Unsupervised Controllable Generation with Self-Training

Grigorios G Chrysos[1]**, Jean Kossaifi**[1,2]**, Zhiding Yu**[2]**, Anima Anandkumar**[2,3]
Imperial College London[1], NVIDIA[2], California Institute of Technology[3]

Abstract

Recent generative adversarial networks (GANs) are able to generate impressive photo-realistic images. However, controllable generation with GANs remains a challenging research problem. Achieving controllable generation requires semantically interpretable and disentangled factors of variation. It is challenging to achieve this goal using simple fixed distributions such as Gaussian distribution. Instead, we propose an unsupervised framework to learn a distribution of latent codes that control the generator through self-training. Self-training provides an iterative feedback in the GAN training, from the discriminator to the generator, and progressively improves the proposal of the latent codes as training proceeds. The latent codes are sampled from a latent variable model that is learned in the feature space of the discriminator. We consider a normalized independent component analysis model and learn its parameters through tensor factorization of the higher-order moments. Our framework exhibits better disentanglement compared to other variants such as the variational autoencoder, and is able to discover semantically meaningful latent codes without any supervision. We demonstrate empirically on both cars and faces datasets that each group of elements in the learned code controls a mode of variation with a semantic meaning, e.g. pose or background change. We also demonstrate with quantitative metrics that our method generates better results compared to other approaches.

1   Introduction

Generative Adversarial Networks (GANs) [1] are the method of choice for synthesis, owing to their ability to generate impressive photo-realistic images. Yet, they fall short in a key aspect of generation in real-world applications, *controllability* – the ability to control the semantic of the generated images in an interpretable, deterministic manner. Controllability will enable on-demand synthesis of images, which has numerous applications, including data augmentation and image editing. Controllable generation relies on semantically interpretable disentangled factors of variation, i.e., factors that modify a single mode of variation, such as length or color of the hair.

However, the unsupervised nature of GANs hinders the development of controllable generation. For instance, given a generator of a standard GAN model trained on facial synthesis, it is not possible to directly control semantic attributes in a new synthesized instance, such as type/length/color of the hair, shape of the face, etc. Adding supervision means getting access to various (labelled) image attributes, which can be expensive. To reduce the amount of supervision required, Nie *et al.* [2] consider semi-supervised learning in StyleGANs and reveal that limited amount of supervision is sufficient for high quality generation. This assumes, nonetheless, that all the semantic attributes are labeled, which may not always be achievable. Instead, it is preferable to achieve controlled generation in a fully unsupervised manner.

Unsupervised disentanglement has been explored in the Variational Autoencoders (VAEs) [3, 4]. However, the generation quality of VAEs does not yet match the quality of GAN-synthesized

images. GAN methods have also been extended to achieve disentanglement of the factors of variation [5–7]. However, these unsupervised GAN and VAE approaches have two major drawbacks: i) the disentangled factors are not guaranteed to be interpretable and ii) such models suffer from non-identifiability [8], meaning that different runs can produce different factors.

Consequently, in this work, we propose ST-GAN, the first fully unsupervised approach to controllable generation with GANs through self-training. Specifically, **we make the following contributions**:

- We propose a novel self-training procedure to discover disentangled and semantically interpretable latent codes driving the generation. The self training feedback loop allows for iterative refinement of the factor codes. We design a framework that encourages the model to produce interpretable factor codes that can faithfully control the generator.

- Instead of sampling from a fixed distribution probability distribution (e.g. Gaussian), we use a flexible latent variable model. Specifically, we employ a normalized independent component analysis models. To learn its parameters, we apply tensor factorization to the higher-order cumulants of the representation learnt in the feature space discriminator. We also experiment with a variational autoencoder as an alternative latent variable model.

- We empirically demonstrate that our approach results in a controllable GAN able to learn the hidden codes in a fully unsupervised manner. We show on two different domains, cars and faces, that the discovered codes are disentangled and semantically interpretable, allowing to control the variations in the synthesized images.

- We propose two quantitative metrics for measuring the semantic changes by modifying a single element of the factor codes. Using them, we quantitatively demonstrate significant improvements with our model over the baselines.Our experiments exhibit how our framework can progressively improve the learning of latent codes as the training proceeds. We also establish the importance of each block of our model through extensive ablation studies.

2 Controlling generation and discovering the latent codes with self-training

Our method discovers the disentangled, semantically meaningful latent codes driving the generation in a fully unsupervised manner. This relies on augmenting a GAN structure towards controllable generation. Specifically, we make the following changes: i) instead of sampling from a fixed distribution, the generator takes as input latent codes from a normalized independent component analysis model, ii) a self-training scheme is proposed to discover the latent codes, iii) a hierarchical structure is used in the generator. Each contribution is analyzed below, while in Fig. 1 an abstract schematic of the framework is illustrated. We first review generative adversarial networks, before motivating and describing our proposed method.

Notation: Vectors (scalars) are denoted with boldface (plain) letters, e.g. $\boldsymbol{x}(x)$. The outer product is symbolized with $\otimes$. We summarize in Table 1 the notation used in this work.

