Deformable Generator Network: Unsupervised Disentanglement of Appearance and Geometry

Xianglei Xing 1, Ruiqi Gao 2, Tian Han 2, Song-Chun Zhu 2, and Ying Nian Wu 2

1College of Automation, Harbin Engineering University, Heilongjiang, China
2Department of Statistics, University of California, Los Angeles, California, USA

Abstract

We propose a deformable generator model to disentangle the appearance and geometric information from images into two independent latent vectors. The appearance generator produces the appearance information, including color, illumination, identity or category, of an image. The geometric generator produces displacement of the coordinates of each pixel and performs geometric warping, such as stretching and rotation, on the appearance generator to obtain the final synthesized image. The proposed model can learn both representations from image data in an unsupervised manner. The learned geometric generator can be conveniently transferred to the other image datasets to facilitate downstream AI tasks.

1 Introduction

A fundamental challenge in developing a conceptual understanding of our world is learning the factorial structure of the observations without supervision [3,27]. Conceptual understanding requires a disentangled representation which separates the underlying explanatory factors and explicitly represents the important attributes of the real-world data [11,35]. For instance, given an image dataset of human faces, a disentangled representation can include the face’s appearance attributes, such as color, light source, identity, gender, and the geometric attributes, such as face shape and viewing angle. A disentangled representation is useful not only for building more transparent and interpretable generative models, but also for a large variety of downstream AI tasks such as transfer learning and zero-shot inference where humans excel but machines struggle [23]. Many exciting applications require generative models that can synthesize novel instances while certain key factors of variation are held fixed. Potential applications include generating a face image with desired attributes, such as color, face shape, expression and view, or transferring the face shape, expression, or view learned from one person to another person.

Generative models have shown great promise in learning disentangled representations of images. The generative models used for unsupervised disentangling usually fall into two categories: the Generative Adversarial Net (GAN) framework [9,11,24,29,33] and the Variational Autoencoder (VAE) framework [18,22,28,31]. InfoGAN [6], a representative of the former family, is motivated by the principle of the maximization of the mutual information between the observations and a subset of latent vectors. However, its disentangling performance is sensitive to the choice of the prior and the number of latent vectors. The β-VAE [14], from the latter family, learns disentangled representations by utilizing a VAE objective with an extra KL penalty to encourage the latent distribution (variational posterior) to be close to the standard normal distribution, giving a more robust and stable solution for disentangling.

In contrast to the existing methods which use one latent vector to encode the factors of variation, our work introduces a deformable generator network that disentangles the appearance and geometric information from an image.

*The work was done while the author worked as a visiting scholar at UCLA.
into two independent latent vectors in an unsupervised manner. Motivated by the Active Appearance Models (AAM) [7,20] which uses a linear model for jointly capturing the appearance and shape variation in an image, the proposed model introduces two nonlinear generators to extract the appearance and geometry information separately. Unlike the AAM method [7,19,20] which requires hand-annotated facial landmarks for each training image, the proposed deformable generator model is purely unsupervised and learns from images alone.

2 Model and learning algorithm

This section provides the details of the model and the associated learning and inference algorithm.

2.1 Model

The proposed model contains two generator networks: one appearance generator and one geometric generator. The two generators are connected with a warping function to produce the final image. The warping function includes a geometric transformation operation for image coordinates and a differentiable interpolation operation. The refining operation is optional for improving the warping function.

The geometric transformation \( \Phi \) can be a rigid affine mapping, as is used in the spatial transformer networks [17], or a non-rigid deformable mapping, which is the case in our work. Specifically, the coordinates displacement \((dx, dy)\) (or the dense optical flow field) of each regular grid \((x, y)\) in the output warping image \(X\) are generated by our geometric generator \(F_g(Z^g; \theta_g)\). The point-wise transformation in this deformable mapping can be formulated in Figure[1]. Suppose an arbitrary image or video frame \(X \in \mathbb{R}^{D_x \times D_y \times 3}\) is generated with two independent latent vectors, \(Z^a \in \mathbb{R}^{d_a}\) which controls its appearance, and \(Z^g \in \mathbb{R}^{d_g}\) which controls its geometric information. Varying the geometric latent vector \(Z^g\) and fixing the latent vector \(Z^a\) of appearance, we can transform an object’s geometric information, such as rotating it with some angles and changing its shape. Varying the \(Z^a\) and fixing the \(Z^g\), we can change the identity or the category of the object, while keeping it within the same geometric status, such as the same viewing angle or the same shape. Thus, the appearance information and the geometric information are disentangled in the ideal situation.

