UNLABELED DISENTANGLING OF GANS WITH GUIDED SIAMESE NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial learning has greatly improved the abilities of generative models to learn from data (e.g. images) and generate convincing data samples by sampling from simple latent noise distributions. However, applications which could benefit from this generative power, such as image manipulation, are hindered by the lack of controlling specific interpretable factors in the latent space of the learned models. Guiding the sampling process to produce sample variation which is semantically disentangled and meaningful for humans is not possible with current generative models unless large amounts of attribute labelled data are available. In this paper, we propose Unlabeled Disentangling GAN (UD-GAN), which decomposes the latent noise space into semantically meaningful dimensions, using only weak supervision, and without using labels or explicit attributes. When we sample a new data point we can independently control each latent noise slice of UD-GAN and manipulate in a targeted fashion the properties of generated data. Our contributions encompass the introduction of our novel architecture combining an adversarial with contrastive loss, an analysis and probabilistic interpretation of the loss function, and multiple experiments to illustrate the capabilities of our method.

1 INTRODUCTION

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) are generative model estimators, where two neural networks (generator and discriminator) are trained in an adversarial setting, so that likelihood-based probabilistic modeling is not necessary. This works particularly well for sampling from a complex probability distribution, such as images. Although GANs yield realistic looking images (Radford et al., 2015), the original formulation in (Goodfellow et al., 2014) only allows for randomly sampling from the data distribution without structural or semantic control over the generated data points. For example, it might be desirable for an application to control the color of an object in an image independently from its other attributes.

One way to control the sampling process is to use conditional GANs (Mirza & Osindero, 2014; Odena et al., 2017). These models modify the generator by conditioning it with supervised labels. Then, they either take the same labels as input in the discriminator (Mirza & Osindero, 2014) and measure the image-label compatibility, or classify the correct label at the output, given the generated image (Odena et al., 2017). Conditional GANs rely on a dataset with labels, which might not always be available or might be time-consuming to collect.

In this paper, we propose Unlabeled Disentangling GAN (UD-GAN), which decomposes various semantic attributes of a dataset without using labels, and therefore enables to independently control each attribute when sampling a data point. This control is possible by passing the generated images through multiple Siamese Networks (Chopra et al., 2005), each of which embeds the images into a different representation space.

2 RELATED WORK

There have been many studies on learning disentangled representations in generative models, which can be grouped into the level of supervision/labeled data they require.

Disentangled representations (supervised). In (Zhu et al., 2014; Yang et al., 2015), the identity and the viewpoint of an object are disentangled via reconstructing the same object from a different viewpoint and minimizing a reconstruction loss. Whereas in (Kingma et al., 2014; Makhzani et al., 2016), the style and category of an object is separated via autoencoders, where an encoder embeds the style of an input image to a latent representation, and a decoder takes the category and style input to reconstruct the input image. In (Tran et al., 2017; Yin et al., 2017), autoencoders and GANs are combined to decompose identity and attribute of an object, where the disentangled representation is obtained at the encoder outputs, and image labels are used at the output of the discriminator.

Disentangled representations (semi-supervised). In (Reed et al., 2014), they clamp the hidden units for a pair of images with the same identity but with different pose or expression to have the same identity representation. Whereas in (Kulkarni et al., 2015), synthesized images are used to disentangle pose, light, and shape of an object by passing a batch of images where only one attribute varies and the rest of the representation is clamped to be the same. These techniques only require a batch of samples with one attribute different at a time.

Disentangled representations (unsupervised). InfoGAN (Chen et al., 2016) is an unsupervised technique that discovers categorical and continuous factors by maximizing the mutual information between a GAN's noise variables and the generated image. The method uncovers disentangled latent spaces that correspond to data properties. After a model is trained, a human needs to investigate which factors map to which data property. As the method is unsupervised not all desirable factors might be represented. In contrast, our method can benefit from weak supervision and allows learning of factors which correspond to the intuition of the system designer.

