Collage with GAN

Stanford CS236G Final Project

Fenglu Hong
Department of Computer Science
Stanford University
hfenglu@stanford.edu

Abstract

Collaging is a creative process of forming visually balanced and aesthetic art work from pieces of images and materials. In this project, we apply state-of-the-art GAN models on a novel collage dataset for collage-like image generation. With mutual information maximization technique, we are able to obtain a latent space with meaningful disentanglement in factors of variations, such as colors, brightness, saturation, and partially control outputs with desired high-level features and layouts.

1 Introduction

Collage describe both technique and the resulting work of art in which pieces of paper, photographs, fabric and other ephemera are arranged and stuck down onto a supporting surface[1]. It is about rearranging patches of images with various textures, contents, and themes in an unexpected way to create a visually balanced and unified piece of art. The quality of the works often lays in colors, balance, and layouts. These are no strict rules in making collage and breaking the rules with creativity is what makes it intriguing to look at.

Recent advancements in the state-of-the-art Generative Adversarial Networks (GANs)[2] show their great potential for many real-life applications, including art creation, photo/video editing, and image translation. In this work, we explore and build different GAN models to create collage art works that are both creative and aesthetic. We deploy and train state-of-the-art GAN models on collage images collected from Instagram for collage generation. We also experiment with mutual information maximization for enforcing meaningful disentanglement in the latent space, and controlling outputs with desired high-level features and layouts.

2 Related Works

Collage. Variations of GAN models have shown great potential in art-related applications. Specifically, for collage-like task or approach, [3] proposes a composite GAN, which consists of several generators and a RNN model for disentangling complex factors of images and generating images part by part. The generated parts are then blended by an alpha blending process to create a new synthesized image. The model is evaluated on datasets such as CelebA dataset and Oxford 102 Flowers, which contain images with highly similar arrangement of components, such as a close-up of a face/flower in the middle of the background. Such unsupervised spatial disentanglement would not make sense on collage data, which does not have uniform layouts or components.

Derivative-Works[4] by Joel Simon and Tal Shiri tries to collage human face by cutting out patches from a source image and rearranging them. A patch generator (DCGAN[5]) is trained to generate a variety of shapes for cutting the source image. Each patch has a corresponding latent vector and transformation matrices, which determine where in the source image the patch is cut from and where on the blank canvas it is placed. Variables in the latent vectors and transformation matrices of a fixed number of patches are optimized to do feature inversion over a face classifier (DLIB[6]'s CNN model),

so that the output images resemble human faces. Again, it is hard to find such signal for learning the latent vectors and transformation matrices for generating a general collage, which assumes no target layouts of patches.

Another model proposed in [7] generates image collages of source templates using set-structured representations. The model takes in a sampled set of memory templates and a content image, predicts blending weights using U-Net with Set Transformer[8] blocks, and uses a permutation-invariant pooling layer to create the collage image. Like the previous work, this method also aims to produce a collage with a target high-level layout that looks like the specified content image.

Our approach generates the collage without a target layout or compositional steps, to more comprehensively model the distribution of the training data with various layouts and styles. To manipulate features in the generated output, we apply unsupervised learning of latent space disentanglement during training to discover factors of variations in the collage data.

Disentanglement. There exists a large body of work on unsupervised representation learning, whose goal is to learn a representation that displays salient semantic features as decodable factors in unlabelled data. Several approaches are built on the variational autoencoder [9] framework. βVAE introduces an adjustable hyperparameter β that balances latent channel capacity and independence constraints with reconstruction accuracy. FactorVAE[10] disentangles by encouraging the distribution of representations to be factorial and hence independent across the dimensions. As for recovering disentangled representations within GAN framework, InfoGAN[11] maximizes the mutual information between a small subset of the latent variables and the observation, by deriving a lower bound of the mutual information objective that can be efficiently optimized. Our appoach mainly follows the techniques used in InfoGAN model. To extent their work and better suit this task, we also use spatial information maximization that will be covered in later sections.