Table 1: Table of notations

Symbol	Definition
$\boldsymbol{h}$	latent code used as input to the generator
$\boldsymbol{x}$	(real/fake) image
R	Rank of the decomposition
$\boldsymbol{\lambda}, \boldsymbol{a}_j$	Parameters of the tensor decomposition
$\boldsymbol{M_1} + w_1 \boldsymbol{M_1}$	Terms for the second order cumulant
$\boldsymbol{T_1} + \sum_{k=1}^{5} w_k \boldsymbol{T_k}$	Terms for the third order cumulant
v	Penultimate layer of the discriminator
Symm	Operator symmetrizing a tensor
$\Phi(\boldsymbol{x})$	Representation extracted for the latent variable model
γ	Regularization hyper-parameter

2.1 Generative Adversarial Networks

Generative Adversarial Networks   A GAN consists of a generator G and a discriminator D engaging in a zero sum game. The goal of the generator is to to model the target distribution $\mathbb{P}_{data}$, while the discriminator aims at discerning the samples synthesized by the generator and the real samples from target (ground-truth) distribution.

The generator samples a latent code from a fixed probability distribution $\mathbb{P}_z$ (typically Gaussian) and maps it to an image $G(\boldsymbol{z})$. The discriminator receives both images synthesized by the generator $G(\boldsymbol{z})$ and samples from the real distribution $\mathbb{P}_{data}$ and tries to distinguish them. The objective function is:

$$\mathcal{L}_{gan} = \mathbb{E}_{\boldsymbol{x} \sim \mathbb{P}_{data}} \Big[\log D(\boldsymbol{x}) \Big] + \mathbb{E}_{\boldsymbol{z} \sim \mathbb{P}_z} \Big[\log(1 - D(G(\boldsymbol{z}))) \Big]. \tag{1}$$

This loss is optimized in an alternating manner as $\min_G \max_D \mathcal{L}_{gan}$.

Style-GAN   proposes hierachical injections in the generator [9]. The injections, i.e. Hadamard products with the inputs, are performed after each layer. In practice, our generator differentiates from Style-GAN injections, as we partition the latent codes (i.e. input to the generator). Specifically, we use s partitions, where s denotes the total number of injections in the generator. That is, for a latent code $\boldsymbol{h} \in \mathbb{R}^d$, we use the elements $[h_1, \ldots, h_{\frac{d}{s}}]$ in the first injection, the next $\frac{d}{s}$ elements in the second and so on. This injection captures higher-order correlations [10, 11]. Therefore, by partitioning the latent codes we capture the correlations of specific elements in each injection.

Semi-Supervised Style-GAN   augments the training procedure of Style-GAN with two additional losses: a Mixup loss and a consistency loss [2]. The consistency loss is expressed as $\mathcal{L}_c = \big\| \boldsymbol{h} - \boldsymbol{W}^T v(G(\boldsymbol{h})) \big\|_2^2$, where v is the representation from the penultimate layer of the discriminator, $\boldsymbol{W}$ a learnable dense layer and $\boldsymbol{h}$ is the input to the generator. The consistency loss constrains the original and the reconstructed latent codes (i.e., $\boldsymbol{h}$) to be close.

The Mixup loss reinforces smoothness in the latent code space. Given a pair of real and fake images $(\boldsymbol{I}_r, \boldsymbol{I}_f)$ and their corresponding latent codes $(\boldsymbol{h}_r, \boldsymbol{h}_f)$, we interpolate between the images and the latent codes to obtain $(\boldsymbol{I}_s, \boldsymbol{h}_s)$. The Mixup loss is then $\mathcal{L}_s = \big\| \boldsymbol{h}_s - \boldsymbol{W}^T v(\boldsymbol{I}_s) \big\|_2^2$.

In this paper, we build on top of both Style-GAN and its semi-supervised variant and incorporate both losses when training our proposed model.

2.2 Driving generation with a latent variable model

In GAN, the generator takes as input a latent code, sampled from a fixed probability distribution. Typically, a Gaussian or a uniform distribution is selected. Replacing this fixed distribution with
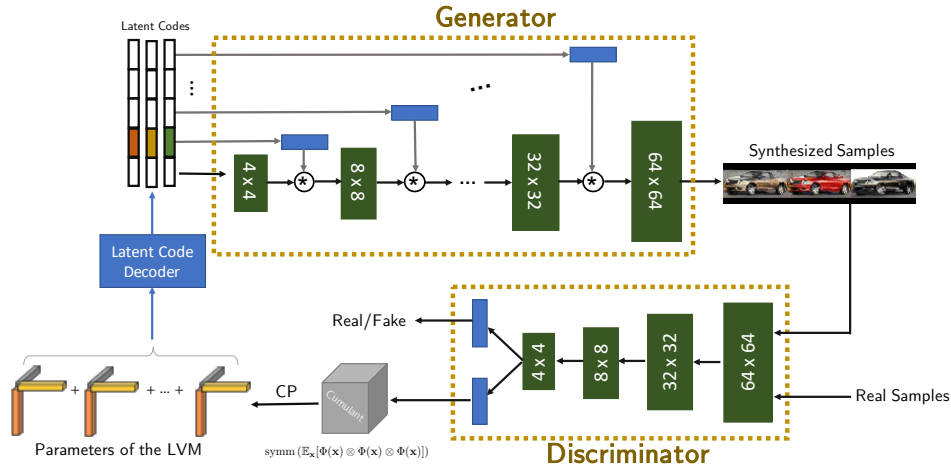

Figure 1: Overview of our proposed ST-GAN.

