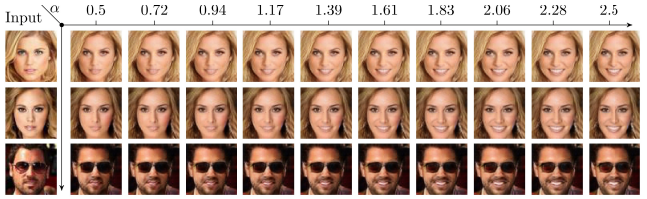
Reverse Variational Autoencoder for Visual Attribute Manipulation and Anomaly Detection

Lydia Gauerhof*
Corporate Research, Robert Bosch GmbH
lydia.gauerhof@de.bosch.com

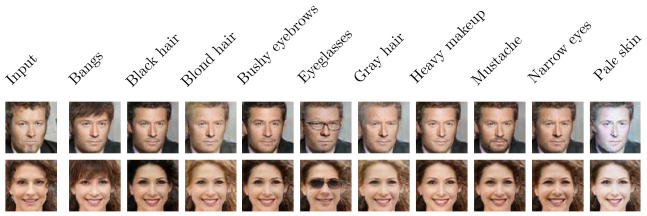
Nianlong Gu*
Institute of Neuroinformatics, ETH Zurich
niangu@ethz.ch

Abstract

In this paper, we introduce the 'Reverse Variational Autoencoder" (Reverse-VAE) which is a generative network. On the one hand, visual attributes can be manipulated and combined while generating images. On the other hand, anomalies, meaning deviations from the data space used for training, can be detected. During training the generator network maps samples from stochastic latent vectors to the data space. Meanwhile the encoder network takes these generated images to reconstruct the latent vector. The generator and discriminator are trained adversarially. The discriminator is trained to distinguish between real and generated data. Overall, our model tries to match the joint latent/data-space distribution of the generator and the latent/data-space joint distribution of the encoder by minimizing their Kullback-Leibler divergence. Desired visual attributes of CelebA images are successfully manipulated. The performance of anomaly detection is competitive with state-of-the-art on MNIST.



(a) The first column is original serious faces. From second to last column, the dominance of a smile increases with the scale factor α changing from 0.5 to 2.5 linearly.



(b) Original images in first column are manipulated.

Figure 1: Adding single visual attributes

1. Introduction

Highly automated driving (HAD) has the potential to revolutionize the way we travel. At the same time, HAD is a safety-critical application in which the violation of safety goals, e.g. a crash with other road users is not acceptable. Consequently, when used for HAD, DL models such as Deep Neural Networks (DNNs) are required to perform robustly [9, 15, 2, 14, 11] despite all kinds of anomalies. Causes for anomalies include lack of diversity in training data set, or changes in sensors over time which may result in shift of distribution of captured images with respect to training data set [16, 29]. Therefore, it is important to detect anomalies - whether the current input image of a DNN is beyond the feature distribution of the training images data set [32, 5]. Then these anomalies can be included to the data set, e.g. by data augmentation based on attribute manipulation, in order to increase robustness.

Images can be regarded as high dimensional vectors which is challenging to analyze the distribution directly. Luckily, the fact that images usually have patterns like human faces indicates that the distribution of a set of images may lie in a low dimensional manifold. This has been experimentally proven by the success of generative adversarial models [12, 4, 18] which can generate various and realistic images. However, a GAN [12] type structure can only learn the mapping from low dimensional latent space to the images space. It is still challenging to both get the embedding of an image and generate new images in a decent way.

Variational Autoencoder (VAE) is one of the earliest model which aims to do both image encoding and image generation. Although VAE can learn meaningful image embedding that can be used for data distribution analysis, it tends to generate images with blurring effects which limits its usage in image generation and manipulation tasks. Inspired by VAE, more advanced models like VAE-GAN

*equal contribution

[22] and ALI [8] were proposed with the goal of improving the image generation performance while keeping the ability of encoding input images to latent space. These models involve in either picking up a certain hidden layer of the discriminator as feature-wise representation, or adopting a sophisticated model structure. Moreover, in these works the experiments are mainly done on 64×64. It remains unclear if those models can be well scaled to deal with images with larger resolution in a more practical scenario.

In this work, we introduce the "Reverse Variational Autoencoder" (Reverse-VAE) which can not only learn an accurate mapping to low dimensional space, but also generate realistic and diverse images. Moreover, our model is compatible with the recently proposed progressively growing strategy [18] to process high resolution images with good scalability. Our model can be reconfigured such that it is used either for anomaly detection or for visual attribute manipulation as a data augmentation method to improve the DNN model robustness against anomalies [1, 10, 28, 34, 33] (examples are found in fig. 1). Our contributions are:

- **A novel form of training settings** reduces the gap between joint latent/data distribution of generator and the joint distribution of encoder by minimizing the Kullback-Leibler divergence. Image **generation / reconstruction** are competitive with state-of-the-art.
- A **simple architecture** makes our model **easy to train with less parameter tuning** and able to be **up-scaled to generate and reconstruct high resolution images** using a PGGAN [18] setting.
- Good reconstruction performance is **restricted on distribution of training data** enables the model to perform well in detecting anomalies.
- For manipulating visual attributes the model is trained **without auxiliary information**, such as labeled attributes. After training we extract **dedicated visual attribute vectors in the latent space** using a small subset of labeled images. We gain flexibility in manipulating new attributes without retraining the model.
- **Combining both applications** leads to a reduced training effort and to an increased development efficiency.

Although in this paper the attribute manipulation and detected anomalies do not necessarily rely on each other, this approach strengthens the development of a unified model for detecting an anomalies and extracting the according attributes in order to augment data.

2. Related Work

First, deep generative models, such as Generative Adversarial Network (GAN) [12] and Variational Autoencoder (VAE) [20], modeling high dimensional data sets are explained. Second, models combining aspects of VAE and GAN are introduced and difference to our model are discussed.

