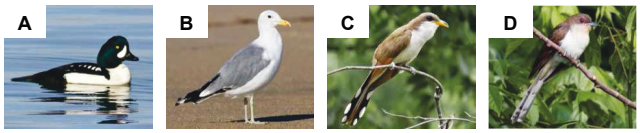
FineGAN: Unsupervised Hierarchical Disentanglement for Fine-Grained Object Generation and Discovery

Krishna Kumar Singh* Utkarsh Ojha* Yong Jae Lee
University of California, Davis

Abstract

We propose FineGAN, a novel unsupervised GAN framework, which disentangles the background, object shape, and object appearance to hierarchically generate images of fine-grained object categories. To disentangle the factors without supervision, our key idea is to use information theory to associate each factor to a latent code, and to condition the relationships between the codes in a specific way to induce the desired hierarchy. Through extensive experiments, we show that FineGAN achieves the desired disentanglement to generate realistic and diverse images belonging to fine-grained classes of birds, dogs, and cars. Using FineGAN's automatically learned features, we also cluster real images as a first attempt at solving the novel problem of unsupervised fine-grained object category discovery. Our code/models/demo can be found at https://github.com/kkanshul/finegan

1. Introduction



Consider the figure above: if tasked to group any of the images together, as humans we can easily tell that birds A and B should not be grouped with C and D as they have completely different backgrounds and shapes. But how about C and D? They share the same background, shape, and rough color. However, upon close inspection, we see that even C and D should not be grouped together as C's beak is yellow and its tails have large white spots while D's beak is black and its tails have thin white strips.[1] This example demonstrates that clustering fine-grained object categories requires not only *disentanglement* of the background, shape, and appearance (color/texture), but that it is naturally facilitated in a hierarchical fashion.

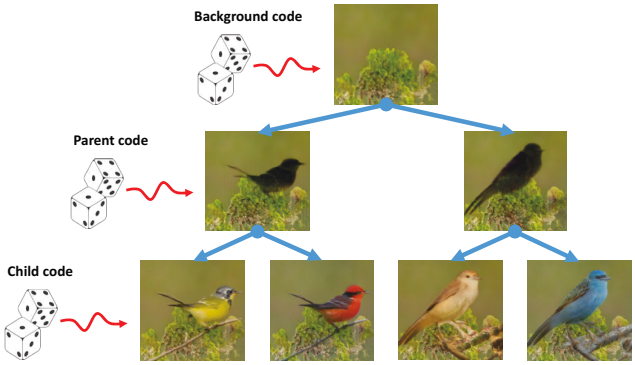


Figure 1. FineGAN disentangles the background, object shape (parent), and object appearance (child) to hierarchically generate fine-grained objects, *without mask or fine-grained annotations*.

In this work, we aim to develop a model that can do just that: model fine-grained object categories by hierarchically disentangling the background, object's shape, and its appearance, without any manual fine-grained annotations. Specifically, we make the first attempt at solving the novel problem of *unsupervised* fine-grained object clustering (or "discovery"). Although both unsupervised object discovery and fine-grained recognition have a long history, prior work on unsupervised object category discovery focus only on clustering entry-level categories (e.g., birds vs. cars vs. dogs) [17, 42, 31, 51, 47, 15], while existing work on fine-grained recognition focus exclusively on the supervised setting in which ground-truth fine-grained category annotations are provided [35, 52, 34, 4, 13, 33, 12, 7, 46].

Why unsupervised discovery for such a difficult problem? We have two key motivations. First, fine-grained annotations require domain experts. As a result, the overall annotation process is very expensive and standard crowdsourcing techniques cannot be used, which restrict the amount of training data that can be collected. Second, unsupervised learning enables the discovery of *latent structure* in the data, which may not have been labeled by annotators. For example, fine-grained image datasets often have an in-

*Equal contribution.

[1]The ground-truth fine-grained categories are A: *Barrow's Goldeneye*, B: *California Gull*, C: *Yellow-billed Cuckoo*, D: *Black-billed Cuckoo*.

herent *hierarchical organization* in which the categories can first be grouped based on one feature (e.g., shape) and then differentiated based on another (e.g., appearance).

Main Idea. We hypothesize that a generative model with the capability of hierarchically generating images with fine-grained details can also be useful for fine-grained grouping of real images. We therefore propose *FineGAN*, a novel hierarchical unsupervised Generative Adversarial Networks framework to generate images of fine-grained categories.

FineGAN generates a fine-grained image by hierarchically generating and stitching together a background image, a parent image capturing one factor of variation of the object, and a child image capturing another factor. To disentangle the two factors of variation of the object *without any supervision*, we use information theory, similar to InfoGAN [9]. Specifically, we enforce high mutual information between (1) the parent latent code and the parent image, and (2) the child latent code, *conditioned on the parent code*, and the child image. By imposing constraints on the relationship between the parent and child latent codes (specifically, by grouping child codes such that each group has the same parent code), we can induce the parent and child codes to capture the object's shape and color/texture details, respectively; see Fig. 1. This is because in many fine-grained datasets, objects often differ in appearance conditioned on a shared shape (e.g., 'Yellow-billed Cuckoo' and 'Black-billed Cuckoo', which share the same shape but differ in their beak color and wing patterns).

Moreover, FineGAN automatically generates masks at both the parent and child stages, which help condition the latent codes to focus on the relevant object factors as well as to stitch together the generated images across the stages. Ultimately, the features learned through this unsupervised hierarchical image generation process can be used to cluster real images into their fine-grained classes.

Contributions. Our work has two main contributions:

(1) We introduce FineGAN, an unsupervised model that learns to hierarchically generate the background, shape, and appearance of fine-grained object categories. Through various qualitative evaluations, we demonstrate FineGAN's ability to accurately disentangle background, object shape, and object appearance. Furthermore, quantitative evaluations on three benchmark datasets (CUB [45], Stanford-dogs [27], and Stanford-cars [29]) demonstrate FineGAN's strength in generating realistic and diverse images.

(2) We use FineGAN's learned disentangled representation to cluster real images for unsupervised fine-grained object category discovery. It produces fine-grained clusters that are significantly more accurate than those of state-of-the-art unsupervised clustering approaches (JULE [51] and DEPICT [15]). To our knowledge, this is the first attempt to cluster fine-grained categories in the unsupervised setting.