The model can be expressed as

\[
X = F(Z^a, Z^g; \theta) = F_w(F_a(Z^a; \theta_a), F_g(Z^g; \theta_g)) + \epsilon
\]

where \(Z^a \sim N(0, I_{d_a}), Z^g \sim N(0, I_{d_g})\), and \(\epsilon \sim N(0, \sigma^2 I_D)\) are independent. \(F_w\) is the warping function, which employs the features generated by the geometric generator \(F_g(Z^g; \theta_g)\) to warp the geometry of the image from the appearance generator \(F_a(Z^a; \theta_a)\) to obtain the final output image \(X\).

2.2 Warping function

A warping function usually includes a geometric transformation operation for image coordinates and a differentiable interpolation (or resampling) operation. The geometric transformation describes the destination coordinates \((x, y)\) for every location \((u, v)\) in the source coordinates. The geometric operation only modifies the positions of pixels in an image without changing their colors and illumination. Therefore, the color and illumination information and the geometric information are naturally disentangled by the geometric generator and the appearance generator in the proposed model.

The geometric transformation \(\Phi\) can be a rigid affine mapping, as is used in the spatial transformer networks [17], or a non-rigid deformable mapping, which is the case in our work. Specifically, the coordinates displacement \((dx, dy)\) (or the dense optical flow field) of each regular grid \((x, y)\) in the output warping image \(X\) are generated by our geometric generator \(F_g(Z^g; \theta_g)\). The point-wise transformation in this deformable mapping can be formulated in Figure[1].
as

\[
\begin{pmatrix}
    u \\
    v
\end{pmatrix} = \Phi_{(Z^a, \theta_a)} \begin{pmatrix}
    x \\
    y
\end{pmatrix} = \begin{pmatrix}
    x + dx \\
    y + dy
\end{pmatrix}
\tag{2}
\]

where \((u, v)\) are the source coordinates of the image generated by the appearance generator \(F_a(Z^a; \theta_a)\).

Since the evaluated \((u, v)\) by Eq. (2) does not usually have integer coordinates, each pixel’s value of the output warping image \(X\) can be computed by a differentiable interpolation operation. Let \(X_o = F_a(Z^a; \theta_a)\) denotes the image generated by the appearance generator. The warping function \(F_w\) can be formulated as

\[
X(x, y) = F_w(X_a(x + dx, y + dy)),
\tag{3}
\]

where \(F_w\) is the differentiable interpolation function. In this study, we employ a differentiable bilinear interpolation of the form

\[
X(x, y) = \sum_{i} \sum_{j} X_a(i, j) M(1 - |u - i|) M(1 - |v - j|)
\tag{4}
\]

where \(M(\cdot) = \max(0, \cdot)\), and from Eq. (2), we have \(u = x + dx, v = y + dy\). The details of back-propagation through this bilinear interpolation can be found in [17].

The displacement \((dx, dy)\) is used in the deformable convolutional networks [8]. The computation of coordinates displacement \((dx, dy)\) is known as the optical flow estimation [4,10,15,16,30,32]. Our work is concerned with modeling and generating the optical flow, in addition to estimating the optical flow.

The displacement \((dx, dy)\) can be caused by the motion of the objects in the scene. It can also be caused by the change of viewpoint relative to 3D objects in the scene. Therefore, it is natural to incorporate motion and 3D models into the geometric generator where the change or variation of \(Z^g\) depends on the motion and 3D information.

2.3 Inference and learning

To learn this deformable generator model, we introduce a learning and inference algorithm for two latent vectors, without designing and learning extra inference networks. Our method is motivated by a maximum likelihood learning algorithm for generator networks [13]. Specifically, the proposed model can be trained by maximizing the log-likelihood on the training dataset \(\{X_i, i = 1, \ldots, N\}\),

\[
L(\theta) = \frac{1}{N} \sum_{i=1}^{N} \log p(X_i; \theta)
\]

\[
= \frac{1}{N} \sum_{i=1}^{N} \log \int p(X_i, Z^a_i, Z^g_i; \theta) dZ^a_i dZ^g_i.
\tag{5}
\]

The uncertainties in inferring \(Z^a_i\) and \(Z^g_i\) are taken into account by the above observed-data log-likelihood.