3 UNLABELED DISENTANGLING GAN

3.1 BACKGROUND: GENERATIVE ADVERSARIAL NETWORKS

In GANs, the generator, $G(.)$, maps a latent variable $\mathbf{z}$, which has an easy-to-sample distribution, into a more complex and unknown distribution, such as images. On the other hand, the discriminator $D(.)$ tries to distinguish real images from the ones that are generated by G. In (Goodfellow et al., 2014), the training is performed as a minimax game as follows:

$$\min_{G}\max_{D} V(G,D) = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathbb{P}_{\mathrm{R}}}[\log D(\mathbf{x})] + \mathop{\mathbb{E}}_{\mathbf{z}\sim\mathbb{P}_{\mathrm{Z}}}[\log(1 - D(G(\mathbf{z})))], \tag{1}$$

where $\mathbb{P}_{\mathrm{R}}$ and $\mathbb{P}_{\mathrm{Z}}$ are the probability distributions of real images and images generated from the latent variable $\mathbf{z}$, respectively. Recently, there have been improvements on training speed and stability of GANs (Berthelot et al., 2017; Seward et al., 2018; Arjovsky et al., 2017). One such example is to use Wasserstein-1 distance along with a gradient penalty (Gulrajani et al., 2017) as follows:

$$\min_{G}\max_{D\in\mathcal{D}} V(G,D) = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathbb{P}_{\mathrm{R}}}[D(\mathbf{x})] - \mathop{\mathbb{E}}_{\mathbf{z}\sim\mathbb{P}_{\mathrm{Z}}}[D(G(\mathbf{z}))] + \lambda_{GP}\mathop{\mathbb{E}}_{\hat{\mathbf{x}}\sim\mathbb{P}_{\hat{\mathbf{x}}}}[(||\nabla_{\hat{\mathbf{x}}}D(\hat{\mathbf{x}})||_2 - 1)^2]. \tag{2}$$

Here, $\mathcal{D}$ is the set of 1-Lipschitz functions, λ_{GP} is the coefficient for the gradient penalty, and $\hat{\mathbf{x}}$ is a pixel-wise linear combination of a real and a generated image, and is distributed with $\mathbb{P}_{\hat{\mathbf{x}}}$. We train the GAN part of our model using the loss in equation 2.

3.2 A NOVEL GAN ARCHITECTURE: UD-GAN

In a standard GAN setting, all of the variation in the distribution of real images is captured by the latent variable $\mathbf{z}$. However, a single dimension or a slice of $\mathbf{z}$ does not necessarily have a semantic meaning. In this paper, our target is to slice the latent variable into multiple vectors, where each vector controls a different semantic variation.

Our network architecture is visualized in Figure 1. In our method, the latent vector $\mathbf{z} = [\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_{N_A}, \mathbf{r}]$ is the concatenation of N_A *knobs*, $\{\mathbf{q}_i\}_{i=1}^{N_A}$, which represent different attributes we aim to disentangle, and a final variable $\mathbf{r}$, which captures the remaining variation (and the noise) that is not picked up by the knobs. In our notation, $\mathbf{q}_{\bar{i}}$ refers to all of the knobs, except $\mathbf{q}_i$. Each latent variable is sampled from a multivariate, zero-mean, unit-variance normal distribution.

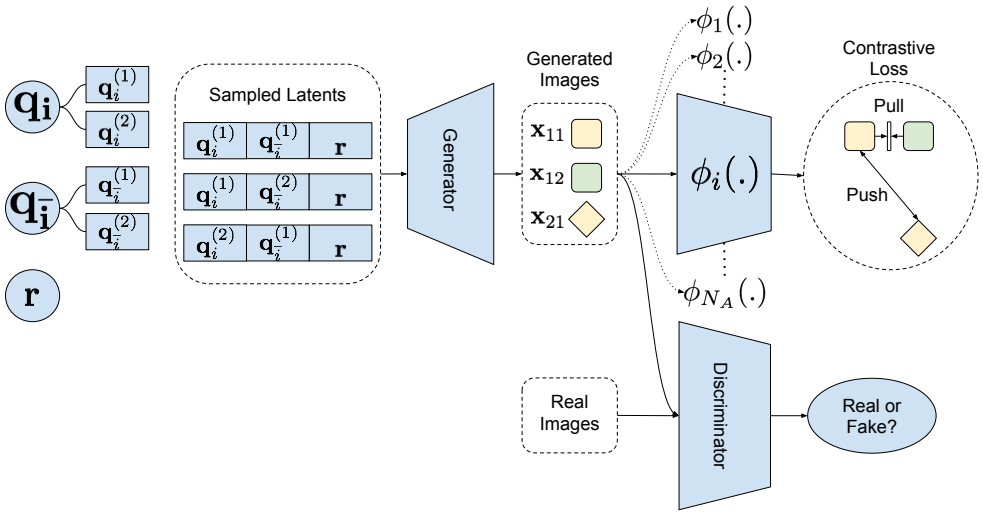


Figure 1: The flowchart of our architecture. Sampled latents from different attributes are combined into latent vectors. Generated images are grouped with respect to different attributes (here, represented by shape) by Siamese Networks (denoted as ϕ_i).