2.1. GAN, VAE and their extensions

The GAN [12] generates more realistic images by making use of an adversarial training procedure. A discriminator learns to distinguish the real images from the images synthesized by a generator. At the same time, the generator tries to "fool" the discriminator by generating more realistic images. Wasserstein-GAN (WGAN) solves the gradient vanishing and mode collapse problem of the original GAN [4] with a minmax game of the Wasserstein distance. Moreover, Chen *et al*. [6] proposed an information-theoretic extension to the GAN (InfoGAN) which is able to learn disentangled representation of limited visual attributes, such as the rotation or stroke of MNIST [23] digits. Nevertheless, the GAN-type models cannot learn a low dimensional embedding as we need for feature distribution analysis.

The VAE predicts the posterior distribution over the latent variables by employing an encoder, and uses an decoder to reconstruct the images given the encoder output [20]. These generated images usually look blurred, though. The Conditional Variational Autoencoder (CVAE) and its variants are proposed for structured output prediction based on the conditional deep generative model with known label information [35]. CVAE is not suitable for our purpose, since we want a model to disentangle information without given auxiliaries during training.

2.2. Models combining aspects of VAE and GAN

After VAE and GAN were proposed, models combining different aspects of VAE and GAN have evolved: for example Adversarial Autoencoder (AAE), VAE-GAN and Adversarially Learned Inference (ALI).

The AAE is a probabilistic autoencoder including a GAN to conduct variational inference by meeting the aggregated posterior of the latent vector with an arbitrary prior distribution [26]. Compared with the original images, the generated images still look blurred.

Apart from AAE, there is also VAE-GAN combining VAE with GAN such that the learned feature representations in the GAN's discriminator are used as a basis for the VAE reconstruction loss [22]. In VAE-GAN the feature-wise reconstruction loss is define as

$$L_{\text{recon\_x}} = \|\text{Dis}_l(\boldsymbol{x}) - \text{Dis}_l(\hat{\boldsymbol{x}})\|_2^2 \tag{1}$$

where $\text{Dis}_l(\boldsymbol{x})$ means the l^{th} hidden layer of the discriminator, and $\boldsymbol{x}$, $\hat{\boldsymbol{x}}$ are input images and reconstructed images respectively. Our model differentiates from VAE-GAN in a way that we did not use such a feature-wise reconstruction

loss primarily, with less parameter tuning such as the selection of l over different data set or different model size. Experiments show that our model tends to balance the tasks of generating high quality images and accurately reconstructing the input images more properly.

Furthermore, the model Adversarially Learned Inference (ALI) was proposed to learn a generation network (generator) and an inference network (encoder) using an adversarial framework [8]. A discriminator is trained to distinguish between joint samples $(\tilde{\boldsymbol{z}}, \boldsymbol{x})$ of the data and the corresponding latent vector from the encoder and the joint samples $(\boldsymbol{z}, \tilde{\boldsymbol{x}})$ from the generator. At the same time the encoder and generator are trained jointly to fool the discriminator. Assuming the discriminator is optimal, the encoder and generator are trained to minimize the Jensen-Shannon divergence [24] between $p(\tilde{\boldsymbol{z}}, \boldsymbol{x})$ and $p(\boldsymbol{z}, \tilde{\boldsymbol{x}})$.

Compared with ALI, our approach also aims to reduce the gap between the joint distribution $p(\boldsymbol{z}, \tilde{\boldsymbol{x}})$ and $p(\tilde{\boldsymbol{z}}, \boldsymbol{x})$. The difference is that we achieve this goal by minimizing the KL-divergence [21] between $p(\boldsymbol{z}, \tilde{\boldsymbol{x}})$ and $p(\tilde{\boldsymbol{z}}, \boldsymbol{x})$ (note that the KL-divergence is not symmetric, so the order does matter). By choosing such a loss function, the discriminator only needs to distinguish between real images and generated images, while in ALI the discriminator is more complicated. We argue that a simpler discriminator structure is advantageous since in ALI the way of combining a pair of a latent vector and an image by concatenation to express the "joint" relationship may influence the stability of training a GAN. Moreover, a compact structure enables our model to be up-scaled to generate and reconstruct high resolution images. For example, we can progressively increase the resolution using the method introduced in PGGAN [18]. On the contrary, it remains unclear how to apply the progressive growing scheme in ALI where the latent vectors and images are concatenated before being fed into the discriminator.

3. Approach

Figure 2 shows the Reverse-VAE network structure. The generator takes the latent vector $\boldsymbol{z}$ whose elements follow Gaussian distribution $\boldsymbol{z} \sim \mathcal{N}(0, I)$, and generates image $\tilde{\boldsymbol{x}}$. Receiving the generated image $\tilde{\boldsymbol{x}}$, the encoder aims to reconstruct the input latent vector of the generator $\hat{\boldsymbol{z}}$. The discriminator learns to distinguish between the generated image $\tilde{\boldsymbol{x}}$ and the real image $\boldsymbol{x}$. Similar to WGAN [4], the output of the discriminator, $\mathrm{Dis}(\boldsymbol{x})$, is used to calculate the Wasserstein distance, which is also called Earth Mover's Distance [30], .

3.1. Mathematical approach

Let θ denote the parameters for the generator, and ϕ denote the parameters for the encoder. Joint distribution of the latent vector and the image for the generator is expressed by

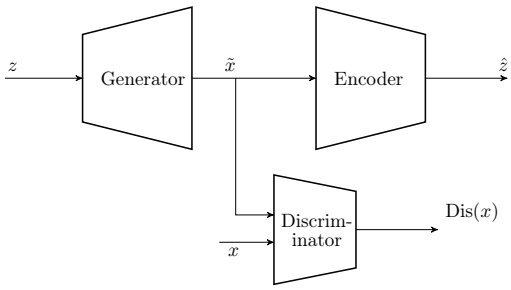


Figure 2: The network structure of the Reverse-VAE model

$p_\theta(\boldsymbol{z}, \boldsymbol{x})$, and joint distribution of the latent vector and the image for the encoder is expressed by $q_\phi(\boldsymbol{z}, \boldsymbol{x})$.