We can evaluate the gradient of \(L(\theta)\) according to the following well-known result which is related to the EM algorithm:

\[
\frac{\partial}{\partial \theta} \log p(X; \theta) = \frac{1}{p(X; \theta)} \frac{\partial}{\partial \theta} \log \int p(X, Z^a, Z^g) dZ^a dZ^g
\]

\[
= \mathbb{E}_{p(Z^a, Z^g | X; \theta)} \left[ \frac{\partial}{\partial \theta} \log p(X, Z^a, Z^g; \theta) \right].
\tag{6}
\]

Since the expectation in Eq. (6) is usually analytically intractable, we employ Langevin dynamics to draw samples from the posterior \(p(Z^a, Z^g | X; \theta)\) and compute the Monte Carlo average to obtain an approximation. For each observation \(X\), the latent vectors \(Z^a\) and \(Z^g\) can be sampled from \(p(Z^a, Z^g | X; \theta)\) alternately by Langevin dynamics: fixing \(Z^g\) and sampling \(Z^a\) from \(p(Z^a | X; Z^g, \theta) \propto p(X, Z^a, Z^g, \theta)\), then fixing \(Z^a\) and sampling \(Z^g\) from \(p(Z^g | X; Z^a, \theta) \propto p(X, Z^a, Z^g, \theta)\). The latent vectors are inferred and updated as follows:

\[
Z^a_{i+1} = Z^a_i + \delta \frac{\partial}{\partial Z^a} \log p(X, Z^a_i, Z^g_i, \theta) + \delta \mathcal{E}^a_i
\]

\[
Z^g_{i+1} = Z^g_i + \delta \frac{\partial}{\partial Z^g} \log p(X, Z^a_i, Z^g_i, \theta) + \delta \mathcal{E}^g_i
\tag{7}
\]

where \(t\) is the number of steps of the Langevin sampling, \(\mathcal{E}\) is standard Gaussian noise added to prevent the chain from becoming trapped in local modes, and \(\delta\) is the step size of Langevin dynamics. The log of the joint density in
Eq.\(7\) can be evaluated by

\[
\log p(X, Z^a; Z^g, \theta) = \log [p(Z^a)p(X|Z^a, Z^g, \theta)] \\
= -\frac{1}{2\sigma^2} \|X - F(Z^a, Z^g; \theta)\|^2 - \frac{1}{2} ||Z^a||^2 + C_1 \\
\log p(X, Z^g; Z^a, \theta) = \log [p(Z^g)p(X|Z^a, Z^g, \theta)] \\
= -\frac{1}{2\sigma^2} \|X - F(Z^a, Z^g; \theta)\|^2 - \frac{1}{2} ||Z^g||^2 + C_2 \\
\tag{8}
\]

where \(F\) and \(\sigma\) are defined in Eq.\(1\), and both \(C_1\) and \(C_2\) are constants. It can be shown that, given sufficient transition steps, the \(Z^a\) and \(Z^g\) obtained from this procedure follow their joint posterior distribution.

Obtaining independent samples of the posterior density in each training iteration is infeasible due to the high computational cost of the MCMC updates. In this paper, the MCMC transitions of both \(Z^a\) and \(Z^g\) start from the updated latent vectors from the previous learning iteration. The persistent updating results in a chain that is long enough to sample from the posterior distribution, and the warm initialization vastly reduces the computational burden of the MCMC updates. The convergence of stochastic gradient descent based on posterior MCMC has been studied by [34].

For each training example \(X_i\), we run the Langevin dynamics in Eq.\(7\) to get the corresponding posterior sample \(Z_i^a\) and \(Z_i^g\). The sample is then used for gradient computation in Eq.\(6\). More precisely, the parameter \(\theta\) is learned through Monte Carlo approximation:

\[
\frac{\partial}{\partial \theta} L(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \frac{\partial}{\partial \theta} \log p(X_i, Z_i^a, Z_i^g, \theta) \\
= \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\sigma^2} (X_i - F(Z_i^a, Z_i^g; \theta)) \frac{\partial}{\partial \theta} F(Z_i^a, Z_i^g; \theta). \tag{9}
\]

The whole algorithm iterates through two steps: (1) inferential step which infers the latent vectors through Langevin dynamics, and (2) learning step which learns the network parameters \(\theta\) by stochastic gradient descent. Gradient computations in both steps are powered by backpropagation. Algorithm 1 describes the details of the learning and inference algorithm.

