), and they control eye, eyebrow, upper right bang, and collar respectively. For the low-level representations, some leading variables control one eyebrow and one eye as shown in Fig. 4(c). We see them achieve better disentangled representations when their receptive fields do not overlap.

Face mixing in scaling direction. Given two images $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$, the conventional image mixing takes a linear combination between $\boldsymbol{z}_A = G^{-1}(\boldsymbol{x}_A)$ and $\boldsymbol{z}_B = G^{-1}(\boldsymbol{x}_B)$ by $\boldsymbol{z} = \lambda \boldsymbol{z}_A + (1-\lambda)\boldsymbol{z}_B$ with $\lambda \in [0, 1]$ and generates the mixed image from $\boldsymbol{x} = G(\boldsymbol{z})$. In our model, latent variables $\boldsymbol{z}$ is

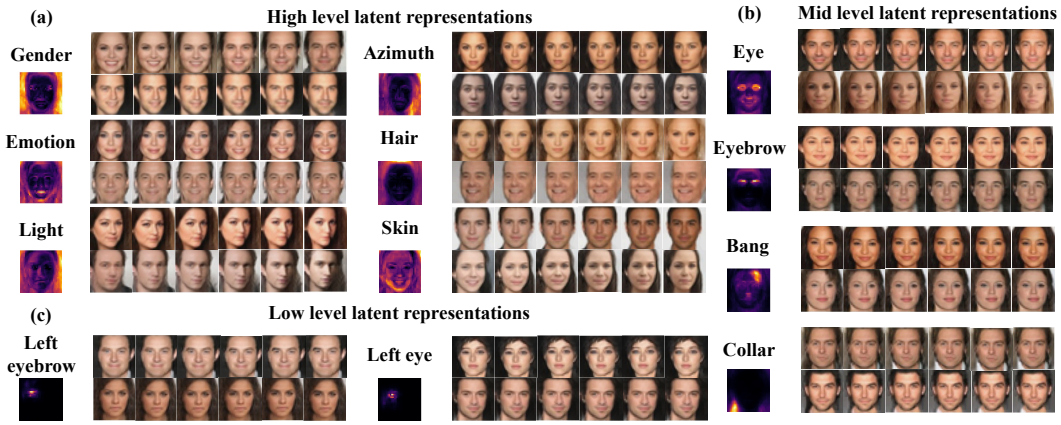

Figure 4: Semantic factors found on different levels. Details can be found in Appendix B

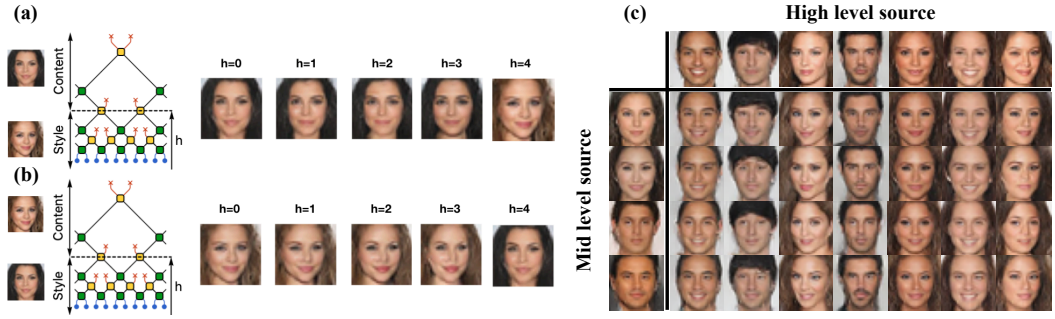

Figure 5: Images mixing in the hyperbolic tree-like latent space.

coordinated by the pixel position (i, j) and the RG level h. The direct access of the latent variable $\boldsymbol{z}^{(h)}_{(i,j)}$ at each point enables us to mix the latent variables in a different manner, which may be dubbed as a "hyperbolic mixing". We consider mixing the large-scale (high-level) features of $\boldsymbol{x}_A$ and the small-scale (low-level) features of $\boldsymbol{x}_B$ by combining their corresponding latent variables via

$$\boldsymbol{z}^{(h)} = \Theta_h(\lambda)\boldsymbol{z}^{(h)}_A + (1 - \Theta_h(\lambda))\, \boldsymbol{z}^{(h)}_B, \tag{9}$$

where $\Theta_h(\lambda)$ is a mask with $\Theta_h(\lambda) = 1$ if $h \geq \lambda$, and $\Theta_h(\lambda) = 0$ otherwise. The parameter λ serves as a dividing line of the scales. As shown in Fig. 5(a), as we tune λ from 0 to 3, more low-level information in the blonde-hair image is mixed with the high-level information of the black-hair image. Especially when $h = 3$, we see the mixed face have similar eyes, nose, eyebrows, and mouth as the blonde-hair image, while the high-level information, such as face orientation and hair color, is taken from the black-hair image. In addition, this mixing is not symmetric under the interchange of $\boldsymbol{z}_A$ and $\boldsymbol{z}_B$, see Fig. 5(b) for comparison. This hyperbolic mixing achieves the similar effect of StyleGAN (Karras et al., 2019; 2020) that we can take mid-level information from one image and mix it with the high-level information of another image. In Fig. 5(c), we show more examples of mixing two human faces.