3

distributions with learnable parameters has been recently explored in the literature, both in the context of GAN [12] and other generative models [13, 14]. Simple fixed distributions, e.g. Gaussian, do not encode any semantic meaning such as 3D pose of the object.

Motivated by such works, we replace sampling the latent code from a fixed distribution with latent codes. The latent codes are sampled from a latent variable model. This underlying model is data-driven and evolves over the training to better reflect the variation of the specific data. In practice, we use a variant of independent component analysis, which produces latent codes $\boldsymbol{h} \sim \mathbb{P}_l$ used by the generator. The independent component analysis [15] mixes linearly independent signals and then corrupts them with Gaussian noise. However, in this work we normalize the hidden independent signals in the range of $[-1, 1]$. A variant of this model with discrete observations is known as topic model, but we use continuous version and term it as normalized independent component analysis. If we denote the latent vector with $\boldsymbol{h}$, the Gaussian noise with $\boldsymbol{\epsilon}$ and the mixing matrix with $\boldsymbol{M}$, the corresponding observation is $\boldsymbol{y} = \boldsymbol{M}\boldsymbol{h} + \boldsymbol{\epsilon}$. As an ablation, we also explore using a VAE.

2.3 Learning the latent variable model with self-training

To obtain the latent codes $\boldsymbol{h}$, we use a self-training scheme with a latent variable model (a flexible normalized independent component analysis model) in the feature space of the discriminator. We learn its parameters through tensor factorization of the higher-order cumulant. We also showcase in ablation that other latent variable models, such as VAEs can also be used as drop-in replacements. During inference, the generator samples latent codes $\boldsymbol{h} \sim \mathbb{P}_l$ from the learnable distribution $\mathbb{P}_l$ and generates samples $G(\boldsymbol{h})$.

Extracting the representation   We learn the latent variable model (i.e., ICA) in the feature space of the discriminator. Specifically, for an input image $\boldsymbol{x}$, the corresponding features $v(\boldsymbol{x})$ are extracted, where $v(.)$ corresponds to the penultimate layer of the discriminator. The representation for the latent variable model is computed through a linear layer with learnable vector of parameters $\boldsymbol{W}$. The final extracted representation is then $\Phi(\boldsymbol{x}) = \boldsymbol{W}^T v(\boldsymbol{x})$. The mixing matrix is learned through tensor factorization. Specifically, we form the higher-order moments of the features $\Phi(\boldsymbol{x})$, and learn their low-rank factorized form.

Learning the parameters of the latent normalized independent component analysis model through tensor factorization   We assume a normalized independent component analysis model, which maps from the extracted features to untangled latent codes. We propose to learn the parameters of that model by decomposing the higher-order moments. By analogy, in the analytical case, we know it is possible to form the moments of the model. From these, the symmetric cumulant tensor (e.g. $\mathrm{Symm}\left(\mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x}) \otimes \Phi(\boldsymbol{x}) \otimes \Phi(\boldsymbol{x})]\right)$ for the third order cumulant) can be obtained by subtracting cross-order terms. Given a specific latent variable model, this symmetric tensor – or cumulant– can be obtained through a closed-form formula. Applying low-rank decompositions to it allows the recovery of the parameters of the model [16].

In this paper, all the components are learned end-to-end, including the factors of the decomposition. Instead of exactly forming the cumulant tensor, following [17], we propose to also learn the weights of each of the cross-terms. Specifically, for the second order, this results in $\boldsymbol{M_1} + w_1\boldsymbol{M_2}$, where $\boldsymbol{M_1} = \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x}) \otimes \Phi(\boldsymbol{x})]$ and $\boldsymbol{M_2} = \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x})] \otimes \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x})]$. Similarly, for the third order term, we form the term $\boldsymbol{T_1} + w_2\boldsymbol{T_2} + w_3\boldsymbol{T_3} + w_4\boldsymbol{T_4} + w_5\boldsymbol{T_5}$, with:

$$\boldsymbol{T_1} = \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x}) \otimes \Phi(\boldsymbol{x}) \otimes \Phi(\boldsymbol{x})], \qquad \boldsymbol{T_2} = \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x}) \otimes \Phi(\boldsymbol{x})] \otimes \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x})]$$
$$\boldsymbol{T_3} = \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x}) \otimes \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x})] \otimes \Phi(\boldsymbol{x})], \qquad \boldsymbol{T_4} = \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x})] \otimes \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x}) \otimes \Phi(\boldsymbol{x})]$$
$$\boldsymbol{T_5} = \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x})] \otimes \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x})] \otimes \mathbb{E}_{\boldsymbol{x}}[\Phi(\boldsymbol{x})]$$