In order to train our model, first, for each $\mathbf{q}_i$, we sample two different vectors, $\mathbf{q}_i^{(1)}$ and $\mathbf{q}_i^{(2)}$, from the noise prior. If we would attempt to form a batch by combinatorially concatenating all knob samples, we get a batch size of 2^{N_A}, which grows exponentially with the number of attributes. To avoid this computational burden, we train our model through stochastic sampling of one attribute at a time. For example, if the ith attribute is chosen, we generate three images using the following combinations:

$$\mathbf{x}_{11} = G(\mathbf{q}_i^{(1)}, \mathbf{q}_{\bar{i}}^{(1)}, \mathbf{r}), \quad \mathbf{x}_{12} = G(\mathbf{q}_i^{(1)}, \mathbf{q}_{\bar{i}}^{(2)}, \mathbf{r}), \quad \mathbf{x}_{21} = G(\mathbf{q}_i^{(2)}, \mathbf{q}_{\bar{i}}^{(1)}, \mathbf{r}). \tag{3}$$

Here, $\{\mathbf{x}_{11}, \mathbf{x}_{12}, \mathbf{x}_{21}\}$ are the generated images, $\{\mathbf{q}_i^{(1)}, \mathbf{q}_i^{(2)}\}$ are the samples for the ith knob, and $\{\mathbf{q}_{\bar{i}}^{(1)}, \mathbf{q}_{\bar{i}}^{(2)}\}$ represent the samples for all knobs except $\mathbf{q}_i$. The main premise is that the two images that are generated with the same $\mathbf{q}_i$ vectors, namely $\{\mathbf{x}_{11}, \mathbf{x}_{12}\}$, should have the same i^{th} attribute, regardless of the values of $\mathbf{q}_{\bar{i}}$. Similarly, $\{\mathbf{x}_{11}, \mathbf{x}_{21}\}$ should have a different i^{th} attribute. We can ensure this via employing a Contrastive Loss (Hadsell et al., 2006) on generated image pairs. To this end, the generated images are processed by Siamese Networks (Chopra et al., 2005), which we denote as $\phi_i(.)$. These networks compute embedding vectors for each generated image, which are used to calculate distances in these spaces. We then use them to disentangle the generation process, with the goal such that if we change the value of a knob $\mathbf{q}_i$, only the corresponding embedding vector $\phi_i(\mathbf{x})$ changes, and the rest of the embedding vectors stay the same. We use a Contrastive Loss function to pull similar image pairs together, and push dissimilar pairs apart. We can write the loss as:

$$\mathcal{L}_{\phi_i} = \mathop{\mathbb{E}}_{\{\mathbf{q}_i^{(1)}, \mathbf{q}_i^{(2)}, \mathbf{q}_{\bar{i}}^{(1)}, \mathbf{q}_{\bar{i}}^{(2)}, \mathbf{r}\} \sim \mathcal{N}} \big|\big| \phi_i(\mathbf{x}_{11}) - \phi_i(\mathbf{x}_{12}) \big|\big|_2^2 + \max(0, \gamma - \big|\big| \phi_i(\mathbf{x}_{11}) - \phi_i(\mathbf{x}_{21}) \big|\big|_2)^2, \tag{4}$$

where, $\mathcal{L}_{\phi_i}$ is the Contrastive Loss for the ith Siamese Network ϕ_i, and γ is the margin for the Euclidean distance between two image embeddings.

The discriminator network D is not modified and is trained to separate real and generated image distributions. Donahue et al. (2018) use a similar latent variable slicing for capturing illumination and pose variations of a face with a fixed identity. Their discriminator needs image pairs, which must be labeled for real images, to judge the quality and identity of the faces. Our method does not require any labels for the real images. Instead, we create pseudo-similar and dissimilar image pairs via concatenating latent variables and generating image batches. Our final loss function is:

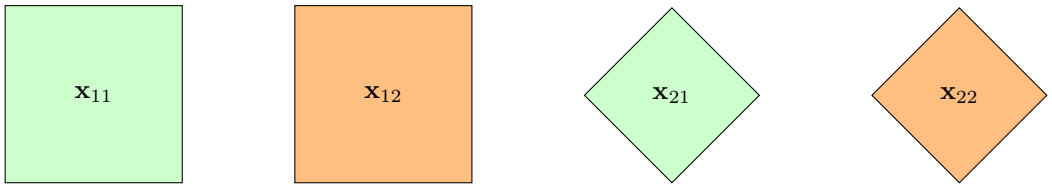
$$\mathcal{L}_\phi = \mathcal{L}_{\phi_i}, \quad i \sim \text{Cat}(N_A)$$
$$\min_G \max_{D \in \mathcal{D}} V(G, D) = \mathcal{L}_{WGAN} + \lambda_\phi \, \mathcal{L}_\phi, \tag{5}$$

where, $\mathcal{L}_{WGAN}$ is the Wasserstein loss described in equation 2, λ_ϕ is the weight of the embedding loss, and the sampling of the latent variables depends on i and is performed as described above.