2. Related work

Fine-grained category recognition involves classifying subordinate categories within entry-level categories (e.g., different species of birds), which requires annotations from domain experts [35, 52, 34, 4, 13, 8, 33, 12, 28, 58, 46]. Some methods require additional part [56, 6, 53], attribute [14], or text [37, 19] annotations. Our work makes the first attempt to overcome the dependency on expert annotations by performing *unsupervised* fine-grained category discovery without any class annotations.

Visual object discovery and clustering. Early work on unsupervised object discovery [41, 17, 42, 31, 32, 39] use handcrafted features to cluster object categories from unlabeled images. Others explore the use of natural language dialogue for object discovery [10, 59]. Recent unsupervised deep clustering approaches [51, 47, 15] demonstrate state-of-the-art results on datasets whose objects have large variations in high-level detail like shape and background. On fine-grained category datasets, we show that FineGAN significantly outperforms these methods as it is able to focus on the fine-grained object details.

Disentangled representation learning has a vast literature (e.g., [3, 44, 22, 49, 9, 21, 11, 23]). The most related work in this space is InfoGAN [9], which learns disentangled representations without any supervision by maximizing the mutual information between the latent codes and generated data. Our work builds on the same principles of information theory, but we extend it to learn a *hierarchical* disentangled representation. Specifically, unlike InfoGAN in which all details of an object are generated together, FineGAN provides explicit distentanglement and control over the generation of background, shape, and appearance, which we show is especially important when modeling fine-grained categories.

GANs and Stagewise image generation. Unconditional GANs [16, 36, 43, 57, 1, 18] can generate realistic images without any supervision. However, unlike our approach, these methods do not generate images hierarchically and do not have explicit control over the background, object's shape, and object's appearance. Some conditional supervised approaches [38, 54, 55, 5] learn to generate fine-grained images with text descriptions. One such approach, FusedGAN [5], generates fine-grained objects with specific pose and shape but it cannot decouple them, and lacks explicit control over the background. In contrast, FineGAN can generate fine-grained images without any text supervision and with full control over the background, pose, shape, and appearance. Also related are stagewise image generators [24, 30, 50, 26]. In particular, LR-GAN [50] generates the background and foreground separately and stitches them. However, both are controlled by a single random vec-

tor, and it does not disentangle the object's shape from appearance.

3. Approach

Let $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$ be a dataset containing unlabeled images of fine-grained object categories. Our goal is to learn an unsupervised generative model, FineGAN, which produces high quality images matching the true data distribution $p_{data}(x)$, while also learning to disentangle the relevant factors of variation associated with images in $\mathcal{X}$.

We consider background, shape, appearance, and pose/location of the object as the factors of variation in this work. If FineGAN can successfully associate each latent code to a particular fine-grained category aspect (e.g., like a bird's shape and wing color), then its learned features can also be used to group the real images in $\mathcal{X}$ for unsupervised find-grained object category discovery.

3.1. Hierarchical fine-grained disentanglement

Fig. 2 shows our FineGAN architecture for modeling and generating fine-grained object images. The overall process has three interacting stages: background, parent, and child. The background stage generates a realistic background image $\mathcal{B}$. The parent stage generates the outline (shape) of the object and stitches it onto $\mathcal{B}$ to produce parent image $\mathcal{P}$. The child stage fills in the object's outline with the appropriate color and texture, to produce the final child image $\mathcal{C}$. The objective function of the complete process is:

$$\mathcal{L} = \lambda\mathcal{L}_b + \beta\mathcal{L}_p + \gamma\mathcal{L}_c$$

where $\mathcal{L}_b$, $\mathcal{L}_p$ and $\mathcal{L}_c$ denote the objectives for the background, parent, and child stage respectively, with λ, β and γ denoting their weights. We train all stages end-to-end.

The different stages get conditioned with different latent codes, as seen from Fig. 2. FineGAN takes as input: i) a continuous noise vector $z \sim \mathcal{N}(0, 1)$; ii) a categorical background code $b \sim \text{Cat}(K = N_b, p = 1/N_b)$; iii) a categorical parent code $p \sim \text{Cat}(K = N_p, p = 1/N_p)$; and iv) a categorical child code $c \sim \text{Cat}(K = N_c, p = 1/N_c)$.

Relationship between latent codes: (1) *Parent code and child code.* We assume the presence of an implicit hierarchy in $\mathcal{X}$ – as mentioned previously, fine-grained categories can often be grouped first based on a common shape and then differentiated based on appearance. To help discover this hierarchy, we impose two constraints: (i) the number of categories of parent code is set to be less than that of child code ($N_p < N_c$), and (ii) for each parent code p, we tie a fixed number of child codes c to it (multiple child codes share the same parent code). We will show that these constraints help push p to capture shape and c to capture appearance. For example, if the shape identity captured by p is that of a duck, then the list of c's tied to this p would all share the same duck shape, but vary in their color and texture.

(2) *Background code and child code.* There is usually some correlation between an object and the background in which it is found (e.g., ducks in water). Thus, to avoid conflicting object-background pairs (which a real/fake discriminator could easily exploit to tell that an image is fake), we set the background code to be the same as the child code during training ($b = c$). However, we can easily relax this constraint during testing (e.g., to generate a duck in a tree).

3.1.1 Background stage

The background stage synthesizes a background image $\mathcal{B}$, which acts as a canvas for the parent and child stages to stitch different foreground aspects on top of $\mathcal{B}$. Since we aim to disentangle background as a separate factor of variation, $\mathcal{B}$ should not contain any foreground information. We hence separate the background stage from the parent and child stages, which share a common feature pipeline. This stage consists of a generator G_b and a discriminator pair, D_b and D_{aux}. G_b is conditioned on latent background code b, which controls the different (unknown) background classes (e.g., trees, water, sky), and on latent code z, which controls intra-class background details (e.g., positioning of leaves). To generate the background, we assume access to an object bounding box detector that can detect instances of the super-category (e.g., bird). We use the detector to locate non-object background patches in each real image x_i. We then train G_b and D_b using two objectives: $\mathcal{L}_b = \mathcal{L}_{bg\_adv} + \mathcal{L}_{bg\_aux}$, where $\mathcal{L}_{bg\_adv}$ is the adversarial loss [16] and $\mathcal{L}_{bg\_aux}$ is the auxiliary background classification loss.