\begin{algorithm}
\caption{Learning and inference algorithm}
\begin{algorithmic}
\Require\(\begin{array}{l}
(1) \text{training examples } \{X_i \in \mathbb{R}^{D_x \times D_y \times 3}, i = 1, \ldots, N\} \\
(2) \text{number of Langevin steps } l \\
(3) \text{number of learning iterations } T
\end{array}\)
\Ensure\(\begin{array}{l}
(1) \text{learned parameters } \theta \\
(2) \text{inferred latent vectors } \{Z_i^a, Z_i^g, i = 1, \ldots, N\}
\end{array}\)
\Repeat
\State \(t \leftarrow 0, \text{initialize } \theta\).
\State \(Z_i^a, Z_i^g, i = 1, \ldots, N\)
\Repeat\(t \leftarrow t + 1\)
\EndRepeat
\EndRepeat
\State \(1: \text{Let } t \leftarrow 0, \text{initialize } \theta\).
\State \(2: \text{Initialize }\{Z_i^a, Z_i^g, i = 1, \ldots, N\}\)
\Repeat
\State \(3: \text{Inference back-propagation: For each } i, \text{run } l \text{ steps of Langevin dynamics to alternatively sample } Z_i^a \text{ from } p(Z_i^a|X_i; Z_i^g, \theta), \text{while fixing } Z_i^g, \text{and sample } Z_i^g \text{ from } p(Z_i^g|X_i; Z_i^a, \theta), \text{while fixing } Z_i^a.\)
\State \(4: \text{Learning back-propagation: Update } \theta_{t+1} \leftarrow \theta_t + \eta_t L'(\theta_t), \text{with learning rate } \eta_t, \text{where } L'(\theta_t) \text{ is computed according to Eq.}(9).\)
\Until \(t = T\)
\end{algorithmic}
\end{algorithm}

2.4 Deformable Variational Auto-encoder

The proposed deformable generator scheme is general and agnostic to different models. In fact, our method can also be learned by VAE\cite{13} to obtain deformable variational auto-encoder, by utilizing extra inference network to infer \((Z^a, Z^g)\) through re-parametrization. Specifically, we learn another \(q(Z^a, Z^g|X; \phi)\) to approximate the intractable posterior \(p(Z^a, Z^g|X; \theta)\). The appearance and geometric latent vectors are assumed to be independent Gaussian in the approximated distribution, i.e., \(q(Z^a, Z^g|X; \phi) = q(Z^a|X; \phi)q(Z^g|X; \phi)\), where the means and variances are modeled by inference network with parameters \(\phi\). This deformable VAE model is a naturally extension of the proposed deformable generator framework developed. We show some preliminary results in Sec\[3.1.1\]. Notice that the proposed scheme can also be used in adversarial learning methods\cite{11}, by designing a separate discriminator network for shape and appearance. We leave it as our further work. In this work, we focus on
the current learning and inference algorithm for the sake of simplicity, so that we do not resort to extra networks.

3 Experiments

In this section, we first qualitatively demonstrate that our proposed deformable generator framework consistently disentangles the appearance and geometric information. We then analyze and evaluate the proposed model quantitatively. The deformable generator network structures and parameters are listed in the Appendix. We set the value of the interpolation parameter $\gamma$ to 10 in the experiments, i.e., we vary the components of the latent vectors within the range $[-\gamma, \gamma]$ when visualizing the effects of the components.

3.1 Qualitative experiments

3.1.1 Experiments on CelebA

We first train the deformable generator on the 10,000 images from CelebA benchmark dataset \[25\]. Some examples in CelebA are shown in Figure 2, which are processed by the OpenFace \[2\] and cropped to $64 \times 64$ pixels. To study the performance of the proposed method for disentangling the appearance and geometric information, we investigate the effect of different combinations of the geometric latent vector $Z^g$ and the appearance latent vector $Z^a$. (1) Set the geometric latent vector $Z^g$ to zero, and each time vary one dimension of the appearance variable $Z^a$ from $[-\gamma, \gamma]$ with a uniform step $\frac{2\gamma}{10}$, while holding the other dimensions of $Z^a$ at zero. Some typical generated images are shown in Figure 3. (2) Set $Z^a$ to be a fixed value, and each time vary one dimension of the geometric latent vector $Z^g$ from $[-\gamma, \gamma]$ with a uniform step $\frac{2\gamma}{10}$, while keeping the other dimensions of $Z^g$ at zero. Some representative generated results are shown in Figure 4. The full images corresponding to each dimensions of $Z^a$ and $Z^g$ are attached in the appendix.

Figure 2: Example training images from CelebA. The training set contains 10000 images from CelebA, and they are cropped to $64 \times 64$ pixels by the OpenFace. These faces have different colors, illuminations, identities, viewing angles, shapes, and expressions.

Figure 3: Each dimension of the appearance latent vector encodes appearance information such as color, illumination and gender. In the fist line, from left to right, the color of background varies from black to white, and the gender changes from a woman to a man. In the second line, the moustache of the man becomes thicker when the corresponding dimension of $Z^a$ approaches zero, and the hair of the woman becomes denser when the corresponding dimension of $Z^g$ increases. In the third line, from left to right, the skin color changes from dark to white. In the fourth line, from left to right, the illumination lighting changes from the left-side of the face to the right-side of the face.