3.3 PROBABILISTIC INTERPRETATION

Siamese Networks, or similar energy-based models, have been used to learn an invariant mapping for a particular task (Chopra et al., 2005). In our case, if each Siamese Network ϕ_i implements a mapping to different semantic attributes, we can achieve disentanglement, and consequently, control over the generation process.

We can gain a probabilistic interpretation of this process on a toy example. Let us assume a problem, where we want to generate images of colored polygons (see Figure 1). The two knobs $\mathbf{q}_i$ and $\mathbf{q}_j$ represent the shape and the color of a polygon, respectively. $\mathbf{r}$ controls the remaining variation in the data distribution, i.e., there can be variation other than shape and color, such as location. When we set $\mathbf{q}_j$ to a certain value and vary $\mathbf{q}_i$, we want to generate polygons with the same color, but different shapes, and vice versa. In order to ensure that, we can get two samples for $\mathbf{q}_i$ and $\mathbf{q}_j$ (the rest of the knobs have the same value), and generate four images as shown in Figure 2:



(a) $G(\mathbf{q}_i^{(1)}, \mathbf{q}_j^{(1)}, ..)$ (b) $G(\mathbf{q}_i^{(1)}, \mathbf{q}_j^{(2)}, ..)$ (c) $G(\mathbf{q}_i^{(2)}, \mathbf{q}_j^{(1)}, ..)$ (d) $G(\mathbf{q}_i^{(2)}, \mathbf{q}_j^{(2)}, ..)$

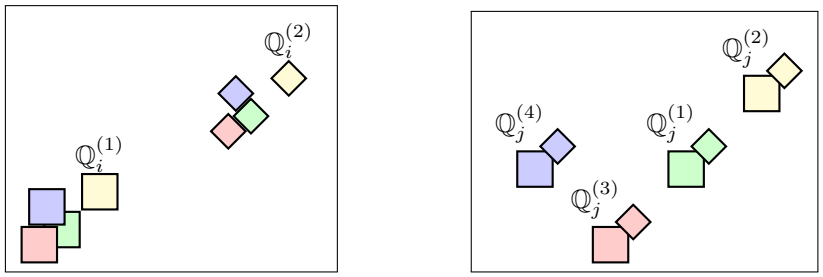
Figure 2: A combination of knobs are used to generate four images with two varying attributes (in this case, shape and color).

Let $\mathbb{P}$ be the probability distribution of colored polygons. For each attribute, $\mathbb{P}$ can be decomposed into a mixture distribution as follows:

$$\mathbb{P} = \sum_{k=1}^{N_i} \pi_i^{(k)} \mathbb{Q}_i^{(k)} \quad \leftarrow \text{for attribute } i, \qquad \mathbb{P} = \sum_{k=1}^{N_j} \pi_j^{(k)} \mathbb{Q}_j^{(k)} \quad \leftarrow \text{for attribute } j \tag{6}$$

where, $\mathbb{Q}_i^{(k)}$ is a mixture component and $\pi_i^{(k)}$ is its corresponding probability of choosing it, and N_i is the number of different values an attribute (in the example case, i corresponds to shape) can take. A similar explanation can be made for attribute j, i.e. color. For the sake of this analysis, we accept that for each attribute, $\mathbb{P}$ can be decomposed into different discrete mixture distributions as shown in Figure 3. For this specific case, $\mathbb{Q}_i^{(1)}$ and $\mathbb{Q}_i^{(2)}$ are the distributions of colored squares and colored diamonds, respectively. For the color attribute, which is indexed by j, each $\mathbb{Q}_j^{(k)}$ corresponds to a distribution of polygons with a single color (i.e., *green polygons*). Our Siamese Networks have two tasks. First, they need to separate all $\mathbb{Q}_i^{(k)}$ from each other within each attribute i. This is achieved via the second term of the contrastive loss in equation 4. Second, the networks need to minimize the variation within each mixture component $\mathbb{Q}_i^{(k)}$. If all samples from $\mathbb{Q}_i^{(k)}$ are mapped to the same embedding vector, the effect of j (and any other attribute) on determining the shape disappears and disentangling is achieved. The first term in equation 4 is minimizing the variance at each mixture component. In the following, we focus on the shape attribute that is represented by i, however, derivations carry over to the color attribute j.