For the adversarial loss $\mathcal{L}_{bg\_adv}$, we employ the discriminator D_b on a patch level [25] (we assume background can easily be modeled as texture) to predict an $N \times N$ grid with each member indicating the real/fake score for the corresponding patch in the input image:

$$\mathcal{L}_{bg\_adv} = \min_{G_b} \max_{D_b} \mathbb{E}_x[\log(D_b(x))] + \mathbb{E}_{z,b}[\log(1 - D_b(G_b(z, b)))]$$

The auxiliary classification loss $\mathcal{L}_{bg\_aux}$ makes the background generation task more explicit, and is also computed on a patch level. Specifically, patches inside (r_i) and outside (r_o) the detected object in real images constitute the training set for foreground (1) and background (0) respectively, and is used to train a binary classifier D_{aux} with cross-entropy loss. We then use D_{aux} to train the generator G_b:

$$\mathcal{L}_{bg\_aux} = \min_{G_b} \mathbb{E}_{z,b}[\log(1 - D_{aux}(G_b(z, b)))]$$

This loss updates G_b so that D_{aux} assigns a high background probability to the generated background patches.

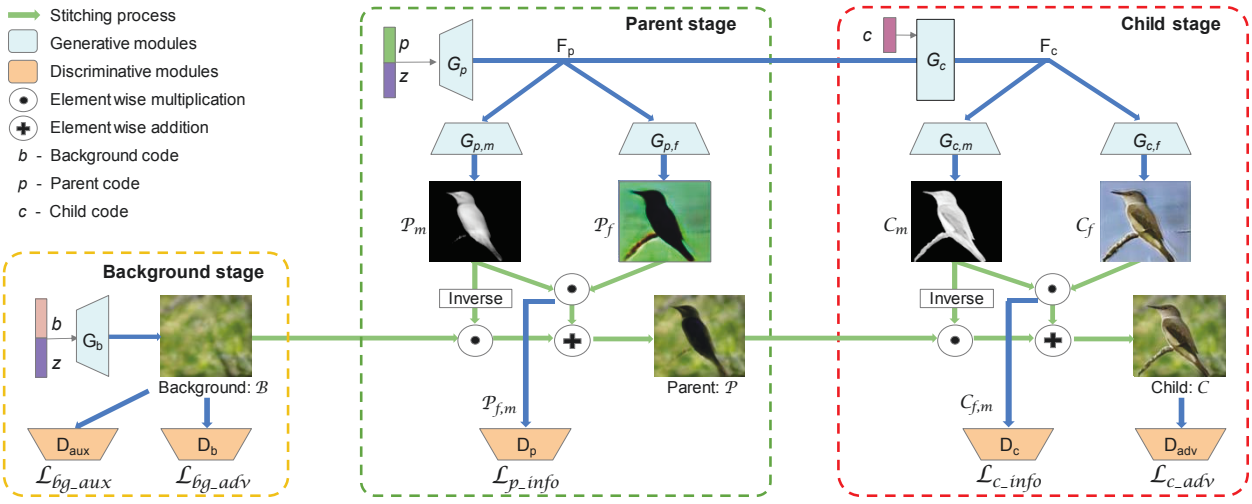


Figure 2. **FineGAN architecture** for hierarchical fine-grained image generation. The background stage, conditioned on random vector z and background code b, generates the background image B. The parent stage, conditioned on z and parent code p, uses B as a canvas to generate parent image P, which captures the shape of the object. The child stage, conditioned on c, uses P as a canvas to generate the final child image C with the object's appearance details stitched into the shape outline.

3.1.2 Parent stage

As explained previously, we model the real data distribution $p_{data}(x)$ through a two-level foreground generation process via the parent and child stages. The parent stage can be viewed as modeling higher-level information about an object like its shape, and the child stage, *conditioned on the parent stage*, as modeling lower-level information like its color/texture.

Capturing multi-level information in this way can have potential advantages. First, it makes the overall image generation process more principled and easier to interpret; different sub-networks of the model can focus only on synthesizing entities they are concerned with, in contrast to the case where the entire network performs single-shot image generation. Second, for fine-grained generation, it should be easier for the model to generate appearance details *conditioned* on the object's shape, without having to worry about the background and other variations. With the same reasoning, such hierarchical features—parent capturing shape and child capturing appearance—should also be beneficial for fine-grained categorization compared to a flat-level feature representation.

We now discuss the working details of the parent stage. As shown in Fig. 2, G_p, which consists of a series of convolutional layers and residual blocks, maps z and p to feature representation F_p. As discussed previously, the requirement from this stage is only to generate a foreground entity, and stitch it to the existing background $\mathcal{B}$. Consequently, two generators $G_{p,f}$ and $G_{p,m}$ transform F_p into parent foreground ($\mathcal{P}_f$) and parent mask ($\mathcal{P}_m$) respectively, so that P_m can be used to stitch P_f on B, to obtain the parent image $\mathcal{P}$:

$$\mathcal{P} = \mathcal{P}_{f,m} + \mathcal{B}_m$$

where $\mathcal{P}_{f,m} = \mathcal{P}_m \odot \mathcal{P}_f$ and $\mathcal{B}_m = (1 - \mathcal{P}_m) \odot \mathcal{B}$ denote masked foreground and inverse masked background image respectively; see green arrows in Fig. 2. This idea of generating a mask and using it for stitching is inspired by LR-GAN [50].

We again employ a discriminator at the parent stage, and denote it as D_p. Its functioning however, differs from the discriminators employed at the other stages. This is because in contrast to the background and child stages where we know the true distribution to be modeled, the true distribution for $\mathcal{P}$ or $\mathcal{P}_{f,m}$ is unknown (i.e., we have real patch samples of background and real image samples of the object, but we do not have any real intermediate image samples in which the object exhibits one factor like shape but lacks another factor like appearance). Consequently, we cannot use the standard GAN objective to train D_p.