We then factorize the resulting higher-order moments computed from the features $\Phi(x)$ from the discriminator. We assume that the cumulants, $\boldsymbol{M_1} + w_1\boldsymbol{M_2}$ and $\boldsymbol{T_1} + \sum_{k=1}^{5}\boldsymbol{T_k}$ admit a rank–R low-rank representation. In other words, we express the (symmetric) cumulants as a weighted sum of R rank–1 tensors. We learn both the weights of the sum (collected in a vector $\boldsymbol{\lambda}$) and the factors $\boldsymbol{a}_1, \cdots, \boldsymbol{a}_R$ of the decomposition. Specifically, we minimize the following loss function $\mathcal{L}_l$:

$$\underset{w_k, \boldsymbol{\lambda}, \boldsymbol{a}_j}{\arg\max} \, \mathcal{L}_l = \left\| \boldsymbol{M_1} + w_1\boldsymbol{M_2} - \sum_{j=1}^{R} \lambda_j \boldsymbol{a}_j \otimes \boldsymbol{a}_j \right\|_F^2 + \left\| \boldsymbol{T_1} + \sum_{k=2}^{5} w_k \boldsymbol{T_k} - \sum_{j=1}^{R} \lambda_j \boldsymbol{a}_j \otimes \boldsymbol{a}_j \otimes \boldsymbol{a}_j \right\|_F^2$$

4

The learned parameters $\boldsymbol{\lambda}, \boldsymbol{a}_j$ form the mixing matrix $\boldsymbol{M}$ for the normalized independent component analysis model. Specifically, the jth row corresponds to the jth factor, i.e. $M_{j,:} = \lambda_j \boldsymbol{a}_j$. Then, for a sample $\boldsymbol{x} \sim \mathbb{P}_{data}$, we obtain the latent code as $\boldsymbol{h} = \boldsymbol{M}^\dagger(\boldsymbol{x} - \epsilon)$ where $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2 \boldsymbol{I})$ and $\boldsymbol{M}^\dagger$ denotes $\boldsymbol{M}$'s pseudo-inverse.

To further encourage the disentanglement of the factors, we add an orthogonality regularization term in the loss, i.e. $\mathcal{L}_l = \mathcal{L}_l + \gamma_o \sum_{j=1}^{R} \left\| \boldsymbol{a}_j^T \boldsymbol{a}_j - \mathbb{I} \right\|_F^2$ where $\mathbb{I}$ is the identity matrix and γ_o is a hyper-parameter.

Ablation: Variational Autoencoder   As an ablation, we investigate replacing the higher-order factorization framework with VAEs, i.e., a VAE is learned as an alternative latent variable model. Similarly to the normalized independent component analysis model, we learn the VAE on the feature space of the discriminator.

Specifically, we maximize the ELBO of the distribution in the feature space of the discriminator: $\mathcal{L}_l = \mathbb{E}_{q(\boldsymbol{h}|\Phi(\boldsymbol{x}))} \left[\log p(\Phi(\boldsymbol{x})|\boldsymbol{h}) \right] - D_{KL} \left(q(\boldsymbol{h}|\Phi(\boldsymbol{x})) \,\|\, p(\boldsymbol{z}) \right)$, where $\boldsymbol{h}$ is the latent csode, $q(\boldsymbol{h}|\Phi(\boldsymbol{x}))$ is the posterior distribution and D_{KL} computes the Kullback-Leibler divergence between two distributions.

Training the model   The self-learning technique of our framework is sensitive to initialization; here, we elaborate on the details of the objective function and the training procedure. For the first 200 iterations, we use a 'warm-up' of the weights by training a using only the GAN loss. The latent codes in the input of the generator are sampled from a Gaussian. Sequentially, the latent variable model parameters are inserted and the corresponding $\mathcal{L}_l$ loss is added. At $2,000$ iterations, the self-training scheme is added along with the remaining loss terms.

Our preliminary experiments demonstrated that a soft transition from the prior distribution to the latent distribution is beneficial. To that end, we use a soft transition with an annealing κ parameter, i.e. $\boldsymbol{h} = \kappa \cdot \boldsymbol{h}_l + (1 - \kappa) \cdot \boldsymbol{h}_p$ where κ starts from 0 and transitions to 1 after $8,000$ iterations. The symbol $\boldsymbol{h}_l$ denotes a sample from the latent variable model, while $\boldsymbol{h}_p$ is a sample from the prior distribution used in the beginning of the training.

To further induce disentanglement, we add an additional "masking loss". The masking loss $\mathcal{L}_m$ encourages each element of the latent code to change one attribute. To achieve that, we synthesize a pair of images with a predictable changes in their latent codes and try to predict what that change was. Given a latent code $\boldsymbol{h}$, we duplicate it into $\hat{\boldsymbol{h}}$ and sequentially perturb each element of the latent code with uniform noise. The masking loss then tries to predict which element was modified from the $\{\Phi(G(\boldsymbol{h})), \Phi(G(\hat{\boldsymbol{h}}))\}$ embeddings.

