Disentangling Factors of Variation by Mixing Them

Qiyang Hu\textsuperscript{1*}, Attila Szabó\textsuperscript{1*}, Tiziano Portenier\textsuperscript{1}, Matthias Zwicker\textsuperscript{2}, Paolo Favaro\textsuperscript{1}

\textsuperscript{1}University of Bern, Switzerland
\textsuperscript{2}University of Maryland, USA

\{hu, szabo, portenier, paolo.favaro\}@inf.unibe.ch
\textsuperscript{2}zwicker@cs.umd.edu

Abstract

We propose an unsupervised approach to learn image representations that consist of disentangled factors of variation. A factor of variation corresponds to an image attribute that can be discerned consistently across a set of images, such as the pose or color of objects. Our disentangled representation consists of a concatenation of feature chunks, each chunk representing a factor of variation. It supports applications such as transferring attributes from one image to another, by simply swapping feature chunks, and classification or retrieval based on one or several attributes, by considering a user specified subset of feature chunks. We learn our representation in an unsupervised manner, without any labeling or knowledge of the data domain, using an autoencoder architecture with two novel training objectives: first, we propose an invariance objective to encourage that encoding of each attribute, and decoding of each chunk, are invariant to changes in other attributes, and decoding of each chunk, are invariant to changes in other attributes and chunks, respectively; and second, we include a classification objective, which ensures that each chunk corresponds to a consistently discernible attribute in the represented image, hence avoiding the shortcut where chunks are ignored completely. We demonstrate the effectiveness of our approach on the MNIST, Sprites, and CelebA datasets.

1. Introduction

Deep learning techniques have led to highly successful natural image representations, some focusing on synthesis of detailed, high resolution images in photographic quality [15, 6], and others on disentangling image features into semantically meaningful properties [25, 22, 7], for example.

In this paper, we learn a disentangled image representation that separates the feature vector into multiple chunks, each chunk representing intuitively interpretable properties, or factors of variation, of the image. We propose a completely unsupervised approach that does not require any labeled data, such as pairs of images where only one factor of variation changes (different viewpoints, for example) [22, 1].

The basic assumption of our technique is that images can be represented by a set of factors of variation, each one corresponding to a semantically meaningful image attribute. In addition, each factor of variation can be encoded using its own feature vector, which we call a feature chunk. That is, images are simply represented as concatenations of feature chunks, in a given order. We obtain disentanglement of feature chunks by leveraging autoencoders, and as a key contribution of this paper, by developing a novel invariance objective. The goal of the invariance objective is that each attribute is encoded into a chunk invariant to changes in other attributes, and that each chunk is decoded into an attribute invariant to changes in other chunks. We implement this objective using a sequence of two feature swapping autoencoders that mix feature chunks.

The invariance objective using feature swapping on its own, however, does not guarantee that each feature chunk represents a meaningful factor of variation. Instead, the autoencoder could represent the image with a single chunk, and ignore all the others. This is called the shortcut problem [1]. We address the shortcut problem with a classification constraint, which forces each chunk to have a consistent, discernible effect on the generated image.

We demonstrate successful results of our approach on several datasets, where we obtain representations consisting of feature chunks that determine semantically meaningful image properties. In summary, we make the following contributions:

- A novel architecture to learn image representations of disentangled factors of variation in a completely unsupervised manner, where the representation consists of a concatenation of a fixed number of feature chunks. Our approach can learn several factors of variation simultaneously.
• A novel invariance objective to obtain disentanglement by encouraging invariant encoding and decoding of image attributes and feature chunks, respectively.

• A novel classification constraint to ensure that each feature chunk represents a consistent, discernible factor of variation of the represented image.

• An evaluation on the MNIST, Sprites, and CelebA datasets to demonstrate the effectiveness of our approach.