Thus, we only use D_p to induce the parent code p to represent the hierarchical concept i.e., the object's shape. With no supervision from image labels, we exploit information theory to discover this concept in a completely unsupervised manner, similar to InfoGAN [9]. Specifically, we maximize the mutual information $I(p, \mathcal{P}_{f,m})$, with D_p approximating the posterior $P(p|\mathcal{P}_{f,m})$:

$$\mathcal{L}_p = \mathcal{L}_{p\_info} = \max_{D_p, G_{p,f}, G_{p,m}} \mathbb{E}_{z,p}[\log D_p(p|\mathcal{P}_{f,m})]$$

We use $\mathcal{P}_{f,m}$ instead of $\mathcal{P}$ so that D_p makes its decision solely based on the foreground object (shape) and not get influenced by the background. In simple words, D_p is asked to reconstruct the latent hierarchical category information (p) from $\mathcal{P}_{f,m}$, which already has this information encoded during its synthesis. Given our constraints from Sec. 3.1 that there are less parent categories than child ones ($N_p < N_c$) and multiple child codes share the same parent code,

FineGAN tries encoding p into $\mathcal{P}_{f,m}$ as an attribute that: (i) by itself cannot capture all fine-grained category details, and (ii) is *common* to multiple fine-grained categories, which is the essence of hierarchy.

3.1.3 Child stage

The result of the previous stages is an image that is a composition of the background and object's outline. The task that remains is filling in the outline with appropriate texture/color to generate the final fine-grained object image.

As shown in Fig. 2, we encode the color/texture information about the object with child code c, which is itself conditioned on parent code p. Concatenated with F_p, the resulting feature chunk is fed into G_c, which again consists of a series of convolutional and residual blocks. Analogous to the parent stage, two generators $G_{c,f}$ and $G_{c,m}$ map the resulting feature representation F_c into child foreground ($\mathcal{C}_f$) and child mask ($\mathcal{C}_m$) respectively. The stitching process to obtain the complete child image $\mathcal{C}$ is:

$$\mathcal{C} = \mathcal{C}_{f,m} + \mathcal{P}_{c,m}$$

where $\mathcal{C}_{f,m} = \mathcal{C}_m \odot \mathcal{C}_f$, and $\mathcal{P}_{c,m} = (1 - \mathcal{C}_m) \odot \mathcal{P}$.

We now discuss the requirements for the child stage discriminative networks, D_{adv} and D_c: (i) discriminate between real samples from $\mathcal{X}$ and fake samples from the generative distribution using D_{adv}; (ii) use D_c to approximate the posterior $P(c|C_{f,m})$ to associate the latent code c to a fine-grained object detail like color and texture. The loss function can hence be broken down into two components $\mathcal{L}_c = \mathcal{L}_{c\_adv} + \mathcal{L}_{c\_info}$, where:

$$\mathcal{L}_{c\_adv} = \min_{G_c} \max_{D_{adv}} \mathbb{E}_x[\log(D_{adv}(x))] + \mathbb{E}_{z,b,p,c}[\log(1 - D_{adv}(\mathcal{C}))],$$

$$\mathcal{L}_{c\_info} = \max_{D_c, G_{c,f}, G_{c,m}} \mathbb{E}_{z,p,c}[\log D_c(c|\mathcal{C}_{f,m})].$$

Again, we use $\mathcal{C}_{f,m}$ instead of $\mathcal{C}$ so that D_c makes its decision solely based on the object (color/texture and shape) and not get influenced by the background. With shape already captured though the parent code p, the child code c can now solely focus to correspond to the texture/color inside the shape.

3.2. Fine-grained object category discovery

Given our trained FineGAN model, we can now use it to compute features for the real images $x_i \in \mathcal{X}$ to cluster them into fine-grained object categories. In particular, we can make use of the final synthetic images $\{\mathcal{C}_j\}$ and their associated parent and child codes to learn a mapping from images to codes. Note that we cannot directly use the parent and child discriminators D_p and D_c—which each categorize $\{\mathcal{P}_{f,m}\}$ and $\{\mathcal{C}_{f,m}\}$ into one of the parent and child codes respectively—-on the real images due to the unavailability of real foreground masks. Instead, we train a pair of convolutional networks (ϕ_p and ϕ_c) to predict the parent and child codes of the final set of synthetic images $\{\mathcal{C}_j\}$:

1. Randomly sample a batch of codes: $z \sim \mathcal{N}(0, 1)$, $p \sim p_p$, $c \sim p_c$, $b \sim p_b$ to generate child images $\{\mathcal{C}_j\}$.
2. Feed forward this batch through ϕ_p and ϕ_c. Compute cross-entropy loss $CE(p, \phi_p(\mathcal{C}_j))$ and $CE(c, \phi_c(\mathcal{C}_j))$.
3. Update ϕ_p and ϕ_c. Repeat till convergence.

To accurately predict parent code p from $\mathcal{C}_j$, ϕ_p has to solely focus on the object's shape as no sensible supervision can come from the randomly chosen background and child codes. With similar reasoning, ϕ_c has to solely focus on the object's appearance to accurately predict child code c. Once ϕ_p and ϕ_c are trained, we use them to extract features for each real image $x_i \in \mathcal{X}$. Finally, we use their concatenated features to group the images with k-means clustering.

4. Experiments

We first evaluate FineGAN's ability to disentangle and generate images of fine-grained object categories. We then evaluate FineGAN's learned features for fine-grained object clustering with real images.

Datasets and implementation details. We evaluate on three fine-grained datasets: (1) **CUB** [45]: 200 bird classes. We use the entire dataset (11,788 images); (2) **Stanford Dogs** [27]: 120 dog classes. We use its train data (12,000 images); (3) **Stanford Cars** [29]: 196 car classes. We use its train data (8,144 images). *We do not use any of the provided labels for training. The labels are only used for evaluation.* Number of parents and children are set as: (1) CUB: N_p = 20, N_c = 200; (2) Stanford dogs: N_p = 12, N_c = 120; and (3) Cars: N_p = 20, N_c = 196. N_c matches the ground-truth number of fine-grained classes per dataset. We set λ = 10, β = 1 and γ = 1 for all datasets.