Although KL-Divergence $D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}, \boldsymbol{x}) \| p_\theta(\boldsymbol{z}, \boldsymbol{x}))$ [36] is not mathematically equal to $D_{\mathrm{KL}}(p_\theta(\boldsymbol{z}, \boldsymbol{x}) \| q_\phi(\boldsymbol{z}, \boldsymbol{x}))$, minimizing $D_{\mathrm{KL}}(p_\theta(\boldsymbol{z}, \boldsymbol{x}) \| q_\phi(\boldsymbol{z}, \boldsymbol{x}))$ is leading to the same goal of matching joint distributions $q_\phi(\boldsymbol{z}, \boldsymbol{x})$ and $p_\theta(\boldsymbol{z}, \boldsymbol{x})$.

The training goal is chosen to minimize the KL divergence between joint distribution $p_\theta(\boldsymbol{z}, x)$ and $q_\phi(\boldsymbol{z}, x)$:

$$
\begin{aligned}
&D_{\mathrm{KL}}(p_\theta(\boldsymbol{z}, \boldsymbol{x}) \| q_\phi(\boldsymbol{z}, \boldsymbol{x})) = \mathbb{E}_{(\boldsymbol{z}, \boldsymbol{x}) \sim p_\theta(\boldsymbol{z}, \boldsymbol{x})} \left[\log \frac{p_\theta(\boldsymbol{z}, \boldsymbol{x})}{q_\phi(\boldsymbol{z}, \boldsymbol{x})} \right] \\
&= \mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} \left[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_\theta(\boldsymbol{z})}{q_\phi(\boldsymbol{x}) q_\phi(\boldsymbol{z}|\boldsymbol{x})} \right] \right\} \\
&= \mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} \left[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z})}{q_\phi(\boldsymbol{x})} \right] \right. \\
&\quad \left. + \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} \left[-\log q_\phi(\boldsymbol{z}|\boldsymbol{x}) \right] + \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} \left[\log p_\theta(\boldsymbol{z}) \right] \right\} \\
&= \mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ D_{\mathrm{KL}}(p_\theta(\boldsymbol{x}|\boldsymbol{z}) \| q_\phi(\boldsymbol{x})) \right. \\
&\quad \left. + \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} [-\log q_\phi(\boldsymbol{z}|\boldsymbol{x})] + \log p_\theta(\boldsymbol{z}) \right\}
\end{aligned}
\tag{2}
$$

Since the prior distribution of $\boldsymbol{z}$ is fixed during the training process, $\mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ \log p_\theta(\boldsymbol{z}) \right\}$ is a constant, it has no contribution to computing the gradient and is neglected here. Therefore, the loss function of the Reverse-VAE model is:

$$
\begin{aligned}
L_{\text{Reverse-VAE}} = &\mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ D_{\mathrm{KL}}(p_\theta(\boldsymbol{x}|\boldsymbol{z}) \| q_\phi(\boldsymbol{x})) \right\} \\
&+ \mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} [-\log q_\phi(\boldsymbol{z}|\boldsymbol{x})] \right\}
\end{aligned}
\tag{3}
$$

The loss function of the Reverse-VAE contains two terms. The first term is the KL divergence between the generator output distribution $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ and the prior distribution $q_\phi(\boldsymbol{x})$ representing the real image data. Similar to the AAE [26], a discriminator is applied to distinguish between generated image (generator output) and the real image. The generator and the discriminator are trained adversarially to minimize the first term $\mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})}[D_{\mathrm{KL}}(p_\theta(\boldsymbol{x}|\boldsymbol{z}) \| q_\phi(\boldsymbol{x}))]$ of eq. (3).

The second term of the loss function in eq. (3) is the reconstruction error. Suppose that $\boldsymbol{z}$ is the input latent vector of the generator, and the encoder output, $\hat{\boldsymbol{z}}$, is the reconstruction of the latent vector. In our model each element of the input vector of the generator $\boldsymbol{z}$ follows independent normal distribution $\mathcal{N}(0,1)$. According to Kingma and Welling [20], we assume each element of the reconstruction of the latent vector $\hat{\boldsymbol{z}}$ also follows independent Gaussian distribution with fixed variance. In this case, the reconstruction error can be transformed to the sum of square error [20], where $c = 1$ is a constant related with the variance of the reconstructed latent vector:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z})}\{\mathbb{E}_{\boldsymbol{x}\sim p_\theta(\boldsymbol{x}|\boldsymbol{z})}[-\log q_\phi(\boldsymbol{z}|\boldsymbol{x})]\} \\ \sim \mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z})}\left[c\left\|\boldsymbol{z}-\hat{\boldsymbol{z}}\right\|_2^2\right]\end{aligned} \tag{4}$$

3.2. Training

The training setting for the generator and discriminator is similar to the training setting of WGAN-GP [13]. The generator loss function is:

$$L_{\text{Gen}} = -\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[\text{Dis}(\text{Gen}(\boldsymbol{z}))\right] \tag{5}$$

The discriminator loss function is:

$$\begin{aligned}L_{\text{Dis}} = &\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[\text{Dis}(\text{Gen}(\boldsymbol{z}))\right] - \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[\text{Dis}(\boldsymbol{x})\right] \\ &+ \lambda\,\mathbb{E}_{\boldsymbol{x}_{\text{int}}\sim p_{\boldsymbol{x}_{\text{int}}}(\boldsymbol{x}_{\text{int}})}\left[(\|\nabla_{\boldsymbol{x}_{\text{int}}}\text{Dis}(\boldsymbol{x}_{\text{int}})\|_2 - 1)^2\right]\end{aligned} \tag{6}$$