In order to push mixture components apart and minimize their variances, we can maximize a divergence between all pairs from $\mathbb{Q}_i^{(k)}$. One way to measure the distance between these distributions is



(a) ϕ_i-space (shape) (b) ϕ_j-space (color)

Figure 3: Illustration of the embedding spaces and separated probability distributions after training our model.

to use the energy distance (Székely & Rizzo, 2004):

$$\begin{aligned} D_E(\mathbb{Q}_i^{(1)}, \mathbb{Q}_i^{(2)}; \phi_i, j) = & \mathop{\mathbb{E}}_{\mathbf{x}_{1n_j} \sim \mathbb{Q}_i^{(1)}, \mathbf{x}_{2n'_j} \sim \mathbb{Q}_i^{(2)}} \left[\rho_i(\mathbf{x}_{1n_j}, \mathbf{x}_{2n'_j}) \right] \\ & - \frac{1}{2} \mathop{\mathbb{E}}_{\mathbf{x}_{1n_j}, \mathbf{x}_{1n'_j} \sim \mathbb{Q}_i^{(1)}} \left[\rho_i(\mathbf{x}_{1n_j}, \mathbf{x}_{1n'_j}) \right] \\ & - \frac{1}{2} \mathop{\mathbb{E}}_{\mathbf{x}_{2n_j}, \mathbf{x}_{2n'_j} \sim \mathbb{Q}_i^{(2)}} \left[\rho_i(\mathbf{x}_{2n_j}, \mathbf{x}_{2n'_j}) \right], \end{aligned} \tag{7}$$

where the index n_j represents an image that is generated with a sample $\mathbf{q}_j^{(n_j)}$ for the jth knob. The function ρ_i has the form:

$$\rho_i(\mathbf{x}, \mathbf{x}') = \left|\left| \phi_i(\mathbf{x}) - \phi_i(\mathbf{x}') \right|\right|_2^2. \tag{8}$$

The energy distance in equation 7 can be interpreted as an instance of Maximum Mean Discrepancy (Bińkowski et al., 2018) and resembles the Contrastive Loss (Hadsell et al., 2006). If we rewrite equation 7 with an unbiased estimator, we have:

$$\begin{aligned} D_E(\mathbb{Q}_i^{(1)}, \mathbb{Q}_i^{(2)}; \phi_i, j) = & \frac{1}{4} \sum_{n_j=1}^{2} \sum_{n'_j=1}^{2} \left|\left| \phi_i(\mathbf{x}_{1n_j}) - \phi_i(\mathbf{x}_{2n'_j}) \right|\right|_2^2 \\ & - \frac{1}{2} \left|\left| \phi_i(\mathbf{x}_{11}) - \phi_i(\mathbf{x}_{12}) \right|\right|_2^2 \\ & - \frac{1}{2} \left|\left| \phi_i(\mathbf{x}_{21}) - \phi_i(\mathbf{x}_{22}) \right|\right|_2^2. \end{aligned} \tag{9}$$

We can rewrite equation 9 using the Contrastive Loss of equation 4 as follows:

$$\begin{aligned} D_E(\mathbb{Q}_i^{(1)}, \mathbb{Q}_i^{(2)}; \phi_i, j) = & \frac{1}{4} \sum_{n_j=1}^{2} \sum_{n'_j=1}^{2} \left|\left| \phi_i(\mathbf{x}_{1n_j}) - \phi_i(\mathbf{x}_{2n'_j}) \right|\right|_2^2 \\ & + \frac{1}{4} \sum_{n_j=1}^{2} \sum_{n'_j=1}^{2} \max(0, \gamma - \left|\left| \phi_i(\mathbf{x}_{1n_j}) - \phi_i(\mathbf{x}_{2n'_j}) \right|\right|_2)^2 \\ & - \mathcal{L}_{\phi_i}. \end{aligned} \tag{10}$$

Note that it was used that $\{\mathbf{x}_{11}, \mathbf{x}_{12}\}$ and $\{\mathbf{x}_{21}, \mathbf{x}_{22}\}$ are equivalent positive pairs, $\{\mathbf{x}_{11}, \mathbf{x}_{21}\}$ and $\{\mathbf{x}_{11}, \mathbf{x}_{22}\}$ are equivalent negative pairs. The sum of the first two terms is a piecewise quadratic function and has its minimum at $\left|\left| \phi_i(\mathbf{x}) - \phi_i(\mathbf{x}') \right|\right|_2 = \gamma/2$ and the value of the minimum is $\gamma^2/2$. So, we can rewrite equation 10 as follows:

$$D_E(\mathbb{Q}_i^{(1)}, \mathbb{Q}_i^{(2)}; \phi_i, j) \geq \frac{\gamma^2}{2} - \mathcal{L}_{\phi_i}. \tag{11}$$