4.1. Fine-grained image generation

We first analyze FineGAN's image generation in terms of realism and diversity. We compare to:

- **Simple-GAN**: Generates a final image ($\mathcal{C}$) in one shot without the parent and background stages. Only has $\mathcal{L}_{c\_adv}$ loss at the child stage. This baseline helps gauge the importance of disentanglement learned by $\mathcal{L}_{c\_info}$. For fair comparison, we use FineGAN's backbone architecture.
- **InfoGAN** [9]: Same as Simple-GAN but with additional $\mathcal{L}_{c\_info}$. This baseline helps analyze the importance of hierarchical disentanglement between background, shape, and appearance during image generation, which is lacking in InfoGAN. N_c is set to be the same as FineGAN for each dataset. We again use FineGAN's backbone architecture.
- **LR-GAN** [50]: It also generates an image stagewise, which is similar to our approach. But its stages only consist of foreground and background, and that too controlled by single random vector z.

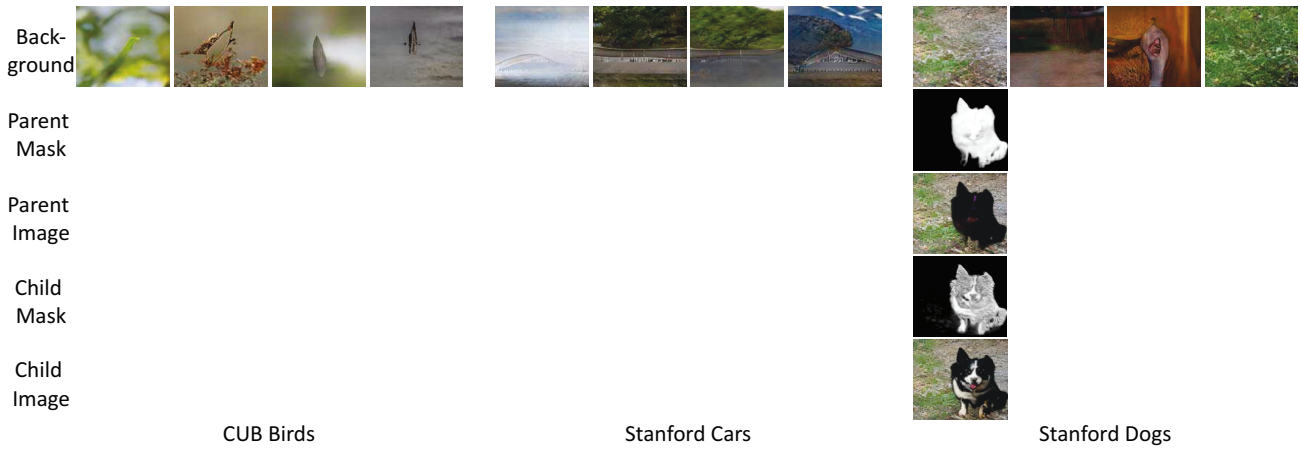


Figure 3. **FineGAN's stagewise image generation.** Background stage generates a background which is retained over the child and parent stages. Parent stage generates a hollow image with only the object's shape, and child stage fills in the appearance to complete the image.

- **StackGAN-v2** [55]: Its unconditional version generates images at multiple scales with $\mathcal{L}_{c\_adv}$ at each scale. This helps gauge how FineGAN fares against a state-of-the-art unconditional image generation approach.

For LR-GAN and StackGAN-v2, we use the authors' publicly-available code. We evaluate image generation using Inception Score (**IS**) [40] and Frechet Inception Distance (**FID**) [20], which are computed on 30K randomly generated images (equal number of images for each child code c), using an Inception Network fine-tuned on the respective datasets [2]. We evaluate on 128 x 128 generated images for all methods except LR-GAN, for which 64 x 64 generated images give better performance.

4.1.1 Quantitative evaluation of image generation

FineGAN obtains Inception Scores and FIDs that are favorable when compared to the baselines (see Table 1), which shows it can generate images that closely match the real data distribution.

In particular, the lower scores by Simple-GAN, LR-GAN, and StackGAN-v2 show that relying on a single adversarial loss can be insufficient to model fine-grained details. Both FineGAN and InfoGAN learn to associate a c code to a variation factor ($\mathcal{L}_{c\_info}$) to generate more detailed images. But by further disentangling the background and object shape (parent), FineGAN learns to generate more diverse images. LR-GAN also generates an image stagewise but we believe it has lower performance as it only separates foreground and background, which appears to be insufficient for capturing fine-grained details. These results strongly suggest that FineGAN's hierarchical disentanglement is important for better fine-grained image generation.

How sensitive is FineGAN to the number of parents? Table 2 shows the Inception Score (IS) on CUB of FineGAN trained with varying number of parents while keeping the number of children fixed (200). IS remains consistently high unless we have very small (N_p=5) or large (N_p=40) number of parents. With very small N_p, we limit diversity in the number of object shapes, and with very high N_p, the model has less opportunity to take advantage of the implicit hierarchy in the data. With variable number of children per parent (N_p=mixed: 6 parents with 5 children, 3 parents with 20 children, and 11 parents with 10 children), IS remains high, which shows there is no need to have the same number of children for each parent. These results show that FineGAN is robust to a wide range of parent choices.

	IS			FID		
	Birds	Dogs	Cars	Birds	Dogs	Cars
Simple-GAN	31.85 ± 0.17	6.75 ± 0.07	20.92 ± 0.14	16.69	261.85	33.35
InfoGAN [9]	47.32 ± 0.77	43.16 ± 0.42	28.62 ± 0.44	13.20	29.34	17.63
LR-GAN [50]	13.50 ± 0.20	10.22 ± 0.21	5.25 ± 0.05	34.91	54.91	88.80
StackGANv2 [55]	43.47 ± 0.74	37.29 ± 0.56	$\mathbf{33.69 \pm 0.44}$	13.60	31.39	16.28
FineGAN (ours)	$\mathbf{52.53 \pm 0.45}$	$\mathbf{46.92 \pm 0.61}$	32.62 ± 0.37	**11.25**	**25.66**	**16.03**

Table 1. Inception Score (higher is better) and FID (lower is better). FineGAN consistently generates diverse and real images that compare favorably to those of state-of-the-art baselines.