Image inpainting and error correction. The existence of the inference causal cone ensures that at most $O(\log L)$ latent variables will be affected, if we have a small local corrupted region to be inpainted. In Fig. 6, we show that RG-Flow can faithfully recover the corrupted region (marked as red) only using latent variables locating inside the inference causal cone, which are around one third of all latent variables. For comparison, if we randomly pick the same number of latent variables to modify in Real NVP, it fails to inpaint as shown in Fig. 6 (Constrained Real NVP). To achieve the recovery of similar quality in Real NVP, as shown in Fig. 6 (Real NVP), all latent variables need to be modified, which are of $O(L^2)$ order. See Appendix D for more details about the inpainting task.

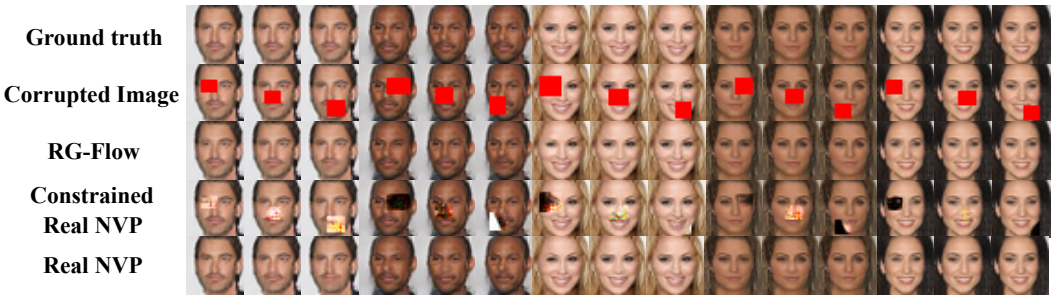

Figure 6: Inpainting experiments for locally corrupted images.

5 Discussion and conclusion

In this paper, we combine the idea of renormalization group and sparse prior distribution to design *RG-Flow*, a probabilistic flow-based generative model. This versatile architecture can be incorporated with any bijective mapping to achieve an expressive flow-based generative model. We have shown that RG-Flow can separate different scale information and encode them in latent variables living on a hyperbolic tree. To visualize the latent encoding in RG-Flow, we defined the receptive field for flow-based models in analogous to that in CNN. Taking CelebA dataset as our main example, we have shown that RG-Flow will not only find high-level representations, but also mid-level and low-level representations. The receptive field serves as a visual guidance for us to find explainable representations. In contrast, the semantic representations of mid-level and low-level structures do not emerge in globally connected multi-scale flow models, such as Real NVP. We also show that the latent representations can be mixed at different scales, which achieves an effect similar to style mixing. In addition, we apply sparse prior distribution for latent variables to further disentangle representations and make them more interpretable.

In our model, if receptive fields of two latent representations do not overlap, they are naturally disentangled. For high-level representations, we propose to utilize sparse prior to encourage disentanglement. We find that if the dataset only contains a few high-level factors, such as the 3D chair dataset (Aubry et al., 2014), it is hard to find explainable high-level disentangled representations. The redundant nature of the encoding from flow-based models also makes it harder to form disentangled representation if the number of high-level factors is small. Incorporating information theoretic criteria to disentangle high-level representations in the redundant coding scheme will be an interesting future direction.

References

Samuel K Ainsworth, Nicholas J Foti, Adrian KC Lee, and Emily B Fox. oi-vae: Output interpretable vaes for nonlinear group factor analysis. In *International Conference on Machine Learning*, pp. 119–128, 2018.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495, 2018.

Mathieu Aubry, Daniel Maturana, Alexei A. Efros, Bryan C. Russell, and Josef Sivic. Seeing 3d chairs: Exemplar part-based 2d-3d alignment using a large dataset of CAD models. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014*, pp. 3762–3769. IEEE Computer Society, 2014.

Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Jörn-Henrik Jacobsen. Invertible residual networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pp. 573–582. PMLR, 2019.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Urs Bergmann, Nikolay Jetchev, and Roland Vollgraf. Learning texture manifolds with the periodic spatial GAN. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 469–477. PMLR, 2017.

Johann Brehmer and Kyle Cranmer. Flows for simultaneous manifold learning and density estimation. *CoRR*, abs/2003.13913, 2020.

Tian Qi Chen, Xuechen Li, Roger B. Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018,*, pp. 2615–2625, 2018a.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018*, pp. 6572–6583, 2018b.