2. Related work

2.1. Autoencoders

Our architecture builds on an autoencoder [5, 13, 2], which is a neural network with two main components, an encoder and a decoder. The encoder is designed to extract a feature representation of the input (image), and the decoder translates the features back to the input. Different flavors of autoencoders have been taught to perform image restoration [29, 21, 4], or image transformation [12]. While basic autoencoders do not impose any constraints on the representation itself, variational autoencoders [17] add a generative probabilistic formulation, which forces the representation to follow a Gaussian distribution and allows sampling images by applying the decoder to a Gaussian noise vector. Thanks to their flexibility, autoencoders became ubiquitous tools in larger systems for domain adaptation [32, 16], or unsupervised feature learning [23]. Autoencoders are also used to learn feature disentangling [25, 22, 1]. In our work we also use them as feature extractors. Our contribution is a novel, unsupervised training method that ensures the separation of factors of variation into several feature chunks.

2.2. GANs

Generative Adversarial Networks (GANs) [10] were designed to learn to sample from data distributions that are given simply as a set of training samples from the desired distributions. They solve the task using two competing neural networks: a generator translates input noise vectors into fake data samples, while a discriminator tries to classify the fake and real samples correctly. In an ideal training scenario, the system converges to a state where the generator produces fake data samples following the same distribution as the training data, and the discriminator can only detect fake samples by chance. By enforcing that generated samples follow a certain distribution, GANs have been successful at synthesizing more realistic images [14], learning representation [24], sampling images from a specific domain [32], or ensuring image-feature pairs have the same distribution when computing one from another [9]. As the adversarial loss constrains the distribution of the generated data but not the individual data samples, it allows to reduce the need for data labeling. In particular, Shrivastava et al. [26] use GANs to transfer known attributes of synthetic, rendered examples to the domain of real images, thus creating virtually unlimited datasets for supervised training. In our work we use GANs to enforce that images look realistic when their attributes are transferred.

2.3. Disentangling and Independence

There are many fully supervised supervised methods [28, 30] that disentangle factors of variation. Arguably the simplest way for disentangling is to swap the feature representation in an autoencoder [25], then force the decoder to produce the result with the swapped attributes.

GANs and adversarial training have been leveraged to reduce the need for complete labeling of all factors of variation. Instead, these techniques allow the separation of some labeled factor of variation, without labeling any other attributes. For example, Mathieu et al. [22] apply adversarial training on the image domain, and Denton et al. [8] propose adversarial training on the feature domain to achieve this.

Recent work [1] describes the fundamental challenges of disentangling under such weak labeling. They prove the existence of the reference ambiguity, which means that only the labeled factor can be provably recovered without additional knowledge. In many cases in practice, however, the reference ambiguity does not arise. Another challenge is the shortcut problem, which means that all information is stored in one feature chunk, while the others are ignored. This can be provably solved as shown in [1]. Unsupervised training without prior knowledge makes the issues with the reference ambiguity (the interpretability of factors) or the shortcut problem more severe. Nonetheless, our method can recover multiple semantically meaningful factors in several datasets.

In some approaches, the physics of the image formation model is integrated into the network training, with factors like depth and camera pose [31] or albedo, surface normals and shading [27]. In the case of [27] the prior knowledge allows training without any labels.

By maximizing the mutual information between synthesized images and latent features, InfoGAN [7] makes the latent features interpretable as semantically meaningful attributes. InfoGAN is also completely unsupervised, as is our approach, but it does not include an encoding stage. In contrast to InfoGAN, we build on an autoencoder, which allows us to recover the disentangled representation from input images, and swap attributes between images. In addition, we use a novel classification constraint instead of the feature consistency approach in InfoGAN.

Two recent techniques, β-VAE [11] and DIP-VAE [18], build on variational autoencoders (VAEs) to disentangle interpretable factors in an unsupervised way, similar to our approach. They encourage the latent features to be in-
dependent by generalizing the KL-divergence term in the VAE objective, which measures the similarity between the prior and posterior distribution of the latent factors. Instead, we build on swapping autoencoders [25] and adversarial training [10]. We encourage disentanglement using an invariance objective, rather than trying to match an isotropic Gaussian prior.