	N_p=20	N_p=10	N_p=40	N_p=5	N_p=mixed
Inception Score (CUB)	**52.53**	52.11	49.62	46.68	51.83

Table 2. Varying number of parent codes N_p, with number of children N_c fixed to 200. FineGAN is robust to a wide range of N_p.

4.1.2 Qualitative evaluation of image generation

We next qualitatively analyze FineGAN's (i) image generation process; (ii) disentanglement of the factors of variation; and provide (iii) in-depth comparison to InfoGAN.

Image generation process. Fig. 3 shows the intermediate images generated for CUB, Stanford Cars, and Stanford Dogs. The background images (1st row) capture the context of each dataset well; e.g., they contain roads for cars, gardens or indoor scene for dogs, leafy backgrounds for birds. The parent stage produces parent masks that capture each object's shape (2nd row), and a textureless, hollow entity as

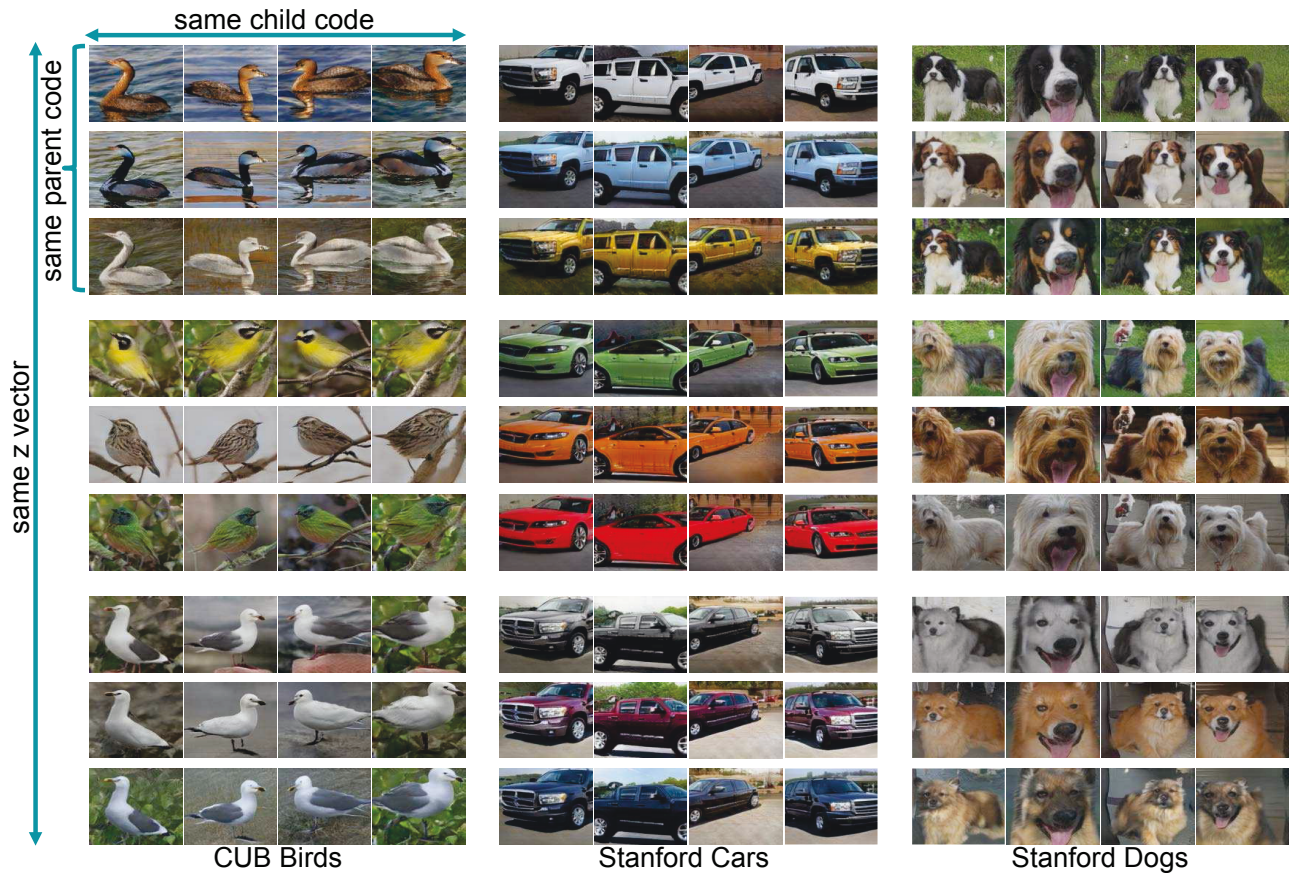


Figure 4. **Varying p vs. c vs. z.** Every three rows correspond to the same parent code p and each row has a different child code c. For the same parent, the object's shape remains consistent while the appearance changes with different child codes. For the same child, the appearance remains consistent. Each column has the same random vector z – we see that it controls the object's pose and position.

the parent image (3rd row) together with the background. The final child stage produces a more detailed mask (4th row) and the final composed image (last row), which has the same foreground shape as that of the parent image with added texture/color details. Note that the generation of accurate masks at each stage is important for the final composed image to retain the background, and is obtained without any mask supervision during training. We present additional quantitative analyses on the quality of the masks in the supplementary material.

Disentanglement of factors of variation. Fig. 4 shows the discovered grouping of parent and child codes by FineGAN. Each row corresponds to different instances with the same child code. Two observations can be made as we move left to right: (i) there is a consistency in the appearance and shape of the foreground objects; (ii) background changes slightly, giving an impression that the background across a row belongs to the same class, but with slight modifications. For each dataset, each set of three rows corresponds to three distinct children of the same parent, which is evident from their common shape. Notice that different child codes for the same parent can capture fine-grained differences in the appearance of the foreground object (e.g., dogs in the third row differ from those in first only because of small brown patches; similarly, birds in the 7th and last rows differ only in their wing color). Finally, the consistency in object viewpoint and pose along each column shows that FineGAN has learned to associate z with these factors.