Tian Qi Chen, Jens Behrmann, David Duvenaud, and Jörn-Henrik Jacobsen. Residual flows for invertible generative modeling. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019*, pp. 9913–9923, 2019.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016*, pp. 2172–2180, 2016.

Brian Cheung, Jesse A. Livezey, Arjun K. Bansal, and Bruno A. Olshausen. Discovering hidden factors of variation in deep networks. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

James J DiCarlo and David D Cox. Untangling invariant object recognition. *Trends in cognitive sciences*, 11(8):333–341, 2007.

Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: non-linear independent components estimation. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martín Arjovsky, Olivier Mastropietro, and Aaron C. Courville. Adversarially learned inference. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019*, pp. 7509–7520, 2019.

Andrew Gambardella, Atilim Günes Baydin, and Philip H. S. Torr. Transflow learning: Repurposing flow models without retraining. *CoRR*, abs/1911.13270, 2019.

L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2414–2423, 2016.

Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015*, pp. 262–270, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 770–778. IEEE Computer Society, 2016.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

Irina Higgins, David Amos, David Pfau, Sébastien Racanière, Loïc Matthey, Danilo J. Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *CoRR*, abs/1812.02230, 2018.

Emiel Hoogeboom, Rianne van den Berg, and Max Welling. Emerging convolutions for generative normalizing flows. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2771–2780. PMLR, 2019.

Hong-Ye Hu, Shuo-Hui Li, Lei Wang, and Yi-Zhuang You. Machine learning holographic mapping by neural network renormalization group. *Phys. Rev. Research*, 2:023369, Jun 2020.

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

Nikolay Jetchev, Urs Bergmann, and Roland Vollgraf. Texture synthesis with spatial generative adversarial networks. *CoRR*, abs/1611.08207, 2016.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision - ECCV 2016 - 14th European Conference, Proceedings, Part II*, volume 9906 of *Lecture Notes in Computer Science*, pp. 694–711. Springer, 2016.

Leo P. Kadanoff. Scaling laws for ising models near T_c. *Physics Physique Fizika*, 2:263–272, Jun 1966.

Mahdi Karami, Dale Schuurmans, Jascha Sohl-Dickstein, Laurent Dinh, and Daniel Duckworth. Invertible convolutional flow. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019*, pp. 5636–5646, 2019.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pp. 4401–4410. Computer Vision Foundation / IEEE, 2019.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020*, pp. 8107–8116. IEEE, 2020.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2654–2663. PMLR, 2018.

Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018*, pp. 10236–10245, 2018.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

Thomas N. Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. In *8th International Conference on Learning Representations, ICLR 2020*. OpenReview.net, 2020.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL http://www.cs.toronto.edu/~kriz/cifar.html.

Y. LeCun. A theoretical framework for back-propagation. 1988.

Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems 2*, pp. 396–404. Morgan Kaufmann, 1989.

Shuo-Hui Li and Lei Wang. Neural network renormalization group. *Phys. Rev. Lett.*, 121:260601, Dec 2018.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pp. 3730–3738. IEEE Computer Society, 2015. doi: 10.1109/ICCV.2015.425. URL https://doi.org/10.1109/ICCV.2015.425.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4114–4124. PMLR, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*. OpenReview.net, 2019.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian J. Goodfellow. Adversarial autoencoders. *CoRR*, abs/1511.05644, 2015.

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

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pp. 8024–8035, 2019.

Aditya Ramesh, Youngduck Choi, and Yann LeCun. A spectral regularizer for unsupervised disentanglement. *CoRR*, abs/1812.01161, 2018.

Danilo Jimenez Rezende, George Papamakarios, Sébastien Racanière, Michael S. Albergo, Gurtej Kanwar, Phiala E. Shanahan, and Kyle Cranmer. Normalizing flows on tori and spheres. *CoRR*, abs/2002.02428, 2020.

Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016*, pp. 901, 2016.

Robin Tibor Schirrmeister, Yuxuan Zhou, Tonio Ball, and Dan Zhang. Understanding anomaly detection with deep invertible networks through hierarchies of distributions and features. *CoRR*, abs/2006.10848, 2020.

Joshua B. Tenenbaum and William T. Freeman. Separating style and content with bilinear models. *Neural Comput.*, 12(6):1247–1283, 2000.

Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1349–1357. JMLR.org, 2016.

Arash Vahdat and Jan Kautz. NVAE: A deep hierarchical variational autoencoder. *CoRR*, abs/2007.03898, 2020.

Kenneth G. Wilson. Renormalization group and critical phenomena. i. renormalization group and the kadanoff scaling picture. *Phys. Rev. B*, 4:3174–3183, Nov 1971.

Juexiao Zhang, Yubei Chen, Brian Cheung, and Bruno A. Olshausen. Word embedding visualization via dictionary learning. *CoRR*, abs/1910.03833, 2019.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.