The hyper parameter λ is set to $\lambda = 10$ [13]. The first part of the discriminator loss function in eq. (6) is related with the negative Wasserstein distance, similar to WGAN [4] and WGAN-GP[13]. The second part includes the gradient penalty term that enforces the Lipschitz constraint [13]. Computing the gradient penalty requires to get random samples from the space between real data distribution and generated data distribution. To approximate this operation, data $\boldsymbol{x}_{\text{int}}$ is uniformly sampled along the straight lines between the pairs of real data $\boldsymbol{x}$ and generated data $\tilde{\boldsymbol{x}}$. This is described in eq. (7) where ϵ is random variable following uniform distribution.

$$\boldsymbol{x}_{\text{int}} = \epsilon\boldsymbol{x} + (1-\epsilon)\tilde{\boldsymbol{x}}, \quad \epsilon\sim U[0,1] \tag{7}$$

During training, the encoder learns to reconstructs the generator input $\boldsymbol{z}\sim p_\theta(\boldsymbol{z})$ given the generated image $\text{Gen}(\boldsymbol{z})$. Let $\hat{\boldsymbol{z}} = \text{Enc}(\text{Gen}(\boldsymbol{z}))$ represent the reconstructed latent vector. According to eq. (4), the latent vector reconstruction loss function is:

$$L_{\text{recon\_z}} = \mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z})}\left[\left\|\boldsymbol{z}-\hat{\boldsymbol{z}}\right\|_2^2\right] \tag{8}$$

$L_{\text{recon\_z}}$ in eq. (8) is optimized for the encoder's parameters with a learning rate $\alpha = 10^{-4}$ and for the generator's parameters with a learning rate $\frac{\alpha}{5}$. We choose a lower learning rate for generator for optimizing $L_{\text{recon\_z}}$ since we need to ensure good quality of generated images, which is optimized via minimizing L_{Gen}.

Since in our model each element of input latent vector $\boldsymbol{z}$ follows independent normal distribution $\mathcal{N}(0,1)$, the L2-norm loss $L_{\text{recon\_z}}$ only ensures that encoder's output has a Gaussian distribution with zero mean. In order to make the variance of elements of the encoder output to be 1, besides $L_{\text{recon\_z}}$ we add an extra loss for the encoder: $|\sigma(\{\hat{\boldsymbol{z}}_d^{(i)}\})-1|$, where $\sigma(\{\hat{\boldsymbol{z}}_d^{(i)}\})$ is the standard deviation of the elements of the encoder's outputs across all dimensions over one mini batch. The overall training procedure is shown in Alg. 1. Adam optimizer [19] is used. Code is available at github.com/nianlonggu/reverse_variational_autoencoder .

Algorithm 1 Training the Reverse-VAE model. $\lambda = 10$, $m = 100$, $n_{\text{dis}} = 5$, $\alpha = 0.0001$,$\beta_1 = 0$,$\beta_2 = 0.99$, $\xi = 0.01$, $\eta = 1$

Require: The gradient penalty coefficient λ, the number of discriminator iterations per generator iteration n_{dis}, the batch size m, Adam hyperparameters α, β_1, β_2, θ is a general notation for model parameters.

while not converged **do**
  for $l = 1, ..., n_{\text{dis}}$ **do**
    Sample a batch of real data $\{\boldsymbol{x}^{(i)}\}_{i=1}^m \sim p_{\text{x}}(\boldsymbol{x})$
    a batch of latent variables $\{\boldsymbol{z}^{(i)}\}_{i=1}^m \sim p_{\boldsymbol{z}}(\boldsymbol{z})$,
    a batch of random variables $\{\epsilon^{(i)}\}_{i=1}^m \sim U[0,1]$.
    $\tilde{\boldsymbol{x}}^{(i)} \leftarrow \text{Gen}(\boldsymbol{z}^{(i)})$
    $\boldsymbol{x}_{\text{int}}^{(i)} \leftarrow \epsilon\boldsymbol{x}^{(i)} + (1-\epsilon)\tilde{\boldsymbol{x}}^{(i)}$
    $L_{\text{Dis}}^{(i)} \leftarrow \text{Dis}(\tilde{\boldsymbol{x}}^{(i)}) - \text{Dis}(\boldsymbol{x}^{(i)})$
      $+\lambda(\left\|\nabla_{\boldsymbol{x}_{\text{int}}}\text{Dis}(\boldsymbol{x}_{\text{int}}^{(i)})\right\|_2 - 1)^2$
    $\theta_{\text{Dis}} \leftarrow \text{Adam}(\nabla_{\theta_{\text{Dis}}}\frac{1}{m}\sum_{i=1}^m L_{\text{Dis}}^{(i)}, \theta_{\text{Dis}}, \alpha, \beta_1, \beta_2)$
  end for
  sample a batch of latent variables $\{\boldsymbol{z}^{(i)}\}_{i=1}^m \sim p_{\boldsymbol{z}}(\boldsymbol{z})$,
  $\tilde{\boldsymbol{x}}^{(i)} \leftarrow \text{Gen}(\boldsymbol{z}^{(i)})$
  $\hat{\boldsymbol{z}}^{(i)} \leftarrow \text{Enc}(\tilde{\boldsymbol{x}}^{(i)})$
  $L_{\text{Gen}}^{(i)} \leftarrow -\text{Dis}(\tilde{\boldsymbol{x}}^{(i)})$
  $L_{\text{recon\_z}}^{(i)} \leftarrow \left\|\boldsymbol{z}^{(i)} - \hat{\boldsymbol{z}}^{(i)}\right\|_2^2$
  $\theta_{\text{Gen}} \leftarrow \text{Adam}(\nabla_{\theta_{\text{Gen}}}\frac{1}{m}\sum_{i=1}^m L_{\text{Gen}}^{(i)}, \theta_{\text{Gen}}, \alpha, \beta_1, \beta_2)$
  $\theta_{\text{Gen}} \leftarrow \text{Adam}(\nabla_{\theta_{\text{Gen}}}\frac{1}{m}\sum_{i=1}^m {}^{(i)}_{\text{recon\_z}}, \theta_{\text{Gen}}, \frac{\alpha}{5}, \beta_1, \beta_2)$
  $\theta_{\text{Enc}} \leftarrow \text{Adam}(\nabla_{\theta_{\text{Enc}}}(\frac{1}{m}\sum_{i=1}^m L_{\text{recon\_z}}^{(i)} + \eta|\sigma(\{\hat{\boldsymbol{z}}_d^{(i)}\}) - 1|), \theta_{\text{Enc}}, \alpha, \beta_1, \beta_2)$
end while