Disentanglement of parent vs. child. Fig. 5 further analyzes the disentanglement of parent (shape) and child code (appearance). Across the rows, we vary parent code p while keeping child code c constant, which changes the bird's shape but keeps the texture/color the same. Across the columns, we vary child code c while keeping parent code p constant, which changes the bird's color/texture but keeps the shape the same. This result illustrates the control that FineGAN has learned *without any corresponding supervision* over the shape and appearance of a bird. Note that we keep background code b to be same across each column.

Disentanglement of background vs. foreground. The figure below shows disentanglement of background from object. In (a), we keep background code b constant and vary the parent and child code, which generates different birds

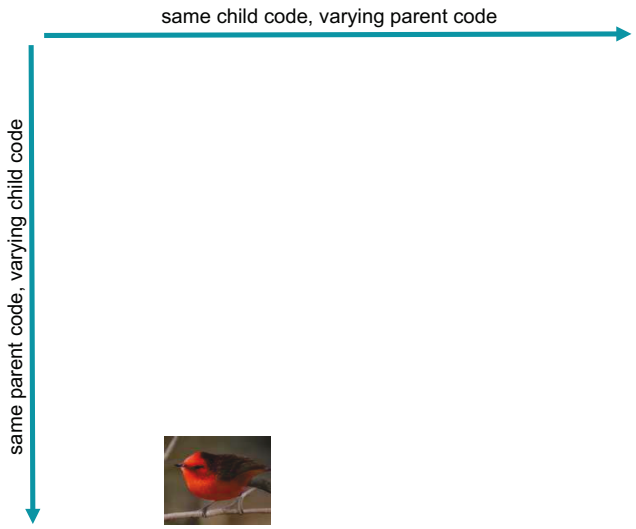
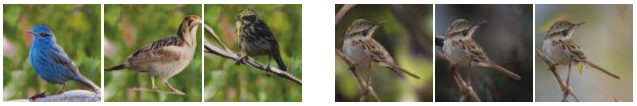


Figure 5. **Disentanglement of parent vs. child codes.** Shape is retained over the column, appearance is retained over the row.

over the same background. In (b), we keep the parent and child codes constant and vary the background code, which generates an identical bird with different backgrounds.



(a) Fixed b, varying p and c (b) Fixed p and c, varying b

Comparison with InfoGAN. In InfoGAN [9], the latent code prediction is based on the complete image, in contrast to FineGAN which uses the masked foreground. Due to this, InfoGAN's child code prediction can be biased by the background (see Fig. 6). Furthermore, InfoGAN [9] does not hierarchically disentangle the latent factors. To enable InfoGAN to model the hierarchy in the data, we tried conditioning its generator on both the parent and child codes, and ask the discriminator to predict both. This improves performance slightly (IS: 48.06, FID: 12.84 for birds), but is still worse than FineGAN. This shows that simply adding a parent code constraint to InfoGAN does not lead it to produce the hierarchical disentanglement that FineGAN achieves.

4.2. Fine-grained object category discovery

We next evaluate FineGAN's learned features for clustering real images into fine-grained object categories. We compare against the state-of-the-art deep clustering approaches **JULE** [51] and **DEPICT** [15]. To make them even more competitive, we also create a JULE variant with ResNet-50 backbone (**JULE-ResNet-50**) and DEPICT with double the number of filters in each conv layer (**DEPICT-Large**). We use code provided by the authors. All methods cluster the same image regions.

For evaluation we use Normalized Mutual Information (**NMI**) [48] and **Accuracy** (of best mapping between cluster assignments and true labels) following [15]. Our approach outperforms the baselines on all three datasets (see Table 3). This indicates that FineGAN's features learned for hierarchical image generation are better able to capture the fine-grained object details necessary for fine-grained object clustering. JULE and DEPICT are unable to capture those details to the same extent; instead, they focus more on high-level details like background and rough shape (see supp. for examples). Increasing their capacity (JULE-RESNET-50 and DEPICT-Large) gives little improvement. Finally, if we only use our child code features, then performance drops (0.017 in Accuracy on birds). This shows that the parent code and child code features are complementary and capture different aspects (shape vs. appearance).

Figure 6. **InfoGAN results.** Images in each group have same child code. The birds are the same, but so are their backgrounds. This strongly suggests InfoGAN takes background into consideration when categorizing the images. In contrast, FineGAN's generated images (Fig. 4) for same c show reasonable variety in background.

	NMI			Accuracy		
	Birds	Dogs	Cars	Birds	Dogs	Cars
JULE [51]	0.204	0.142	0.232	0.045	0.043	0.046
JULE-ResNet-50 [51]	0.203	0.148	0.237	0.044	0.044	0.050
DEPICT [15]	0.290	0.182	0.329	0.061	0.052	0.063
DEPICT-Large [15]	0.297	0.183	0.330	0.061	0.054	0.062
Ours	**0.403**	**0.233**	**0.354**	**0.126**	**0.079**	**0.078**

Table 3. Our approach outperforms existing clustering methods.

5. Discussion and limitations

There are some limitations of FineGAN worth discussing. First, although we have shown that our model is robust to a wide range of number of parents (Table 2), it along with the number of children are hyperparameters that a user must set, which can be difficult when the true number of categories is unknown (a problem common to most unsupervised grouping methods). Second, the latent modes of variation that FineGAN discovers may not necessarily correspond to those defined/annotated by a human. For example, our results in Fig. 4 for cars show that the children are grouped based on color rather than car model type. This highlights the importance of a good evaluation metric for unsupervised methods. Finally, while we significantly outperform unsupervised clustering methods, we are far behind fully-supervised fine-grained recognition methods. Nonetheless, we feel that this paper has taken important initial steps in tackling the challenging problem of unsupervised fine-grained object modeling.

Acknowledgments. This work was supported in part by NSF IIS-1751206, IIS-1748387, AWS ML Research Award, Google Cloud Platform research credits, and GPUs donated by NVIDIA.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[2] Shane Barratt and Rishi Sharma. A note on the inception score. In *arXiv:1801.01973*, 2018.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *TPAMI*, 2013.

[4] Thomas Berg and Peter Belhumeur. Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *CVPR*, 2013.

[5] Navaneeth Bodla, Gang Hua, and Rama Chellappa. Semi-supervised fusedgan for conditional image generation. In *ECCV*, 2018.