Therefore, as the margin γ is a constant, minimizing our embedding loss $\mathcal{L}_{\phi_i}$ maximizes the lower bound for the energy distance D_E. This corresponds to learning a Siamese Network ϕ_i that separates the probability distributions $\mathbb{Q}_i^{(1)}$ and $\mathbb{Q}_i^{(2)}$, i.e., colored squares and colored diamonds, from each other and minimizes the variance within each distribution, thus resulting in disentangling the effect of j from i. The same derivation can be made for the color attribute. After jointly training the Siamese Networks, we hope to achieve the embedding spaces represented in Figure 3.

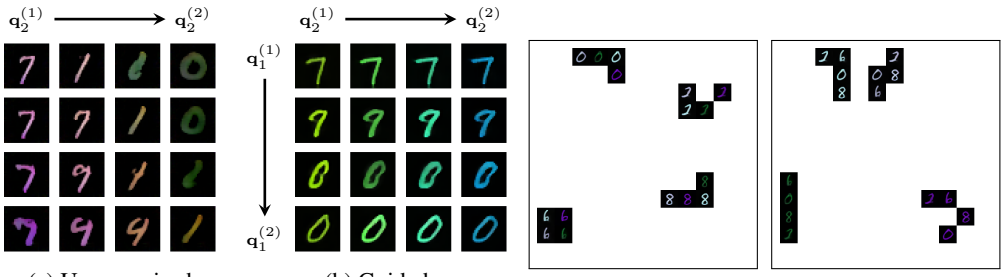
3.4 Guided Siamese Networks

The Siamese Networks ϕ_i are desired to map images into embedding spaces, where they can be grouped within a distinct semantic context. For the example shown in Figure 4, where we disentangle the shape and the color, this might not be achievable in a completely unsupervised setting, because the separation in equation 6 is not unique. However, we can still benefit from the disentangling capability of our method via small assumptions and domain knowledge, without collecting labeled data.

Consider the toy example, where we extend the MNIST dataset (LeCun & Cortes, 2010) to have a random color, sampled from a uniform RGB color distribution. We define our problem to independently capture the shape of a digit with $\mathbf{q}_1$ and its color with $\mathbf{q}_2$.

In Figure 4(a), we show images created by a generator, which is trained along with two networks, ϕ_1 and ϕ_2, without any guidance in an unsupervised setting. We can see that the knobs, $\mathbf{q}_1$ and $\mathbf{q}_2$, capture the variations in the data, however, these variations are not semantically meaningful. Each knob modifies a complicated combination of shape and color.

However, if we *design* a network architecture in a *slightly smarter* way, we should be able to separate the shape and the color attributes. This is exemplified in Figure 4(b), where instead of feeding the whole image to ϕ_2, we feed the average color of some randomly sampled pixels from a generated image. This choice prevents ϕ_2 to capture the spatial structure of the generated digit and to focus only on color. After the training our method with a modified ϕ_2, the first network captures shape of a digit, and the second one captures the color variations. This can also be observed in Figure 4(c) and 4(d), where we use t-SNE (van der Maaten & Hinton, 2008) to visualize embedding spaces for shape and color, respectively.



(a) Unsupervised (b) Guided (c) Shape t-SNE (d) Color t-SNE

Figure 4: (a) Samples from the colored version of the MNIST dataset. (b) Images generated after an unsupervised training with two knobs and (c) after a guided training (the knob values are interpolated between two values and then concatenated to generate the final image). (d) The t-SNE representation of the embedding vectors for shape and (e) color.

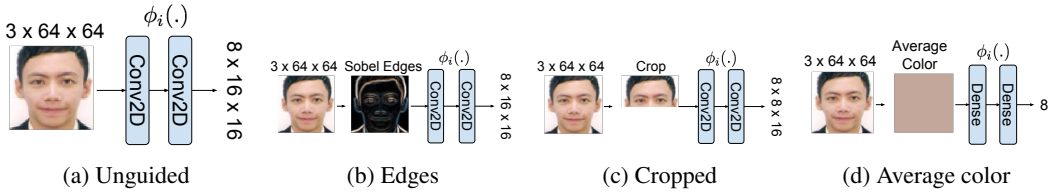
4 Experiments

4.1 Setup

We perform our experiments on a server with Intel Xeon Gold 6134 CPU, 256GB system memory, and an NVIDIA V100 GPU with 16GB of graphics memory. Our generator and discriminator layers are the same as in (Karras et al., 2018), except our architecture is modified to generate 64×64 pixel

images. Each knob is a 4-dimensional slice of the latent variable, and the remaining variation vector $\mathbf{r}$ has 128 dimensions.