The complete loss function used is $\mathcal{L} = \mathcal{L}_{gan} + \gamma_l \mathcal{L}_l + \gamma_s \mathcal{L}_s + \gamma_c \mathcal{L}_c + \gamma_m \mathcal{L}_m$, where $\gamma_l, \gamma_s, \gamma_c$ and γ_m are regularization hyper-parameters.

Algorithm 1: GAN iteration.

1: **function** RUN_ITERATION($\mathbb{P}_{data}, steps\_D$)
2:     **for** i=1:$steps\_D$ **do**
3:         $\boldsymbol{I}_r \sim \mathbb{P}_{data}$
4:         $\boldsymbol{h} \leftarrow LVM(\Phi(\boldsymbol{I}_r))$
5:         $\boldsymbol{I}_f \leftarrow G(\boldsymbol{h})$
6:         update_$D(D(\boldsymbol{I}_r, \boldsymbol{I}_f), \mathcal{L}_{LM})$
      end
7:     # *Update the generator*
8:     $\boldsymbol{h} \leftarrow LVM(\Phi(\boldsymbol{I}_r))$
9:     update_$G(D(G(\boldsymbol{h})), \mathcal{L}_{LM})$
10:   return G, D
11: **end function**

---

3 Experiments

In this section, we describe the experimental setup and the comparisons conducted with the proposed framework. We utilize both the popular CelebA [18] and Cars dataset for our experiments[1]. CelebA contains $202,000$ images of faces; we use $160,000$ images for training, while the Cars dataset includes $16,000$ images; we use the $12,000$ for training. All the images are resized to 64×64.

Metrics: The well-established Frechet Inception Distance (FID) metric [20] is chosen for the generation quality[2]. The metrics proposed for unsupervised disentanglement require an auxiliary encoder to be trained [3,4,22], which is not available in our case. In addition, using direct ground-truth labels is discouraged [8]. We opt to report auxiliary semantic metrics that demonstrate the changes affected by each element of the latent code. Specifically, to achieve controllable generation each element of the factor code should modify a factor of variation of the data and be interpretable. To that end, we utilize the following metric: we sample i) an element to perturb and ii) a perturbation in the range $(-1, 1)$. We add the perturbation, generate the two images and then compare them. We repeat this procedure for each element of the latent code for 10 perturbations. To measure the difference between each pair of images we use both the standard mean absolute error (MAE) and the LPIPS [23] metric that correlates with the perceptual changes. A higher value in both, means that the single element of the latent code has made a larger (perceptual) change in the image.

Implementation details: Our implementation is based on the GAN architecture of Miyato *et al.* [24]. That is, both the generator and the discriminator include residual blocks, while the rest hyper-parameters (i.e. optimizer, hinge loss, learning rate) remain unchanged. The injections in the generator follow the implementation of [10]. The models are implemented using PyTorch [21] and TensorLy for all tensor methods [25]. We used $\gamma_l = 1, \gamma_s = 0.1, \gamma_c = 0.1$. The hyper-parameter γ_m is augmented during the training; in iteration $2,000$ it starts as $\gamma_m = 1$ and in iteration $10,000$ it takes the value $\gamma_m = 100$. The dimensionality of the latent code is $d = 50$, i.e., $\boldsymbol{h} \in \mathbb{R}^{50}$. We apply an elementwise normalization by the max element in the latent codes before feeding them ins the generator.

3.1 Ablation studies

We investigate the different losses as well as the inductive bias of the architecture selected. Specifically, we will denote the model *ST-GAN-$\mathcal{L}_m$* when it does not include the loss $\mathcal{L}_m$. Similarly for the other losses. The rest of the hyper-parameters are not tuned again, but remain the same. In Table 2, the quantitative results illustrate that our method performs similarly when removing one loss at a time. However, the final model (with all the losses) outperforms all the variants.

Table 2: Ablation study on the losses.

Method	CelebA			Cars		
	FID ($\downarrow$)	**MAE ($\uparrow$)**	**LPIPS ($\uparrow$)**	**FID ($\downarrow$)**	**MAE ($\uparrow$)**	**LPIPS ($\uparrow$)**
ST-GAN	**25.21**	**0.265**	**0.040**	**12.96**	0.271	0.057
ST-GAN-$\mathcal{L}_s$	32.40	0.249	0.038	35.23	0.279	0.062
ST-GAN-$\mathcal{L}_c$	33.90	0.247	0.034	16.66	0.258	0.060
ST-GAN-No ortho	30.85	0.250	0.041	14.81	**0.314**	**0.069**
ST-GAN-$\mathcal{L}_m$	27.47	0.237	0.036	13.47	0.292	0.060

3.2 Comparison with other models

Two baseline architectures are considered: i) SNGAN [24] (referred to as *GAN* henceforth), ii) VAE with convolutional encoder and decoder. For the SNGAN we consider two additional variants: a) one where we include injections in the generator [9] (referred as *GAN-Inj*), b) one with a similar hierarchical generator as in our model (*GAN-H*). The goal is to assess whether the controllable

---

[1]CelebA dataset reportedly includes gender and racial biases [19], thus we encourage the development of better datasets to address the bias.

[2]The features from the pretrained Inception network of Paszke *et al.* [21] are used.