As we can observe from Figure 3 (1) although the training faces from CelebA have different viewing angles, the appearance latent vector only encodes front-view information, and (2) each dimension of the appearance latent vector encodes appearance information such as color, illumination and identity. For example, in the fist line of Figure 3, from left to right, the color of background varies from black to white, and the identity of the face changes from a women to a man. In the second line of Figure 3, the moustache of the man becomes thicker when the value of the corresponding dimension of $Z^a$ decreases, and the
Figure 4: Each dimension of the geometric latent vector encodes fundamental geometric information such as shape and viewing angle. In the fist line, the shape of the face changes from fat to thin from left to the right. In the second line, the pose of the face varies from left to right. In the third line, from left to right, the vertical tilt of the face varies from downward to upward. In the fourth line, the face width changes from stretched to cramped.

hair of the woman becomes denser when the value of the corresponding dimension of $Z^a$ increases. In the third line, from left to right, the skin color varies from dark to white, and in the fourth line, from left to right, the illumination lighting changes from the left-side of the face to the right-side of the face.

From Figure 4, we have the following interesting observations. (1) The geometric latent vectors does not encode any appearance information. The color, illumination and identity are the same across these generated images. (2) Each dimension of the geometric latent vector encodes fundamental geometric information such as shape and viewing angle. For example, in the first line of Figure 4, the shape of the face changes from fat to thin from left to right; in the second line, the pose of the face varies from left to right; in the third line, from left to right, the tilt of the face varies from downward to upward; and in the fourth line, the expression changes from stretched to cramped.

The appearance and geometric information could also be effectively disentangled by the introduced deformable VAE. For the extra inference network, or encoder network, we use the mirror structure of our generator model in which we use convolution layers instead of convolution transpose layers. The generator network structure as well as other parameters are kept the same as the model learned by alternating back-propagation. Figures 5 and 6 show interpolation results following the same protocol described before. From the results in Figures 5 and 4 we find that

Figure 5: Appearance interpolation results by deformable VAE. Each dimension of the appearance latent vector encodes appearance information such as illumination, color, and gender. In the first line, from left to right, the illumination varies from bright to dark, and the gender changes from a woman to a man. In the second line, the color changes from blue grey to golden yellow. In the third line, from left to right, the illumination lighting changes from the right-side of the face to the left-side of the face. In the fourth line, from left to right, the size of sunglasses varies from large to small, finally without sunglasses. In the fifth line, the hair of the woman becomes denser when the corresponding dimension of $Z^a$ decreases, the moustache of the man becomes thicker when the corresponding dimension of $Z^a$ increases. In the sixth line, the hair of the woman becomes denser when the corresponding dimension of $Z^a$ decreases, the eyebrow of the man becomes denser when the corresponding dimension of $Z^a$ increases.

the appearance and geometric information of face images have been disentangled effectively. Therefore, we can apply the geometric warping (e.g. operations in Figure 4) learned by the geometric generator to all the canonical faces (e.g. generated faces in Figure 3) learned by the appearance generator. Figure 7 demonstrates the effect of applying geometric warping to the generated canonical faces in Figure 3. Comparing Figure 3 with Figure 7 we find that the rotation and shape warping operations do not modify the identity information of the canonical faces, which corroborates the disentangling power of the proposed deformable generator model.
Figure 6: Geometry interpolation results by deformable VAE. Each dimension of the geometric latent vector encodes fundamental geometric information such as shape and viewing angle. In the first line, the shape of the face changes from fat to thin from left to the right. In the second line, the pose of the face varies from left to right. In the third line, from left to right, the vertical tilt of the face varies from downward to upward. In the fourth line, the face expression changes from happiness to sadness. In the fifth line, from left to right, the eyes vary from close to open. In the sixth line, the eyes change from looking right to looking left.

Furthermore, we evaluate the disentangling ability of the proposed model by transferring and recombining geometric and appearance vectors from different faces. Specifically, we first feed 7 unseen images from CelebA into our deformable generator model to infer their appearance vectors $Z_a^1, Z_a^2, \ldots, Z_a^7$ and geometric vectors $Z_g^1, Z_g^2, \ldots, Z_g^7$ using the Langevin dynamics (with 300 steps) in Eq. (7). Then, we transfer and recombine the appearance and geometric vectors and use $\{Z_a^1, Z_g^2\}, \ldots, \{Z_a^7, Z_g^1\}$ to generate six new face images, as shown in the second row of Figure 8. We also transfer and recombine the appearance and geometric vectors and use $\{Z_a^2, Z_g^3\}, \ldots, \{Z_a^7, Z_g^3\}$ to generate another six new faces, as shown in the third row of Figure 8. From the 2nd to the 7th column, the images in the second row have the same appearance vector $Z_a^a$, but the geometric latent vectors $Z_g^a$ are swapped between each image pair. As we can observe from the second row of Figure 8: (1) the geometric information of the original images are swapped in the synthesized images, and (2) the inferred $Z_g^a$ can capture the view information of the unseen images.