In order to guide the training, we modify our Siamese Networks to obtain different effects as illustrated in Figure 5. We adjust our guided networks to map images into 8-dimensional embeddings, which are on patch-level for Figure 5(a), 5(b), and 5(c), and image-level for Figure 5(d). The embedding vectors are normalized with L_2 norms along the feature dimensions.



(a) Unguided (b) Edges (c) Cropped (d) Average color

Figure 5: Siamese networks can be guided to measure (or ignore) certain information via feeding (a) a generated image, (b) detected edges of an image, (c) a cropped image, or (d) its average color.

We use ADAM (Kingma & Ba, 2014) as an optimizer for our training with the following parameters: learning rate=0.001, $\beta_1 = 0$, $\beta_1 = 0$, $\beta_2 = 0.99$, $\epsilon = 10^{-8}$. The weight values for the Wasserstein gradient penalty is $\lambda_{GP} = 10$ and for the contrastive loss is $\lambda_\phi = 100$.

Our training images are from the aligned MS-Celeb-1M dataset (Guo et al., 2016). We randomly sample one picture for each celebrity, obtain a dataset of around 100K images, and resize the images to 64×64 pixels. This random sampling prevents our model to rely on near-identical images in the dataset and makes the learning process generalize better.

4.2 Unguided Disentangling

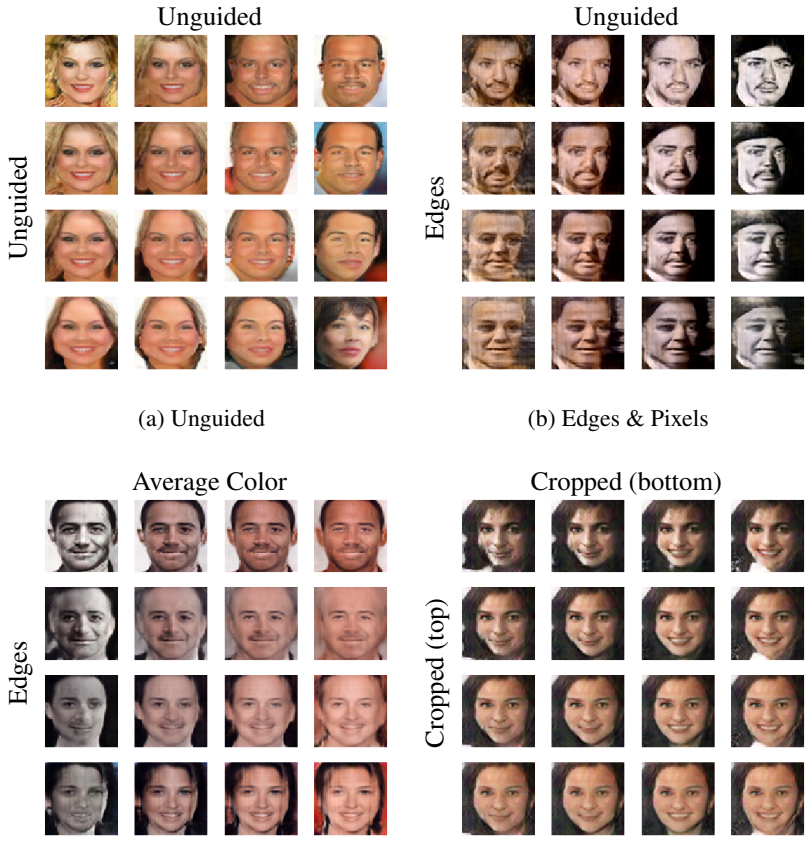
We first train a GAN with two attributes with the unguided network in Figure 5(a). Results from this experiment are illustrated in Figure 6(a). Here, we can see that, two knobs capture some variations in the data that are coupled, i.e. they both modify similar attributes, and are not semantically meaningful. This result is similar to what we obtain for colored MNIST digits in Figure 4(a).

4.3 Guided Disentangling

In order to show the effect of the guided siamese networks, we perform three experiments on the MS-Celeb dataset by using different guiding networks from Figure 5. In the first experiment, only one of the two networks is guided with an edge detector at the input, as shown in Figure 5(b). Results of this experiment are shown in Figure 6(b). We can see that the first knob, which is connected to edges, captures the overall outline and roughly controls the identity of the generated face. On the other hand, the unguided second knob modifies the image with minimal changes to image edges. This change, in this case, corresponds to the lighting of the face.