[6] Steve Branson, Grant Van Horn, Serge Belongie, and Pietro Perona. Bird species categorization using pose normalized deep convolutional nets. *BMVC*, 2014.

[7] Sijia Cai, Wangmeng Zuo, and Lei Zhang. Higher-order integration of hierarchical convolutional activations for fine-grained visual categorization. In *ICCV*, 2017.

[8] Yuning Chai, Victor Lempitsky, and Andrew Zisserman. Symbiotic segmentation and part localization for fine-grained categorization. In *ICCV*, 2013.

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

[10] Harm de Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron Courville. Guess-what?! visual object discovery through multi-modal dialogue. In *CVPR*, 2017.

[11] Emily L Denton and vighnesh Birodkar. Unsupervised learning of disentangled representations from video. In *NIPS*, 2017.

[12] Yang Gao, Oscar Beijbom, Ning Zhang, and Trevor Darrell. Compact bilinear pooling. In *CVPR*, 2016.

[13] Efstratios Gavves, Basura Fernando, Cees GM Snoek, Arnold WM Smeulders, and Tinne Tuytelaars. Fine-grained categorization by alignments. In *ICCV*, 2013.

[14] Timnit Gebru, Judy Hoffman, and Li Fei-Fei. Fine-grained recognition in the wild: A multi-task domain adaptation approach. In *ICCV*, 2017.

[15] Kamran Ghasedi Dizaji, Amirhossein Herandi, Cheng Deng, Weidong Cai, and Heng Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *ICCV*, 2017.

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

[17] Kristen Grauman and Trevor Darrel. Unsupervised learning of categories from sets of partially matching image features. In *CVPR*, 2006.

[18] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *NIPS*, 2017.

[19] Xiangteng He and Yuxin Peng. Fine-grained image classification via combining vision and language. In *CVPR*, 2017.

[20] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Gunter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS*, 2017.

[21] 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. In *ICLR*, 2017.

[22] Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang. Transforming auto-encoders. In *ICANN*, 2011.

[23] Qiyang Hu, Attila Szab, Tiziano Portenier, Paolo Favaro, and Matthias Zwicker. Disentangling factors of variation by mixing them. In *CVPR*, 2018.

[24] Daniel Jiwoong Im, Chris Dongjoo Kim, Hui Jiang, and Roland Memisevic. Generating images with recurrent adversarial networks. *http://arxiv.org/abs/1602.05110*, 2016.

[25] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017.

[26] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *CVPR*, 2018.

[27] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization*, 2011.

[28] Shu Kong and Charless Fowlkes. Low-rank bilinear pooling for fine-grained classification. In *CVPR*, 2017.

[29] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, 2013.

[30] Hanock Kwak and Byoung-Tak Zhang. Generating images part by part with composite generative adversarial networks. *arXiv preprint arXiv:1607.05387*, 2016.

[31] Yong Jae Lee and Kristen Grauman. Unsupervised learning of categories from sets of partially matching image features. In *CVPR*, 2010.

[32] Yong Jae Lee and Kristen Grauman. Learning the easy things first: Self-paced visual category discovery. In *CVPR*, 2011.

[33] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *ICCV*, 2015.

[34] Jiongxin Liu, Angjoo Kanazawa, David Jacobs, and Peter Belhumeur. Dog breed classification using part localization. In *ECCV*, 2012.

[35] M-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008.

[36] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *ICLR*, 2016.

[37] Scott Reed, Zeynep Akata, Honglak Lee, and Bernt Schiele. Learning deep representations of fine-grained visual descriptions. In *CVPR*, 2016.

[38] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text-to-image synthesis. In *ICML*, 2016.

[39] Michael Rubinstein, Armand Joulin, Johannes Kopf, and Ce Liu.. Unsupervised joint object discovery and segmentation in internet images. In *CVPR*, 2013.

[40] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NIPS*, 2016.

[41] Joseph Sivic, Bryan Russell, Alexei Efros, Andrew Zisserman, and William Freeman. Discovering objects and their location in images. In *ICCV*, 2005.

[42] Joseph Sivic, Bryan Russell, Andrew Zisserman, William Freeman, and Alexei Efros. Unsupervised discovery of visual object class hierarchies. In *CVPR*, 2008.

[43] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised cross-domain image generation. *ICLR*, 2017.

[44] J. Tenenbaum and W. Freeman. Separating style and content with bilinear models. *Neural Computation*, 2000.

[45] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, 2011.

[46] Yaming Wang, Vlad I Morariu, and Larry S Davis. Weakly-supervised discriminative patch learning via cnn for fine-grained recognition. *CVPR*, 2018.

[47] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *ICML*, 2016.

[48] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *SIGIR*, 2003.

[49] Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. Attribute2image: Conditional image generation from visual attributes. In *ECCV*, 2016.

[50] Jianwei Yang, Anitha Kannan, Dhruv Batra, and Devi Parikh. Lr-gan: Layered recursive generative adversarial networks for image generation. *ICLR*, 2017.

[51] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *CVPR*, 2016.

[52] Bangpeng Yao, Aditya Khosla, and Li Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *CVPR*, 2011.

[53] Han Zhang, Tao Xu, Mohamed Elhoseiny, Xiaolei Huang, Shaoting Zhang, Ahmed Elgammal, and Dimitris Metaxas. Spda-cnn: Unifying semantic part detection and abstraction for fine-grained recognition. In *CVPR*, 2016.

[54] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *ICCV*, 2017.

[55] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *arXiv: 1710.10916*, 2017.

[56] Ning Zhang, Jeff Donahue, Ross Girshick, and Trevor Darrell. Part-based r-cnns for fine-grained category detection. In *ECCV*, 2014.

[57] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. *ICLR*, 2017.

[58] Heliang Zheng, Jianlong Fu, Tao Mei, and Jiebo Luo. Learning multi-attention convolutional neural network for fine-grained image recognition. In *ICCV*, 2017.

[59] Bohan Zhuang, Qi Wu, Chunhua Shen, Ian Reid, and Anton van den Hengel. Parallel attention: A unified framework for visual object discovery through dialogs and queries. In *CVPR*, 2018.