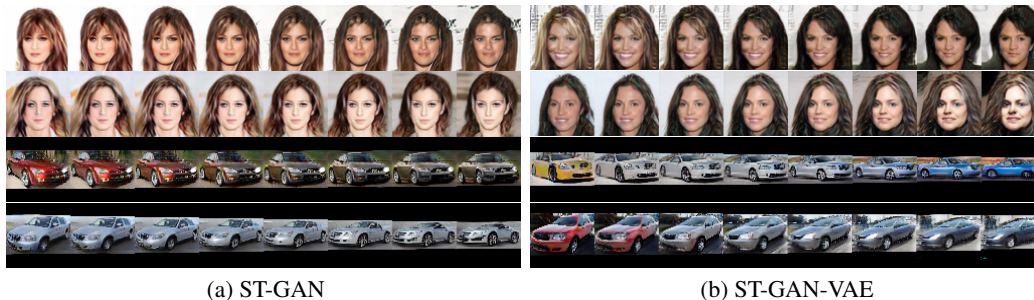| (a) ST-GAN | (b) ST-GAN-VAE |

Figure 2: Linear interpolation from a source to a target latent code. The synthesized images corresponding to each code are visualized. The linear interpolation in the latent space of ST-GAN illustrates how the proposed framework can vary all the modes of variation jointly for making nonlinear changes in the image space.

generation is caused by these inductive biases alone. Our model is denoted as 'ST-GAN', while the ablation with the VAE model in the latent space as 'ST-GAN-VAE'.

The quantitative results in Table 3 exhibit that our method outperforms all the baselines in both the MAE and the LPIPS metric. Intuitively, this means that each element of the factor code makes larger changes to the image than the baselines. To illustrate the differences, we perform two visualizations: a) linear interpolation in the latent space, b) linear interpolation in one element of the latent code at a time. The former visualization in Fig. 2 simply confirms that changing the latent code linearly results in realistic nonlinear changes in the image. In Fig. 3, each row depicts linear interpolation in a single element of the latent code. The rest elements remain the same. We notice how our model ('ST-GAN') changes the pose or the color in any object by using a single element. On the contrary, as we show in the supplementary, all the baseline models fail to make such large changes by changing a single element of their latent code.

Table 3: Performance of GAN models in terms of image quality (FID) and controllability (LPIPS, MAE). ST-GAN consistently provides better controllability as indicated from the reported metrics of LPIPS and MAE.

Method	CelebA			Cars		
	FID ($\downarrow$)	MAE ($\uparrow$)	LPIPS ($\uparrow$)	FID ($\downarrow$)	MAE ($\uparrow$)	LPIPS ($\uparrow$)
GAN [24]	23.21	0.169	0.024	23.96	0.243	**0.061**
GAN-Inj	**20.07**	0.179	0.026	15.06	0.240	0.049
GAN-H	21.17	0.178	0.024	**12.13**	0.218	0.042
VAE	91.20	0.003	0.003	120.02	0.003	0.003
ST-GAN	25.21	**0.265**	**0.040**	12.96	**0.271**	0.057
ST-GAN-VAE	23.95	0.166	0.027	14.19	0.207	0.041

4   Related work

Controllable generation in GAN   : Both GAN and VAE models have been utilized for achieving controllability, while some studies even indicate that the techniques in one do not work in the other [26]. While the majority of existing work on disentanglement focuses on a (semi-)supervised setting [27–34], our work focuses on the unsupervised seeting. Here, we review the most closely related methods below. Such methods can be classified into three categories: a) post-training interpretable methods, b) information theory motivated disentanglement, c) hierarchical disentanglement. The post-processing methods [35, 36] assume a pretrained generator and find interpretable directions in the latent space of the generator. Post-processing methods have twofold drawbacks: i) they do not have any guarantees that we will discover such interpretable directions, ii) multi-step training is required.

| (a) ST-GAN | (b) ST-GAN-VAE |

Figure 3: Each row depicts linear interpolation in a single element of the latent code. The rest elements remain the same. (a) The first row corresponds to the elements in the first split, while the second row corresponds to an element from the second split. The first row changes the pose and the second the colors (of hair and car respectively). (b) Similar patterns with the VAE model are observed, however the changes are less obvious.

The perspective of information theory is frequently used to tackle (unsupervised) disentanglement. The core idea relies on maximizing the mutual information between the latent codes and the synthesized images. The seminal work of InfoGAN [5] along with its extensions [26, 37, 38] use the mutual information to disentangle the factors of variation in a GAN setting. The drawback of mutual information is that they require strong inductive biases to disentangle the factors of variation [26].

A line of work that relates to our method is that of hierarchical disentanglement. FineGAN [39] utilizes bounding boxes to define a hierarchical disentanglement. Kaneko *et al.* [6] use instead class-level supervision to form the upper level of a hierarchy. The lower-levels are trained with strong inductive bias (e.g. one-hot vectors) and tailored curriculum learning methods. A core difference is that in the related works, they use some form of weak supervision.