Figure 7: Applying the (a) rotation warping and (b) shape warping operations learned by the geometric generator to the canonical faces generated by the appearance generator. Compared with Figure 8, only the pose information varies, and the identity information is kept in the process of warping.

Figure 8: Transferring and recombining geometric and appearance vectors. The first row shows the 7 unseen faces from CelebA. The second row shows the generated faces by transferring and recombining the first row’s 2th-7th faces’ geometric vectors with the first row’s 1th face’s appearance vector. The third row shows the generated faces by transferring and recombining the first row’s 2th-7th faces’ appearance vectors with the first row’s 1th face’s geometric vector.
The learned geometric information can be directly applied to the faces of animals such as cats and monkeys, as shown in Figure 9. The monkey and cat faces rotate from left to right when the rotation warping learned from human faces is applied. The shape of both the monkey and cat faces changes from fat to thin when the shape warping learned by the geometric generators is used.

3.1.2 Experiments on expression dataset

We next study the performance of the proposed deformable generator model on the face expression dataset CK+ [26]. Following the same experimental protocol as the last subsection, we can investigate the change produced by each dimension of the appearance latent vector (after setting the value of geometric latent vector to zero) and the geometric latent vector (after setting the appearance latent vector to a fixed value). The disentangled results are shown in Figure 10. The training faces from CK+ have labels of expressions, but we do not use any such labels in our unsupervised learning method. Although the dataset contains faces of different expressions, the learned appearance latent vector usually encodes a neutral expression. The geometric latent vector controls major variation in expression, but does not change the identity information.

To test whether appearance and geometric information are disentangled in the proposed model, we try to transfer the learned expression from CK+ to another face dataset, Multi-Pie [12], by fine-turning the appearance generator on the target dataset while fixing the parameters in the geometric generator. Figure 10(c) shows the result of transferring the expressions of 10(b) into the faces of Multi-Pie. The expressions from the gray faces of CK+ have been transferred into the color faces of Multi-Pie.

3.1.3 Experiment on CIFAR-10

We further test our model on the CIFAR-10 [21] dataset, which includes various object categories and has 50,000 training examples. We randomly sample $Z^a$ from $\mathcal{N}(0, \mathbf{I}_d)$. For $Z^g$, we interpolate one dimension from $-\gamma$ to $\gamma$ and fix the other dimensions to 0. Figure 11 shows interpolated examples generated by model learned from the car category. For each row, we use different $Z^a$ and interpolate the same dimension of $Z^g$. The results show that each dimension of $Z^g$ controls a specific geometric transformation.
13. As we can observe from the left and the center sub-
we plot covariance relationship between the latent vector
where 

To quantitatively study the covariance between each di-
view variations as follows:

Finally, we compute the covariance between each dimen-
dimension of the latent vectors (Zg, Za) and input images with geometric variation, we select images with

For each of the 5 viewing angles {−30°, −15°, 0°, 15°, 30°}, we feed 100 images into the

The left subfigure shows covariance with the geometric (or appearance) latent vectors and view variations for the face images from Multi-Pie.

Figure 12: Absolute value of covariance between each dimension of the geometric (or appearance) latent vectors and view variations for the face images from Multi-Pie. The left subfigure shows covariance with the geometric latent vector; the right subfigure shows covariance with the appearance latent vector.

3.2 Quantitative experiments
3.2.1 Covariance between the latent vectors and geo-
metric variation
To quantitatively study the covariance between each di-

where \( i \) denotes the \( i \)-th dimension of latent vector \( Z^g \) or 

Similarly, we construct a 5-dimensional vector \( \bar{Z}^a(i) \) under each dimension of \( Z^a \). We normalize the viewing angles vector \( \theta = [-30, -15, 0, 15, 30] \) to have unit norm.

Finally, we compute the covariance between each dimen-

\[
R^g_i = |\bar{Z}^g(i)^\top \theta|, \quad R^a_i = |\bar{Z}^a(i)^\top \theta|
\] (10)

where \( i \) denotes the \( i \)-th dimension of latent vector \( Z^g \) or 

and \( |\cdot| \) denotes the absolute value. We summarize the 

covariance responses \( R^g \) and \( R^a \) of the geometric and appearance latent vectors in Figure 12. As we can observe in Figure 12, the \( R^g \) tends to be much larger than \( R^a \).