3. Unsupervised Disentanglement of Factors of Variation

A representation of images where the factors of variations are disentangled can be exploited for various computer vision tasks. At the image level, it allows to transfer attributes from one image to another. At the feature level, this representation can be used for image retrieval and classification. To achieve this representation and to enable the applications at both the image and feature level, we leverage autoencoders. Here, an encoder transforms the input image $x$ to its feature representation $f = Enc(x)$, where $f = [f^1, f^2, \ldots, f^n]$ consists of multiple chunks $f^i \in \mathbb{R}^d$. The dimension of the full feature is therefore $n \times d$. In addition, a decoder transforms the feature representation back to the image via $Dec(f) = x$.

Our main objective is to learn a disentangled representation, where each feature chunk corresponds to an image attribute. For example, when the data $x$ are face images, chunk $f^1$ could represent the hair color, $f^2$ the gender and so on. With a disentangled representation, we can transfer attributes from one image to another simply by swapping the feature chunks. An image $x_3 = Dec([f^1_3, f^2_3, f^3_3, \ldots, f^n_3])$ could take the hair color from image $x_1$ and all the other attributes from $x_2$.

In our approach, we interpret disentanglement as invariance. In a disentangled representation, the encoding of each image attribute into its feature chunk should be invariant to transformations of any other image properties. Vice versa, the decoding of each chunk into its corresponding attribute should be invariant to changes of other chunks. In our example, if $x_1$ and $x_2$ have the same gender, we must have $f^2_1 = f^2_2$ irrespective of any other attribute. Hence, a disentangled representation is also useful for image retrieval, where we can search for nearest neighbors of a specified attribute. Invariance is also beneficial for classification, where a simple linear classifier is sufficient to classify each attribute based on its corresponding feature chunk. This observation inspired previous work [18] to quantify disentanglement performance using linear classifiers on the full features $f$.

In the following, we describe how we learn a disentangled representation from data without any additional knowledge (e.g., labels, data domain) by using swapping autoencoders. One of the main challenges in the design of the autoencoder and its training is that the encoder and the decoder could just make use of a single feature chunk (provided this is sufficient to represent the whole input image) and ignore the other chunks. We call this failure mode a shortcut taken by the autoencoder during training. We propose a novel invariance objective to obtain disentanglement, and a classification objective to avoid the shortcut problem.

3.1. Network Architecture

Our network architecture is shown in Figure 1. There are three main components: We enforce invariance using a sequence of two swapping autoencoders, and a discriminator; we avoid the shortcut problem using a classifier. They are all implemented as neural networks.

Swapping Autoencoders. We leverage a sequence of two swapping autoencoders to enforce invariance, ensuring that we encode each attribute into a feature chunk invariant to changes in other attributes, and that we decode each chunk similarly in an invariant manner into its attribute. More precisely, the sequence of two swapping autoencoders performs the following operations (Figure 1):

1. Sample two images $x_1$ and $x_2$ independently, and encode them into $f_1 = Enc(x_1)$ and $f_2 = Enc(x_2)$.

Figure 1: Overview of our architecture. The core component is a sequence of two swapping autoencoders (top). This implements our invariance objective, which encourages that the decoding of each feature chunk into an image attribute is invariant to a perturbation (swapping) in other chunks, and similarly, the encoding of each attribute into a chunk is invariant to a perturbation of other attributes. We include an adversarial loss to ensure the intermediate images obtained by perturbing some chunks is from our data distribution (bottom left). Finally, a classification objective avoids the shortcut problem, where chunks would be ignored completely. Components with the same name share weights.
2. Mixing: swap a randomly selected set of feature chunks specified by a mask $m = [m^1, m^2, \ldots, m^n]$, where $m^i \in \{0, 1\}$ and $1 = [1, 1, \ldots, 1] \in \mathbb{R}^d$ from $f_1$ into $f_2$. Swapping leads to a new feature $f_{12} = m \odot f_1 + (1 - m) \odot f_2$, where $\odot$ is the element-wise multiplication.

3. Decode a new image $x_3 = \text{Dec}(f_{12})$

4. Encode again, $f_3 = \text{Enc}(x_3)$

5. Swap the remaining original feature chunks, given by mask $1 - m$, from $f_1$ into $f_3$, that is, $f_{31} = m \odot f_3 + (1 - m) \odot f_1$.