The work that is most closely related to our work is that of [7] which combines VAE and GAN to achieve both high quality of synthesis and unsupervised disentanglement. However, the proposed method requires significant engineering (two-step training process), while it demonstrates only partial disentanglement in high-dimensional distributions. In addition, the latent codes from the latent variable model (VAE) is used as an auxiliary input to the GAN generator. By contrast, in our approach, generation is controlled solely by the learned latent variable model.

Latent variable models : Learning the joint distribution over both observed and latent variables is a crucial topic in machine learning. The normalizing flows [40, 41] provide an elegant way to learn the joint distribution; they learn an invertible network. However, their likelihood-based loss results in images that are not competitive to recent generative models. Recently, [42] prove that a prerequisite for learning the joint distribution is to have an identifiable model. They make a step further and modify VAE to achieve threefold goals: (i) fit the target distribution, (ii) disentanglement of the variables, (iii) identifiability of the model. They extend VAE, since they argue that it state-of-the-art in (i) and (ii). However, recent works [2] exhibit that GANs outperform VAEs in semi-supervised disentanglement on challenging high-resolution data. This motivates us to use latent variable models in a GAN for realistic generation. To that end, we use moments to fit effective latent variable models for capturing salient features in images.

Self-supervised learning and GAN : The concept of self-supervised learning has emerged in a GAN training [43–45]. In [46] the authors augment the GAN loss with an auxiliary loss to predict rotated versions of the image in the discriminator. In [47] they propose a self-supervised loss that can stabilize the training, while in [48] they involve both the generator and the discriminator for synthesizing samples in the trained model. Self-supervised GANs differ substantially from our goal; they predetermine a set of fixed auxiliary tasks (e.g. rotation), while in our case the goal is to drive the semantic generation through interpretable codes.

8

5   Conclusion

In this paper, we proposed the first self-trained GAN to enable controllable image generation without any supervision. Our method discovers disentangled and semantically interpretable factors of variation driving the generation in a fully unsupervised framework. The hidden factors are modelled explicitly by a flexible latent variable model, from which the generator samples its inputs. The parameters of that latent variable model are learned through a tensor factorization of higher-order cumulants. We empirically demonstrate that the codes learned by our model are semantically interpretable.

Acknowledgements

Broader Impact

Our work enables the synthesis of high-fidelity images through a self-learning method. Specifically, we augment the Generative Adversarial Network (GAN) for achieving controllable generation.

Our method is likely to encourage the consideration of controllability in the generation process. This will be beneficial for (commercial) applications. For instance, being able to edit the synthesized images on-demand can be a great addition to existing artistic tools or beautification-related applications. On the academic front, this controllability can be embraced for augmenting the training data.

In our work, we include carefully tuned modules in our framework to obtain interpretable generation. The discussion about interpretability and the inductive bias of the deep neural networks is challenging, but yet significant for their generalization.

On the other hand, our method can be potentially used for generation of 'real-looking' synthetic images . We emphasize though that such technologies are publicly available for several years, e.g. BiGAN. We encourage further work to understand how to detect synthesized images. Our method can be used for training powerful classifiers that detect synthesized images.

We encourage the research community to search for a technology that can separate the two types of content (generated vs real). The majority of the content is published online, therefore some form of explicit control of 'real' content could be an intermediate solution.

References

[1] 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 (NeurIPS)*, 2014.

[2] Weili Nie, Tero Karras, Animesh Garg, Shoubhik Debhath, Anjul Patney, Ankit B Patel, and Anima Anandkumar. Semi-supervised stylegan for disentanglement learning. *arXiv*, pages arXiv–2003, 2020.

[3] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. *arXiv preprint arXiv:1802.05983*, 2018.

[4] Tian Qi Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in neural information processing systems (NeurIPS)*, pages 2610–2620, 2018.

[5] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems (NeurIPS)*, pages 2172–2180, 2016.

[6] Takuhiro Kaneko, Kaoru Hiramatsu, and Kunio Kashino. Generative adversarial image synthesis with decision tree latent controller. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6606–6615, 2018.

[7] Wonkwang Lee, Donggyun Kim, Seunghoon Hong, and Honglak Lee. High-fidelity synthesis with disentangled representation. *arXiv preprint arXiv:2001.04296*, 2020.

[8] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning (ICML)*, 2019.

[9] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[10] Grigorios Chrysos, Stylianos Moschoglou, Yannis Panagakis, and Stefanos Zafeiriou. Polygan: High-order polynomial generators. *arXiv preprint arXiv:1908.06571*, 2019.

[11] Grigorios Chrysos, Stylianos Moschoglou, Giorgos Bouritsas, Jiankang Deng, Yannis Panagakis, and Stefanos Zafeiriou. $\pi-$nets: Deep polynomial neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[12] Maxim Kuznetsov, Daniil Polykovskiy, Dmitry P Vetrov, and Alex Zhebrak. A prior of a googol gaussians: a tensor ring induced prior for generative models. In *Advances in Neural Information Processing Systems*, pages 4104–4114, 2019.

[13] Jakub M Tomczak and Max Welling. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

[14] Matthias Bauer and Andriy Mnih. Resampled priors for variational autoencoders. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

[15] Pierre Comon. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

[16] Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15:2773–2832, 2014.