4. Experiments and Results

In subsection 4.1 we present results of the Reverse-VAE for reconstructed images and randomly synthesized images of the generator. Latent space interpolation is introduced in subsection 4.2. Based on this, in subsection 4.4 visual at-

Table 1: FID of Progressive Reverse-VAE on CelebA 256x256 is similar to DCGAN 64x64 and thus, generates good and diverse images. FID of DCGAN trained by a two time-scale update rule (ttur) and of DCGAN from [17].

Method	learning rate	update	FID
DCGAN TTUR [17]	1e-4, 5e-4	225,000	12.5
DCGAN [17]	5e-4	70,000	21.4
PG Reverse-VAE	1e-3	107,496	29.2

tribute manipulation is proposed. Finally, in subsection 4.5 the Reverse-VAE is applied in anomaly detection.

4.1. Random Generation and Image Reconstruction

We trained and tested the Reverse-VAE model on the MNIST [23], the SVHN data set [27] and the CelebA data set [25].

Figures 4a, 4c, 4e show randomly generated images, tested on the MNIST, SVHN and CelebA data set, respectively looking realistic and diverse. In Figures 4b, 4d, 4f the reconstructed images accurately capture the structure, stroke and slope of the digits in MNIST, the center digits as well as the surrounding distracting digits in SVHN, and the main characteristics of faces, including skin color, hair color, hair line, gesture, and facial emotions in CelebA data set, respectively. Based on the results, we conclude that the Reverse-VAE model successfully learns the mapping from the input images to the latent vectors while generating realistic images.

4.2. Latent Space Interpolation

In order to interpolate between two real images, the encoder converts two real images $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ into the corresponding latent vectors $\tilde{\boldsymbol{z}}_1 = \text{Enc}(\boldsymbol{x}_1)$ and $\tilde{\boldsymbol{z}}_2 = \text{Enc}(\boldsymbol{x}_2)$. Then new points $\boldsymbol{z}_{\text{interp}}$ are linearly sampled between the straight line from $\tilde{\boldsymbol{z}}_1$ to $\tilde{\boldsymbol{z}}_2$ with the interpolation factor γ linearly increasing from 0 to 1:

$$\boldsymbol{z}_{\text{interp}} = \gamma\tilde{\boldsymbol{z}}_2 + (1-\gamma)\tilde{\boldsymbol{z}}_1 \tag{9}$$

Afterwards the generator converts the linearly sampled latent vectors to images $\boldsymbol{x}_{\text{interp}} = \text{Gen}(\boldsymbol{z}_{\text{interp}})$ where $\boldsymbol{x}_{\text{interp}}$ are the interpolated images between two real images.

CelebA interpolated images in Figure 5 look realistic implying that the Reverse-VAE learns latent features which generalize well, and that the probability mass does not concentrate around the latent vectors of training samples.

4.3. Progressively Growing (PG) Reverse-VAE

To scale up our model to generate or reconstruct higher resolution images, we adopted the strategy of progressively growing resolution introduced in PGGAN [18]. We train our Reverse-VAE model starting from a very low resolution, 4×4, then we progressively increase the resolution to 8×8 by adding a block of up-sampling and convolution. During the resolution transition stage, a weight factor α increasing from 0 to 1 linearly is used to weight the contribution of the newly added 8×8 block and the previous 4×4 block to the generation of 8×8 images. For the discriminator and the encoder, similar operations of adding a new higher-resolution block and resolution transition are used. We increase the resolution in this manner until reaching the resolution of 256×256, due to a limitation of computation resources.

We also adopted the PGGAN's strategies of stabilizing the training, including minibatch standard deviation, pixelwise normalization, and equalized learning rate. Furthermore, like PGGAN, we remove the sigmoid activation function at the generator's output and rescale the image pixel value into the range of [-1,1]. During the training at each resolution, we are still use the loss functions introduced in Section 3.2 and the training setting is similar to Alg. 1.

Compared with PGGAN, our model has one extra progressively trained encoder, which increases the application scenarios of our model beyond generating HD images. For example, one can reconstruct an input HD image with good accuracy. This enables our model to be used for high resolution image inpainting which means reconstructing lost or deteriorated parts of images. Moreover, our model can easily perform interpolation between two real images using the method in Section 4.2. Figure 3 shows the image random generation, reconstruction, inpainting and morphing results.

The progressively growing Reverse-VAE is shown to be able to generate realistic images and accurately reconstruct features like hair color, skin, facial emotion and gesture in a large image scale. Although in image inpainting the inpainted area has inconsistent brightness, the facial expression looks natural and coincides well with unmasked area. These results further prove the scalability of our model.