6. Decode the final image $x_4 = \text{Dec}(f_{31})$, from the swapped features of $f_3$ and $f_1$.

Finally, we minimize the $L_1$ distance between $x_1$ and $x_4$, thus the loss function can be written as

$$L_S(\theta_{\text{enc}}, \theta_{\text{dec}}) = E_{x_1, x_2} \left[ \sum_m |x_4 - x_1| \right],$$

where we sum over all possible mask settings, and $\theta_{\text{enc}}$ and $\theta_{\text{dec}}$ are the encoder and decoder parameters respectively.

Intuitively, the key idea is that the cycle of decoding and re-encoding of the mixed feature vector $f_{12}$ should preserve the chunks from $f_1$ that were copied in $f_{12}$. In other words, these chunks from $f_1$ should be decoded into corresponding attributes of $x_3$. In addition, re-encoding into $f_3$ the intermediate image $x_3$ consisting of a mix of attributes from $x_1$ and attributes from $x_2$, should return the same feature chunks originally from $x_1$.

**Discriminator.** To ensure that the generated perturbed images $x_3$ are valid images according to the input data distribution, we impose an additional adversarial term, which is defined as

$$L_G(\theta_{\text{enc}}, \theta_{\text{dec}}, \theta_{\text{dsc}}) = \sum_m E_{x_1, x_2} \left[ \log(D_{\text{dsc}}(x_1)) + \log(1 - D_{\text{dsc}}(x_3)) \right],$$

where $\theta_{\text{dsc}}$ are the discriminator parameters. In the ideal case when the GAN objective reaches the global optimum, the distribution of fake images should match the real image distribution. With the invariance and adversarial loss, however, is still possible to encode all image attributes into one feature chunk and keep the rest constant. This solution optimizes both the invariance loss and the adversarial loss perfectly. As mentioned before, this is called the shortcut problem and we address it using an additional loss based on a classification task.

**Classifier.** The last component of our network takes three images as inputs. The input images $x_1$ and $x_2$ and the generated image $x_3$. It tries to decide for every chunk whether the composite image was generated using the feature from the first or the second input image. The formal loss function is

$$L_C(\theta_{\text{enc}}, \theta_{\text{dsc}}, \theta_{\text{cls}}) = E_{x_1, x_2} \left[ - \sum_m \sum_i m^i \log(y^i) + (1 - m^i) \log(y^i) \right],$$

where $\theta_{\text{cls}}$ are the classifier parameters, and its outputs are $\text{Cls}(x_1, x_2, x_3) = y = [y^1, y^2, \ldots, y^n]$. The classifier consists of $n$ binary classifiers, one for each chunk, that decide whether the composite image $x_3$ was generated using the corresponding chunk from the first image or the second. We use the cross entropy loss for classification, so the last layer of the classifier is a sigmoid. The classifier loss can only be minimized if there is a meaningful attribute encoded in every chunk. Hence, the shortcut problem cannot occur as it would be impossible to decide which chunks were used to create the composite image.

Finally, our overall objective consists of the weighted sum of the three components described above,

$$\min_{\theta_{\text{enc}}, \theta_{\text{dec}}, \theta_{\text{cls}}, \theta_{\text{dsc}}} \max_{\theta_{\text{dsc}}} \lambda_S L_S + \lambda_G L_G + \lambda_C L_C.$$  (4)

Note that during training, we randomly sample the masks $m$ instead of computing a sum over all possibilities for all image sample pairs (Equation 1, 2 and 3).