Moreover, for the two largest \( R^g_i \) and the largest \( R^a_i \), we plot covariance relationship between the latent vector \( \bar{Z}^g(i) \) (or \( \bar{Z}^a(i) \)) and viewing angles vector \( \theta \) in Figure 13. As we can observe from the left and the center sub-
figures from Figure 13, the \( \bar{Z}^g(i) \) corresponding to the two largest \( R^g_i \) (\( R^g_{1}, R^g_{38} \)) have very strong negative and

positive covariance respectively with change in viewing angle. However, as shown in the right sub-figure, the \( \bar{Z}^a(i) \) corresponding to the largest \( R^a_i \) (\( R^a_{1} \)) does not have strong covariance with the change of viewing angle. We wish to point out that we should not expect \( Z^a \) to encode the identity exclusively and \( Z^g \) to encode the view exclusively, because different persons may have shape changes, and different views may have lighting or color changes.

Furthermore, we generate face images by varying the dimension of \( Z^g \) corresponding to the two largest covariance responses from values \([-\gamma, +\gamma]\) with a uniform step \( \frac{2\gamma}{19} \), while holding the other dimensions of \( Z^g \) to zero as we did in the subsection 4.1.1. Similarly, we generate face images by varying the dimension of \( Z^a \) corresponding to the largest covariance responses from values \([-\gamma, +\gamma]\) with a uniform step \( \frac{2\gamma}{19} \), while holding the other dimen-
The generated images are shown in Figure 13(b). We can make several important observations. (1) The variation in viewing angle in the first two rows is very obvious, and the magnitude of the change in view in the first row is larger than that in the second row. This is consistent with the fact that $R_{g5} > R_{g38}$ and with the observation that the slope in the left subfigure of Figure 13(a) is steeper than that of the center subfigure of Figure 13(a). (2) In the first row, the faces rotate from right to left and the covariance relationship in the left subfigure of Figure 13(a) is nearly perfect negative covariance. In the second row, the faces rotate from left to right and the covariance relationship in the center subfigure of Figure 13(a) is nearly perfect positive covariance. (3) It is difficult to find obvious variation in viewing angle in the third row. Therefore, these generated images further verify that the geometric generator of the proposed model mainly captures geometric variation, while the appearance generator is not sensitive to geometric variation.

### Methods

<table>
<thead>
<tr>
<th>Methods</th>
<th>VAE</th>
<th>ABP</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reconstruction Error</td>
<td>89.02</td>
<td>94.66</td>
<td>76.52</td>
</tr>
</tbody>
</table>

Table 1: The Mean Square Reconstruction Errors per image for unseen multi-view faces from the Multi-Pie dataset.

#### 3.2.2 Reconstruction error on unseen multi-view faces

Since the proposed deformable generator model can disentangle the appearance and geometric information from an image, we can transfer the geometric warping operation learned from one dataset into another dataset. Specifically, given 1000 front-view faces from the Multi-Pie dataset [12], we can fine-tune the appearance generator’s parameters while fixing the geometric generator’s parameters, which are learned from the CelebA dataset. Then we can reconstruct unseen images that have various viewpoints. In order to quantitatively evaluate the geometric knowledge transfer ability of our model, we compute the reconstruction error on 5000 unseen images from Multi-Pie for the views $\{-30^\circ, -15^\circ, 0^\circ, 15^\circ, 30^\circ\}$, with 1000 faces for each view. We compare the proposed model with the state-of-art generative models, such as VAE [5, 18] and ABP [13]. For fair comparison, we first train the VAE and ABP models with the same CelebA training set of 10,000 faces, and then fine-tune them on the 1000 front-view faces from the Multi-Pie dataset. The mean square reconstruction error per image for each method is shown in Table 1. As we can observe from Table 1, the proposed method obtains the lowest reconstruction error. Our model benefits from the transferred geometric knowledge learned from the CelebA dataset, while both the VAE and ABP models cannot efficiently learn or transfer purely geometric information.

#### 3.3 Balancing explaining-away competition

The proposed deformable generator model utilizes two generator networks to disentangle the appearance and geometric information from an image. Since the geometric generator only produces displacement for each pixel without modifying the pixel’s value, the color and illumination information and the geometric information are naturally disentangled by the proposed model’s specific structure.