Furthermore, we provide the Fréchet Inception Distance (FID) [17, 7] for random generated images and compare them with other models in Table 1. The smaller FID score is, the higher the quality is and the more diverse the generated images are. The FID of Progressively Growing Reverse-VAE on CelebA 256x256 is similar to DCGAN on CelebA 64x64. The FID confirms that PG Reverse-VAE generates high quality and diverse images.

4.4. Visual Attributes Manipulation

Usage of Feature-wise Reconstruction Loss Although the results of image generation, reconstruction, morphing as well as high resolution image reconstruction show that our model can learn a meaningful embedding and reconstruct the main image features accurately without the feature-wise reconstruction loss $L_{\text{recon\_x}}$ from Equation 1, we do observe

(a) Randomly generated images.

(b) Image reconstructions.

(c) Image inpainting.

(d) Image morphing.

Figure 3: Random image generation, image reconstruction, image inpainting and image morphing using the Progressively Growing Reverse-VAE are tested on CelebA 256×256. For image reconstruction results the first column are input images and the second are reconstructions. For image inpainting we reconstruct the input image in the first column, then only keep the reconstructed area where the mask is, and finally combine it with the input image. For image morhing, the first and last images are real images and the images in between are generated images.

(a) MNIST randomly generated images.

(b) MNIST reconstructions.

(c) SVHN randomly generated images.

(d) SVHN reconstructions.

(e) CelebA randomly generated images.

(f) CelebA reconstructions.

Figure 4: In Figures 4a, 4c, 4e randomly generated images and in Figures 4b, 4d, 4f reconstructions on the MNIST, SVHN and CelebA data set, respectively, are shown. For the reconstruction results, odd columns are the original images from test data set and even columns are the corresponding reconstructions.

adding such a loss to the generator improves the reconstruction performance slightly when tested on the CelebA 64×64. This extra loss may force the model to learn to favor a better reconstruction of the detail. In the experiments of visual attributes manipulation (section 4.4) and anomaly detection (section 4.5), we add the loss $L_{\text{recon\_x}}$ by default, since an accurate reconstruction is important for these two tasks.

In contrast to GAN [12], the visual attributes of images can be analyzed in the latent space and particular latent vec-

Figure 5: The transition from real image in first column to real image in last column (e.g. woman to man) is based on latent space interpolations with γ increasing from 0 to 1.

tors which represent disentangled visual attributes can be extracted. If we want to give a serious face a smile, it is required to add a visual attribute vector $\boldsymbol{v}_{\text{add-smile}}$ which represents the change from 'serious' to 'smiling' to the latent vector of the serious face.

After training, the CelebA validation data set which includes different celebrity identities is used to compute the visual attribute vectors. Each image is labeled with 40 attributes like hair styles, face emotions and hair colors. For each identity i, the encoder maps each smiling face to a latent vector and then the mean latent vector of smiling faces $\bar{\boldsymbol{z}}^{(i)}_{\text{smiling}}$ is calculated. The same is conducted for the serious face to obtain a mean latent vector of serious faces $\bar{\boldsymbol{z}}^{(i)}_{\text{serious}}$. Then 'serious' latent vector is subtracted from 'smiling' latent vector to obtain latent vector of adding smile $\boldsymbol{v}^{(i)}_{\text{add-smile}}$ for the identity i. Afterwards the latent vector of adding smile is averaged over all possible identities to the visual attribute vector $\bar{v}_{\text{add-smile}}$.

After the encoder processes the corresponding latent vector $\boldsymbol{z}_{\text{serious}}$ for a new image of a serious face $\boldsymbol{x}_{\text{serious}}$, the visual attribute vector $\boldsymbol{v}_{\text{add-smile}}$ is added to the latent vector $\boldsymbol{z}_{\text{serious}}$ to get the transformed latent vector $\boldsymbol{z}_{\text{smiling}}$. Finally, the generator receives $\boldsymbol{z}_{\text{smiling}}$ to generate an image with smiling face $\boldsymbol{x}_{\text{smiling}}$. If the visual attribute vector is disentangled, only the desired visual attribute will be manipulated.

Furthermore, we found that the direction of the visual attribute vector $\bar{v}_{\text{add-smile}}$ determines the type of visual attribute, and the magnitude determines the dominance of the visual attribute. A scale factor α is used to adjust the magnitude of the visual attribute vector. This is achieved by adding the scaled visual attribute vector $\alpha\bar{v}_{\text{add-smile}}$ to $\boldsymbol{z}_{\text{serious}}$ to get the converted latent vector $\boldsymbol{z}_{\text{smiling}}$.

Increasing the scale factor α linearly from 0.5 to 2.5 in Figure 1a, the smile on faces is broadened without influencing other facial attributes. We regard the transition of smile as natural and realistic and suppose that the Reverse-VAE model learns disentangled visual attributes. Manipulated images with 10 visual attributes are shown in Figure 1b.

$$\boldsymbol{z}_{\text{attri,sum}} = \boldsymbol{z}_{\text{original}} + \sum_{j=1}^{m} \alpha_j \bar{v}_{\text{add-attri},j} \tag{10}$$

A set of visual attribute vectors $\{\bar{v}_{\text{add-attri},j}\}_{j=1\ldots m}$ is combined linearly and added to the latent vector of an image in eq. 10, where m is the number of visual attributes and j the attribute index. In figure 6, each α_j is empirically chosen such that the visual attribute is equivalently dominant. Finally, the generator takes the latent vector $\boldsymbol{z}_{\text{attri,sum}}$ to generate the image with desired visual attributes.