### 3.2. Implementation

We use a network architecture similar to DCGAN [24] for encoder, decoder, and discriminator. For the classifier, we use Alexnet with batch normalization after each convolutional layer, but we do not use any dropout. The image inputs of the classifier are concatenated along the RGB channels. We use equal weights $\lambda_S = \lambda_G = \lambda_C = 1$ for the swapping auto-encoder, GAN, and classifier for our experiments on the Mnist and Sprites datasets. For CelebA, we increase the weight of the swapping auto-encoder to $\lambda_S = 30$. In all experiments, we separate the feature after the last layer of the encoder into 8 chunks, where each chunk is expected to represent one attribute, with equal size for each of the eight chunks. We experimented with different chunk sizes, and found that the chunk size does not affect the results as long as it is reasonably large. Our results are obtained with chunk size 8 for Mnist, 64 for Sprites and 64 for CelebA. We observed that reducing the chunk size in CelebA leads to lower rendering quality. For CelebA, we also show experiments with a network architecture similar to BEGAN [3] for encoder, decoder and discriminator. The encoder has the same architecture as the encoder part of the discriminator in BEGAN, and we choose $\gamma = 0.5$. 
Method | body | skin | vest | hair | arm | leg | pose | average
--- | --- | --- | --- | --- | --- | --- | --- | ---
Random | 0.5 | 0.25 | 0.33 | 0.17 | 0.5 | 0.5 | 0.006 | 0.32
AE | 0.56 | 0.37 | 0.40 | 0.31 | 0.54 | 0.56 | 0.46 | 0.46
SWAP | 0.57 | 0.61 | 0.51 | 0.62 | 0.54 | 0.94 | **0.53** | 0.62
SWAP + C | 0.57 | 0.65 | 0.43 | **0.63** | 0.55 | 0.58 | 0.51 | 0.56
SWAP + G | **0.59** | 0.31 | 0.44 | 0.24 | 0.54 | 0.96 | 0.47 | 0.51
SWAP + C + G | 0.58 | **0.80** | **0.94** | 0.49 | **0.58** | **0.96** | 0.52 | **0.70**

Table 1: For a quantitative analysis of the ablation study we report the mean average precision performance of nearest neighbor classification on the Sprites dataset, which comes with labeled attributes. Each row contains different methods, while the columns show the classification performance of different attributes.

Figure 2: Comparison of different methods on Sprites. For all subfigures, we take one chunk from the topmost row, while all the others are from the leftmost column. Each column is rendered with a different chunk taken from the topmost row. The red frame indicates whether a chunk encodes an attribute. SWAP+G+C disentangles pose, torso color, hair color, and leg color (columns marked with red boxes, left to right).

Figure 3: Attribute transfer on the MNIST dataset by swapping individual chunks between pairs of source images, shown in the topmost row and leftmost column. To generate an image in column \(i\) and row \(j\), we take one chunk from the \(i\)-th image in the top row, and the other chunks from the \(j\)-th image in the leftmost column. In each subfigure, the swapped chunk corresponds to the attribute indicated in the caption of the subfigure.

4. Experiments

We experimented on three public datasets, the MNIST [19] handwritten digits, Sprites [25] animated figures, and CelebA [20] faces. We show qualitative results on all datasets and quantitative evaluations and ablation studies on Sprites and CelebA.

4.1. MNIST

The MNIST dataset consists of 60K handwritten digits for the training and 10K for the test set, given as grayscale images with a size of \(28 \times 28\) pixels. There are 10 different classes referring to the different digits. Other attributes like rotation angle or stroke width are not labeled. Our method can disentangle the labeled attribute as well as some non-labeled ones. Figure 3 shows visual attribute transfer for three factors, digit class, rotation angle, and stroke width. The three chunks were chosen by visually inspecting which chunk corresponds to which attribute. We can see that our method disentangles the factors efficiently on the image level.

4.2. Sprites

The sprites dataset has 672 synthetically rendered animated characters (sprites). The dataset is split into a training set with 500, validation set with 72, and test set with 100 sprites. Each sprite is rendered at 178 positions, thus the number of images is 120K in total. The dataset has
many labeled attributes: body shape, skin color, vest color, hairstyle, arm and leg color, and finally weapon type. The pose labels can be extracted from the frame number of the animations. This rich attribute labeling is ideal for testing the disentanglement of our algorithms.