[17] Forough Arabshahi and Animashree Anandkumar. Spectral methods for correlated topic models. *arXiv preprint arXiv:1605.09080*, 2016.

[18] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision (ICCV)*, pages 3730–3738, 2015.

[19] Kimmo Kärkkäinen and Jungseock Joo. Fairface: Face attribute dataset for balanced race, gender, and age. *arXiv preprint arXiv:1908.04913*, 2019.

[20] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems (NeurIPS)*, pages 6626–6637, 2017.

[21] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NeurIPS-W*, 2017.

[22] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *International Conference on Learning Representations (ICLR)*, 2(5):6, 2017.

[23] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018.

[24] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[25] Jean Kossaifi, Yannis Panagakis, Anima Anandkumar, and Maja Pantic. Tensorly: Tensor learning in python. *Journal of Machine Learning Research*, 20(1):925–930, 2019.

[26] Zinan Lin, Kiran Koshy Thekumparampil, Giulia Fanti, and Sewoong Oh. Infogan-cr: Disentangling generative adversarial networks with contrastive regularizers. *arXiv preprint arXiv:1906.06034*, 2019.

[27] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 3, page 7, 2017.

[28] Jianfu Zhang, Yuanyuan Huang, Yaoyi Li, Weijie Zhao, and Liqing Zhang. Multi-attribute transfer via disentangled representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9195–9202, 2019.

[29] Taihong Xiao, Jiapeng Hong, and Jinwen Ma. Dna-gan: Learning disentangled representations from multi-attribute images. *arXiv preprint arXiv:1711.05415*, 2017.

[30] Scott Reed, Kihyuk Sohn, Yuting Zhang, and Honglak Lee. Learning to disentangle factors of variation with manifold interaction. In *International Conference on Machine Learning (ICML)*, pages 1431–1439, 2014.

[31] Attila Szabó, Qiyang Hu, Tiziano Portenier, Matthias Zwicker, and Paolo Favaro. Challenges in disentangling independent factors of variation. *arXiv preprint arXiv:1711.02245*, 2017.

[32] Zhijie Deng, Hao Zhang, Xiaodan Liang, Luona Yang, Shizhen Xu, Jun Zhu, and Eric P Xing. Structured generative adversarial networks. In *Advances in neural information processing systems (NeurIPS)*, pages 3899–3909, 2017.

[33] LI Chongxuan, Taufik Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets. In *Advances in neural information processing systems (NeurIPS)*, pages 4088–4098, 2017.

[34] Linh Tran, Jean Kossaifi, Yannis Panagakis, and Maja Pantic. Disentangling geometry and appearance with regularised geometry-aware generative adversarial networks. *International Journal of Computer Vision*, 127(6-7):824–844, 2019.

[35] Antoine Plumerault, Hervé Le Borgne, and Céline Hudelot. Controlling generative models with continuous factors of variations. In *International Conference on Learning Representations (ICLR)*, 2020.

[36] Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the gan latent space. *arXiv preprint arXiv:2002.03754*, 2020.

[37] Bingchen Liu, Yizhe Zhu, Zuohui Fu, Gerard de Melo, and Ahmed Elgammal. Oogan: Disentangling gan with one-hot sampling and orthogonal regularization. In *AAAI Conference on Artificial Intelligence*, 2020.

[38] Insu Jeon, Wonkwang Lee, and Gunhee Kim. Ib-gan: Disentangled representation learning with information bottleneck gan. 2018.

[39] Krishna Kumar Singh, Utkarsh Ojha, and Yong Jae Lee. Finegan: Unsupervised hierarchical disentanglement for fine-grained object generation and discovery. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6490–6499, 2019.

[40] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[41] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. In *International Conference on Learning Representations (ICLR)*, 2017.

[42] Ilyes Khemakhem, Diederik P Kingma, and Aapo Hyvärinen. Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

[43] Ting Chen, Xiaohua Zhai, and Neil Houlsby. Self-supervised gan to counter forgetting. *arXiv preprint arXiv:1810.11598*, 2018.

[44] Rui Huang, Wenju Xu, Teng-Yok Lee, Anoop Cherian, Ye Wang, and Tim Marks. Fx-gan: Self-supervised gan learning via feature exchange. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 3194–3202, 2020.

[45] Sheng Qian, Wen-ming Cao, Rui Li, Si Wu, et al. Self-supervised gan for image generation by correlating image channels. In *Pacific Rim Conference on Multimedia*, pages 78–88. Springer, 2018.

[46] Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. Self-supervised gans via auxiliary rotation loss. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12154–12163, 2019.

[47] Ngoc-Trung Tran, Viet-Hung Tran, Bao-Ngoc Nguyen, Linxiao Yang, et al. Self-supervised gan: Analysis and improvement with multi-class minimax game. In *Advances in neural information processing systems (NeurIPS)*, pages 13232–13243, 2019.

[48] Michael Arbel, Liang Zhou, and Arthur Gretton. Kale: When energy-based learning meets adversarial training. *arXiv preprint arXiv:2003.05033*, 2020.