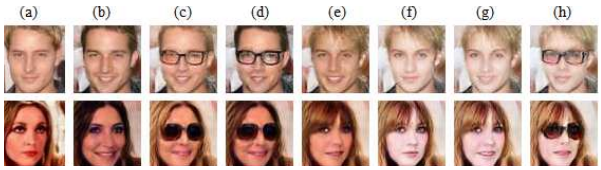


Figure 6: Six different visual attributes are combined: I. Black hair; II. Eyeglasses; III. Smiling; IV. Mouth slightly open; V. Bangs; VI. Pale skin. Each column represents one combination: (a) Original images; (b) Attri. I, III; (c) Attri. II, III; (d) Attri. I, II, III; (e) Attri. IV, V; (f) Attri. V, VI; (g) Attri. IV, V, VI; (h) Attri. II, IV, V, VI.

Different from ALI [8], the Reverse-VAE model is trained without image attributes information. Nevertheless, disentangled visual attributes vectors can be extracted in the latent space learned by the Reverse-VAE, and used for visual attributes manipulation with comparable performance.

Further experiments show that extracting visual attribute vectors without using identity information (such as proposed by Larsen *et al.* [22]) leads to more entangled visual attribute manipulation (e.g gender). In figure 7 adding the visual attribute "blond hair", "heavy makeup" or "pale skin" without identity information to a male face leads to a female face with the desired visual attribute.

4.5. Anomaly Detection

Similar to [3, 37, 31], the image reconstruction error is used to detect anomaly samples. Learning the distribution of training data, the Reverse-VAE can reconstruct the images which are within the distribution of training data with small reconstruction error. For the anomaly images, the reconstruction error is large. Let $\boldsymbol{x}$ denote the input image, $\hat{\boldsymbol{x}}$

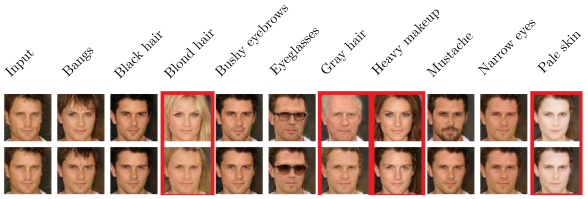


Figure 7: Original images in first column are manipulated without (first row) and with identity information (second row). Visual attribute vectors extracted without using identity information (first row) lead to more entangled visual attribute manipulation.

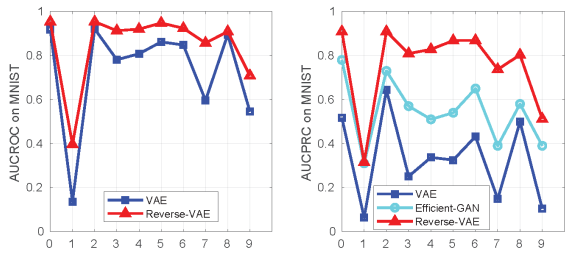


Figure 8: Comparison of anomaly detection performances of VAE [3], Efficient-GAN [37] and Reverse-VAE on the MNIST, evaluated by AUCROC and AUCPRC.

the reconstructed image, and $\text{Dis}_l(x)$ the output of l^{th} layer of the discriminator (here 3rd layer). Reconstruction error $E_{\boldsymbol{x}}$ of input $\boldsymbol{x}$ is defined by:

$$E_{\boldsymbol{x}} = \|\text{Dis}_l(\hat{\boldsymbol{x}}) - \text{Dis}_l(x)\|_2 \tag{11}$$

$E_{\boldsymbol{x}^{(i)}}$ represents the reconstruction error of a sample $\boldsymbol{x}^{(i)}$ from the training set. The anomaly score $A(\boldsymbol{x})$ represents the likelihood that an input image $\boldsymbol{x}$ is an anomaly and is defined by the ratio of number of training samples whose reconstruction error is less than $E_{\boldsymbol{x}}$ to the total number of training samples, whereas $card()$ is the Cardinality sign:

$$A(\boldsymbol{x}) \simeq \frac{card\big(\{\boldsymbol{x}^{(i)} | E_{\boldsymbol{x}^{(i)}} < E_{\boldsymbol{x}}\}\big)}{card\big(\{\boldsymbol{x}^{(i)}\}\big)} \tag{12}$$

The process of anomaly detection is shown in Alg. 2. The Reverse-VAE is evaluated regarding its anomaly detection performance on the MNIST [23].

In **MNIST**, for each type of digits $a \in \{0, 1, ..., 9\}$, we treat digit a as anomaly and all the other digits as normal data. There are 10 different models each trained to detect an anomaly digit respectively. Similar to [3], 80% of the normal data is used for training. The rest 20% of normal data and all the anomaly data are used for testing. Pixels of images are normalized to the range $[0, 1]$. Parameter setting of the generator, discriminator and encoder is the same as for EfficientGAN [37]. The performance of the anomaly detection is evaluated by the area under the curve of the receiver operating characteristic (AUCROC) and the area under the curve of the precision recall curve (AUCPRC).

Figure 8 shows that Reverse-VAE model performs better than VAE [3] and Efficient GAN [37], evaluated by AUCROC. As shown in Figure 9, reconstructions of the anomaly digits resemble samples from normal data set with structural similarity. For example, reconstructions of anomaly digit 7 are mostly 9 or 4. By comparing the reconstructions of anomalies (Figure 9) and normal digits (Figure 4b), we conclude that anomalies can be detected based on reconstruction error, being larger than that of normal digits. Our model has a state-of-the-art performance when evaluated by precision and recall.

Algorithm 2 Process of anomaly detection.

1: Given input image $\boldsymbol{x}$, compute reconstruction error $E_{\boldsymbol{x}}$.
2: Compute the anomaly score $A(\boldsymbol{x})$ according to eq. 12
3: Select threshold ϵ. $\boldsymbol{x}$ is anomaly when $A(\boldsymbol{x}) > \epsilon$, and $\boldsymbol{x}$ is not anomaly when $A(\boldsymbol{x}) \leq \epsilon$.

Figure 9: Reconstructions of anomaly digits are given. The first row show anomaly digits and the second row show corresponding reconstructions.