We perform a second experiment with the edge detector, where in this case, the second knob is guided with the average color of the generated image (see Figure 5(d)). In Figure 6(c), we can observe the results of our disentangled image manipulation. The first knob with the edge detector again captures the outline of the face, and the second average color knob modifies a combination of the light and the skin color, similar to the results in Figure 6(b).

In our final experiment, we employ the cropped guidance networks in Figure 5(c). The two knobs receive the cropped top and bottom part of the image for training. Although these image crops are not independent, we still get acceptable results that are shown in Figure 6(d). Adjusting the first knob only modifies the upper part of the face; the hair and the eyes. Similarly, the second knob is responsible for determining the chin and mouth shape.



(a) Unguided

(b) Edges & Pixels

(c) Edges & Average Color

(d) Cropped

Figure 6: The results of UD-GAN using differently guided siamese networks.

4.4 Discussion

In (Dumoulin et al., 2017; Donahue et al., 2016; Zhu et al., 2017), in order to perform semantic manipulation, an encoder infers the latent representation of an input image. Here, we instead employ multiple Siamese Networks, each of which is related to independent slices of the latent variable, thus enabling disentangled image manipulation with targeted control over potentially predefined features.

In our experiments, we use the aligned MS-Celeb dataset, where each face is roughly centered around the nose. This reduces the variation and simplifies the problem of embedder design. We can employ our method in more complex scenarios with better guided networks and pre-trained models. From a general perspective, we can use any differentiable function, whether it is fixed, trainable, or pre-trained, as our siamese networks ϕ_i. In addition, with pre-trained models, information learned on one dataset can be used to disentangle other unlabeled datasets.

5 Conclusion

In this paper we introduced UD-GAN, a novel GAN formulation which employs Siamese networks with contrastive losses in order to make slices of the latent noise space disentangled and more semantically meaningful. Our experiments encompassed guided and unguided approaches for the embedding networks, and illustrate how UD-GAN can be used for semantically meaningful image manipulation. Our visualisations of the embedding spaces, and the performed image manipulations confirm that our method can adjust well to the intrinsic factors of variation of the data. In future work we plan to investigate more powerful forms of embedders, e.g. extracting information from pre-trained networks for semantic segmentation and landmark detection can allow for even more powerful novel image manipulation techniques.

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, 2017.

David Berthelot, Tom Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *CoRR*, 2017.

M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. Demystifying MMD GANs. *ICLR*, 2018.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *NIPS*, 2016.

Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005.

C. Donahue, Z. C. Lipton, A. Balsubramani, and J. McAuley. Semantically Decomposing the Latent Spaces of Generative Adversarial Networks. *ICLR*, 2018.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *CoRR*, 2016.

Vincent Dumoulin, Mohamed Ishmael Diwan Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. Adversarially learned inference. 2017.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*. 2014.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*. 2017.

Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. MS-Celeb-1M: A dataset and benchmark for large scale face recognition. In *European Conference on Computer Vision*, 2016.

R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. *CVPR*, 2006.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, 2014.

Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*. 2014.

Tejas D Kulkarni, William F. Whitney, Pushmeet Kohli, and Josh Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*. 2015.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow. Adversarial autoencoders. In *ICLR*, 2016.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *CoRR*, 2014.

A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. In *ICML*, 2017.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, 2015.

Scott Reed, Kihyuk Sohn, Yuting Zhang, and Honglak Lee. Learning to disentangle factors of variation with manifold interaction. In *ICML*, 2014.

Calvin Seward, Thomas Unterthiner, Urs Bergmann, Nikolay Jetchev, and Sepp Hochreiter. First order generative adversarial networks. In *ICML*, 2018.

Gábor J. Székely and Maria L. Rizzo. Testing for equal distributions in high dimensions. *InterStat*, 2004.

L. Tran, X. Yin, and X. Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, 2017.

Laurens van der Maaten and Geoffrey Hinton. *Journal of Machine Learning Research*, 2008.

Jimei Yang, Scott Reed, Ming-Hsuan Yang, and Honglak Lee. Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. In *NIPS*, 2015.

Weidong Yin, Yanwei Fu, Leonid Sigal, and Xiangyang Xue. Semi-latent gan: Learning to generate and modify facial images from attributes. *CoRR*, 2017.

Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems*. 2017.

Zhenyao Zhu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Multi-view perceptron: a deep model for learning face identity and view representations. In *NIPS*, 2014.