We perform ablation studies on the components of our method. We compare the SWAP, SWAP+G, SWAP+C and SWAP+G+C methods, where SWAP denotes the swapping loss, G the adversarial loss and C the classifier loss in the objective. The qualitative results are shown in Figure 2. We transfer attributes encoded in single chunks from the top row to the leftmost column images. Each column of generated images corresponds to a different chunk taken from the image in the top row. We can see that SWAP already learned to disentangle 2 chunks. SWAP+G does not improve the disentangling, as the role of G is to make the images look more realistic. However the rendering quality with SWAP is already good. SWAP+C does not improve disentangling either, it rather creates artifacts in the rendering. The intuitive explanation is that adding C solves the shortcut problem in the sense that all chunks carry information, so the classifier can decide its origin. However the information is put into the artifacts, while the interpretable attributes are ignored. SWAP+G+C on the other hand improves the performance, as the artifacts are eliminated by G, so the shortcut problem can only be avoided by disentangling the factors. The method recovers 4 independent factors.

For quantitative analysis we perform nearest neighbor search using a chunk of the features and compute the mean average precision using an attribute as ground truth. We repeat the search for all chunk and attribute pairs, and for each attribute we choose the best performing chunk to represent it. We ignore the weapon type attribute in our evaluation, as it is only visible in a small subset of poses. We also compare our method to the vanilla autoencoder AE. The autoencoder only has one chunk, but its dimensionality is the same as the full feature of the other methods. In table 1 we compare the results of our methods. We can see that SWAP already has a good disentanglement performance compared to AE. As we expect adding only the G or C term does not help, but when we combine all three losses, we achieve substantial improvement.

Figure 4 visualizes attribute transfer for all chunks, similar as in Figure 3, using our top performing method SWAP+C+G. Our method could recover the leg and vest colors into single chunks, while the pose and arm color attribute pair is represented by one chunk. The skin color and hairstyle attributes are also entangled and represented by another chunk. There are 6 positions, where the sprites stand on a white bar. Even though this attribute is fully determined by the position, our method separates it to its own chunk.
Figure 5: Attribute transfer on the CelebA dataset with our method using (a) DCGAN and (b) BEGAN. For every subfigure, one chunk is taken from the topmost row and the rest from the leftmost column. Different subfigures show the role of different chunks. The captions indicate the attribute associated with the chunk.

### 4.3. CelebA

CelebA contains 200K color images of celebrity faces. The training, validation, and test sizes are 160K, 20K and 20K respectively. There are 40 labeled binary attributes indicating gender, hair color, facial hair and so on. We applied our method with both BEGAN and DCGAN architectures. Figure 5 shows the attribute transfer for each chunk. We can see that DCGAN exhibits more pronounced attribute transfer, while BEGAN tends to blur out the changes. Figure 6 shows the nearest neighbors of some query images in the dataset using DCGAN. We used the $L_2$ distance on the specified feature chunks to search for top matches. For
Figure 6: Image retrieval on CelebA of our method with DCGAN. Subfigures show the nearest neighbor matches for different feature chunks. For all subfigures, the first column contains the query images and subsequent columns contain the top matches using the $L_2$ distance. The caption indicates the discovered semantic meaning.

Table 2: The classification performance on CelebA. Each row contains different methods, while the columns show the different attributes (“eyebr.” is arched eyebrows and “attr.” is attractive).

Each chunk the top matches preserve a semantic attribute of the query image. Our method could recover five semantically meaningful attributes: brightness, glasses, hair color, hair style, and pose and smile. Notice that the attributes discovered with attribute transfer match the attributes in image retrieval. For brevity we only show those five chunks.

We performed quantitative tests on our learned features. We followed the evaluation technique based on the equivariant disentanglement property described in [18]. A feature representation is considered disentangled when the attributes can be classified using a simple linear classifier. In our special case when an attribute depends only on one chunk (a subspace), a linear classifier would perform well by setting the classifier weights w.r.t. the other chunks to zero. We train binary classifiers on the whole feature vector, each with a different labeled attribute as ground truth. The classifier prediction is $\text{sign}(w^T f + b)$, where the classifier weights are computed as

$$w = \frac{1}{\left| \{i : c_i > 0\} \right|} \sum_{i : c_i > 0} f_i - \frac{1}{\left| \{i : c_i < 0\} \right|} \sum_{i : c_i < 0} f_i,$$

(5)

where $c_i$ are the attribute labels. The bias term $b$ is set by minimizing the hinge loss. For a fair comparison we normalize the features by setting the variance for each coordinate to one, as in [18] the features are already normalized by the variational autoencoder. The results are shown in table 2. We can see that our network is competitive with the state of the art methods, $\beta$-VAE [11] and DIP-VAE [18]. The BEGAN architecture performs slightly worse than DCGAN, despite the superior rendering quality of the latter.