Nevertheless, the reconstruction error based strategy is vulnerable to the anomaly images which are structurally simple or similarly appears in other samples. This tendency is also found for VAE and Efficient GAN. Especially detecting anomaly digit 1 is worse than random guess. As shown in Figure 9, the reconstructions of anomaly digit 1 (a) with thick stroke is usually thin version of digit 8 or 3; (b) with normal stroke is usually 7 or 9, since its vertical stroke makes up a large part of digit 7 and 9. The simple structure of digit 1 is present in many other digits, so that it is difficult to detect anomaly digit 1.

5. Conclusion and outlook

We introduced the 'Reverse Variational Autoencoder" (Reverse-VAE) for two applications: visual attribute manipulation and anomaly detection. The Kullback-Leibler divergence between joint latent/data-space distribution of generator and the latent/data-space joint distribution of encoder is minimized during training to learn meaningful mapping from data space to latent space. Based on this mapping both applications are enabled. Desired visual attributes of CelebA images are successfully manipulated. The performance of anomaly detection is competitive with state-of-the-art on MNIST. The anomaly detection can be used as a monitor of a Deep Learning (DL) model trained on the same data as the Reverse-VAE. A positive finding could lead to measures for performing in a safe manner. Furthermore, the good scalability enables our model to be up-scaled for high resolution image visual attribute manipulation which can be used for data augmentation in a practical usage scenario.

References

[1] U. Aftab and G. F. Siddiqui. Big data augmentation with data warehouse: A survey. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 2775–2784, Dec 2018. 2

[2] O. A. Aghdam and H. K. Ekenel. Robust deep learning features for face recognition under mismatched conditions. In *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4, May 2018. 1

[3] J. An and S. Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2:1–18, 2015. 7, 8

[4] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. *arXiv e-prints*, page arXiv:1701.07875, Jan 2017. 1, 2, 3, 4

[5] S. Burton, L. Gauerhof, B. B. Sethy, I. Habli, and R. Hawkins. Confidence arguments for evidence of performance in machine learning for highly automated driving functions. In *International Conference on Computer Safety, Reliability, and Security*, pages 365–377. Springer, 2019. 1

[6] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *CoRR*, abs/1606.03657, 2016. 2

[7] D. Dowson and B. Landau. The fréchet distance between multivariate normal distributions. *Journal of multivariate analysis*, 12(3):450–455, 1982. 5

[8] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially Learned Inference. *arXiv e-prints*, page arXiv:1606.00704, Jun 2016. 2, 3, 7

[9] A. Fawzi, O. Fawzi, and P. Frossard. Analysis of classifiers' robustness to adversarial perturbations. In *arXiv:1502.02590*, 2015. 1

[10] A. Fawzi, H. Samulowitz, D. Turaga, and P. Frossard. Adaptive data augmentation for image classification. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3688–3692, Sep. 2016. 2

[11] L. Gauerhof, P. Munk, and S. Burton. Structuring validation targets of a machine learning function applied to automated driving. In B. Gallina, A. Skavhaug, and F. Bitsch, editors, *Computer Safety, Reliability, and Security*, pages 45–58, Cham, 2018. Springer International Publishing. 1

[12] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Networks. *arXiv e-prints*, page arXiv:1406.2661, June 2014. 1, 2, 6

[13] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017. 4

[14] M. A. Hanif, F. Khalid, R. V. W. Putra, S. Rehman, and M. Shafique. Robust machine learning systems: Reliability and security for deep neural networks. In *2018 IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS)*, pages 257–260, July 2018. 1

[15] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and surface variations. *arXiv preprint arXiv:1807.01697*, 2018. 1

[16] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *CoRR*, abs/1610.02136, 2016. 1

[17] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017. 5

[18] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 1, 2, 3, 5

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 4

[20] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014. 2, 4

[21] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 03 1951. 3

[22] A. B. L. Larsen, S. K. Sønderby, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. *CoRR*, abs/1512.09300, 2015. 2, 7

[23] Y. LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. 2, 5, 8

[24] J. Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991. 3

[25] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015. 5

[26] A. Makhzani, J. Shlens, N. Jaitly, and I. J. Goodfellow. Adversarial autoencoders. *CoRR*, abs/1511.05644, 2015. 2, 3

[27] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, 2011. 5

[28] X. Peng, Z. Tang, F. Yang, R. S. Feris, and D. Metaxas. Jointly optimize data augmentation and network training: Adversarial data augmentation in human pose estimation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2226–2234, June 2018. 2

[29] J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. Dataset shift in machine learning. MIT Press, 2017. 1

[30] Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 59–66. IEEE, 1998. 3

[31] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International Conference on Information Processing in Medical Imaging*, pages 146–157. Springer, 2017. 7

[32] C. Schorn, A. Guntoro, and G. Ascheid. Efficient on-line error detection and mitigation for deep neural network accelerators. In B. Gallina, A. Skavhaug, and F. Bitsch, editors, *Computer Safety, Reliability, and Security*, pages 205–219, Cham, 2018. Springer International Publishing. 1

[33] H. Shi, L. Wang, G. Ding, F. Yang, and X. Li. Data augmentation with improved generative adversarial networks. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 73–78, Aug 2018. 2

[34] J. Shijie, W. Ping, J. Peiyi, and H. Siping. Research on data augmentation for image classification based on convolution neural networks. In *2017 Chinese Automation Congress (CAC)*, pages 4165–4170, Oct 2017. 2

[35] K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491, 2015. 2

[36] J. Su. Variational inference: A unified framework of generative models and some revelations. *CoRR*, abs/1807.05936, 2018. 3

[37] H. Zenati, C. S. Foo, B. Lecouat, G. Manek, and V. R. Chandrasekhar. Efficient gan-based anomaly detection. *arXiv preprint arXiv:1802.06222*, 2018. 7, 8