Figure 13: (a) Covariance relationship between the mean latent vector $\bar{Z}_g(i)$ (or $\bar{Z}_a(i)$) and viewing angles vector $\theta$. We choose two dimensions of $Z_g$ ($Z_{g5}$ and $Z_{g38}$, left and center) with the largest covariance and one dimension of $Z_a$ with the largest covariance ($Z_{a25}$, right). (b) Images generated by varying the values of the three dimensions in (a) respectively, while fixing the values of other dimensions to be zero.
In order to properly disentangle the identity (or category) and the view (or geometry) information, the learning capacity between the appearance generator and geometric generator should be balanced. The appearance generator and the geometric generator cooperate with each other to generate the images. Meanwhile, they also compete against each other to explain away the training images. If the learning of the appearance generator outpaces that of the geometric generator, the appearance generator will encode most of the knowledge (including the view and shape information), while the geometric generator will only learn minor warping operations. On the other hand, if the geometric generator learns much faster than the appearance generator, the geometric generator will encode most of the knowledge (including the identity or category information, which should be encoded by the appearance network).

To control the tradeoff between the two generators, we introduce a balance parameter $\alpha$, which is defined as the ratio of the number of filters within each layer between the appearance and geometric generators. The balance parameter $\alpha$ should not be too large or too small. We set $\alpha$ to 0.625 in our experiments.

4 Conclusion

In this study, we propose a deformable generator model which aims to disentangle the appearance and geometric information of an image into two independent latent vectors $Z_a$ and $Z_g$. The learned geometric generator can be transferred to other datasets, or can be used for the purpose of data augmentation to produce more variations in the training data for better generalization.

In addition to the learning and inference algorithm adopted in this paper, the model can also be trained by VAE and GAN, as well as their generalizations such as $\beta$-VAE and info-GAN, which target disentanglement in general.

The model can be generalized to a dynamic one by adding transition models for the latent vectors. While the transition model for the appearance vector may generates dynamic textures of non-trackable motion, the transition model for the geometric vector may generate intuitive physics of trackable motion. The geometric generator can also be generalized to incorporate 3D information of rigid or non-rigid 3D objects.

In our work, the warping function based on coordinate displacements is hand designed. A refinement of this warping function in the form of a residual in addition to the warping function may be learned from the data. However, we tend to believe that the warping function itself or more importantly the notion of coordinate displacements may have to be a fundamentally innate part of a model for vision that may not be learned from the data.

Acknowledgment

This work was supported by DARPA SIMPLEX N66001-15-C-4035, ONR MURI N00014-16-1-2007, DARPA ARO W911NF-16-1-0579, DARPA N66001-17-2-4029, Natural Science Foundation of China No. 61703119, Natural Science Fund of Heilongjiang Province of China No. QC2017070 and Fundamental Research Funds for the Central Universities of China No. HEUCFM180405. We thank Mitchell K. Hill for his assistance with writing.

References


### A Deformable generator’ Network Structure and Parameters

<table>
<thead>
<tr>
<th>Layers</th>
<th>In-Out Shape</th>
<th>Conv-Kernel size</th>
<th>stride</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Z_a$</td>
<td>64 x 1</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>Fc1</td>
<td>4 x 4 x 80</td>
<td>64 x 1280</td>
<td></td>
</tr>
<tr>
<td>Deconv1</td>
<td>8 x 8 x 40</td>
<td>3 x 3 x 80 x 40</td>
<td>2</td>
</tr>
<tr>
<td>Deconv2</td>
<td>16 x 16 x 20</td>
<td>3 x 3 x 40 x 20</td>
<td>2</td>
</tr>
<tr>
<td>Deconv3</td>
<td>32 x 32 x 10</td>
<td>5 x 5 x 20 x 10</td>
<td>2</td>
</tr>
<tr>
<td>Out(Deconv4)</td>
<td>64 x 64 x 3</td>
<td>5 x 5 x 10 x 3</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Layers</th>
<th>In-Out Shape</th>
<th>Conv-Kernel size</th>
<th>stride</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Z_g$</td>
<td>64 x 1</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>Fc1</td>
<td>4 x 4 x 128</td>
<td>64 x 2048</td>
<td></td>
</tr>
<tr>
<td>Deconv1</td>
<td>8 x 8 x 64</td>
<td>3 x 3 x 128 x 64</td>
<td>2</td>
</tr>
<tr>
<td>Deconv2</td>
<td>16 x 16 x 32</td>
<td>3 x 3 x 64 x 32</td>
<td>2</td>
</tr>
<tr>
<td>Deconv3</td>
<td>32 x 32 x 16</td>
<td>5 x 5 x 32 x 16</td>
<td>2</td>
</tr>
<tr>
<td>Out(Deconv4)</td>
<td>64 x 64 x 2</td>
<td>5 x 5 x 16 x 2</td>
<td>2</td>
</tr>
</tbody>
</table>