The qualitative and quantitative experiments showed that there is still room for improvement for both image and feature level disentanglement. We believe that using a better network architecture or training (especially for GAN) could boost the performance significantly.

5. Conclusions

We have introduced a novel method to disentangle factors of variation of a single set of images where no annotation is available. Our representation is computed through an autoencoder, which is trained by imposing constraints between the encoded features and the rendered images. We train the decoder to render realistic images by feeding features obtained by randomly mixing features from two images and by using adversarial training. Moreover, we force the autoencoder to make full use of the features by training it jointly with a classifier that determines how features have been mixed from an input image. We show that this technique successfully disentangles factors of variation in the MNIST, Sprites and CelebA datasets.
References

[1] Anonymous. Challenges in disentangling independent factors of variation. *International Conference on Learning Representations*, 2018. 1, 2

[2] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013. 2

[3] D. Berthelot, T. Schumm, and L. Metz. BEGAN: Boundary equilibrium generative adversarial networks. *arXiv*:1703.10717, 2017. 4

[4] S. A. Bigdeli, M. Jin, P. Favaro, and M. Zwicker. Deep mean-shift priors for image restoration. *arXiv*:1709.03749, 2017. 2

[5] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4):291–294, 1988. 2

[6] Q. Chen and V. Koltun. Photographic image synthesis with cascaded refinement networks. *arXiv*:1707.09405, 2017. 1

[7] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016. 1, 2

[8] E. Denton and V. Birodkar. Unsupervised learning of disentangled representations from video. *arXiv*:1705.10915, 2017. 2

[9] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *International Conference on Learning Representations*, 2017. 2

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2, 3

[11] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2016. 2, 8

[12] G. E. Hinton, A. Krizhevsky, and S. D. Wang. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*, pages 44–51. Springer, 2011. 2

[13] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. 2

[14] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2016. 2

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

[16] T. Kim, M. Cha, H. Kim, J. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv*:1703.05192, 2017. 2

[17] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 2

[18] A. Kumar, P. Sattigeri, and A. Balakrishnan. Variational Inference of Disentangled Latent Concepts from Unlabeled Observations. *arXiv*:1711.00848, 2017. 2, 3, 8

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[20] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 5

[21] X.-J. Mao, C. Shen, and Y.-B. Yang. Image restoration using convolutional auto-encoders with symmetric skip connections. *arXiv*:1606.08921, 2016. 2

[22] M. F. Mathieu, J. J. Zhao, J. Zhao, A. Ramesh, P. Sprechmann, and Y. LeCun. Disentangling factors of variation in deep representation using adversarial training. In *Advances in Neural Information Processing Systems*, pages 5041–5049, 2016. 1, 2

[23] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 2

[24] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv*:1511.06434, 2015. 2, 4

[25] S. E. Reed, Y. Zhang, Y. Zhang, and H. Lee. Deep visual analogy-making. In *Advances in Neural Information Processing Systems*, pages 1252–1260, 2015. 1, 2, 3, 5

[26] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, 2017. 2

[27] Z. Shu, E. Yumer, S. Hadap, K. Sunkavalli, E. Shechtman, and D. Samaras. Neural face editing with intrinsic image disentangling. In *CVPR*, 2017. 2

[28] L. Tran, X. Yin, and X. Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, 2017. 2

[29] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine Learning*, ACM, 2008. 2

[30] P. Xi, Y. Xiang, S. Kihyuk, M. Dimitris, and C. Mannmohan. Reconstruction for feature disentanglement in pose-invariant face recognition. *arXiv*:1702.03041, 2017. 2

[31] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 2

[32] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv*:1703.10593, 2017. 2