3 Datasets

A Dataset of 5,266 images is created for this task. 2,646 images are from all posts in Instagram account *"collage_expo"*[12] dated until Feb 20, 2021, except videos and 72 non-collage images (artist holding their artwork, plain text, etc.). The rest images are a selection of images from account *"thecollageclub"*[13]. Visual inspection is applied for filtering non-collage images and selecting high-quality collage from *"thecollageclub"*'s posts. All images are downloaded with python package *instaloader*[14]. We split the images with a train/dev/test ratio of 8:1:1, and train the models on the training set. Data preprocessing includes proportional resizing and center crop. Fig. 1 displays 36 training samples without preprocessing.

4 Approach

Baseline. We choose the light-weight GAN structure proposed in [15] as the baseline model, because of its low computational cost and high fidelity outputs, even with a small training dataset. The main contribution of light-weight GAN is threefold: the Skip-Layer channel-wise Excitation (SLE) module, a self-supervised discriminator as a feature-encoder, and a computational-efficient GAN model with the previous two techniques. SLE replaces the element-wise addition between the activation from different conv-layers in the Residual structure with channel-wise multiplications, which can realize skip-connection between resolutions with long range and improve gradient flow with no extra computational cost. Aside from the adversarial training, the model enforces a strong regularization on the discriminator, which is treated as a feature-encoder, and is jointly trained with a small decoder with reconstruction loss on real images only. This self-supervised auto-encoding training encourage the discriminator to extract a more holistic representation from the inputs and provide better signals for training the generator. We use official implementation from [16] for training a baseline model and building our improved models.

Disentanglement. To achieve partially controllable generation, we decide to train the generator with a well-disentangled latent space. Since our dataset is unlabeled, conditional generation is not suitable here. Another route is to regularize and disentangle the noise space after the generator is trained, using gradients from classifier or feature extractor to guide noise vectors. As there are no clear groups of different collage or many directly identifiable features, we would have to use other

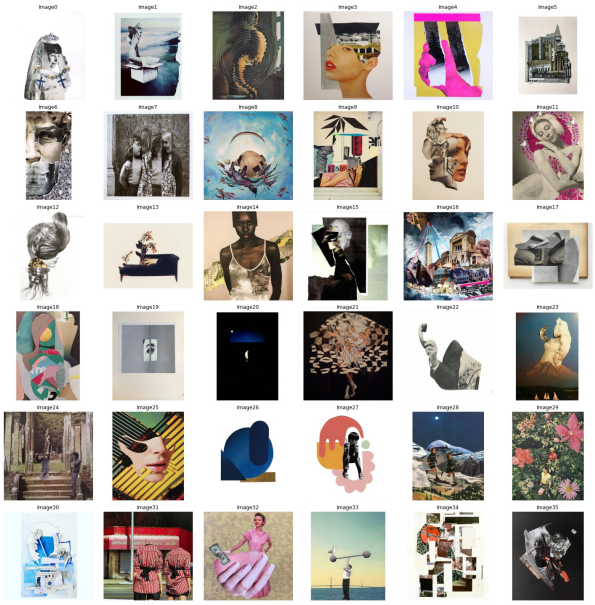

Figure 1: Training samples from Dataset II

unsupervised methods to extract features and learn latent codes to guide the gradient ascent for the noise vectors. However, training such models, such as a VAE, with well-disentangled and sufficiently informative latent space, is nontrivial and redundant. Thus, encouraging latent disentanglement while training the generator is the most straightforward and ideal option for our goal.

To enforce disentanglement so that a subset of the learned latent codes are interpretable and associated with distinct properties of the target distribution, we apply Variational Mutual Information Maximization as proposed in InfoGAN. We choose to model the latent code with a set of continuous variables, $c_i \sim \text{Unif}(-1, 1)$, since there is no obvious categorical distribution in the dataset. As described in InfoGAN model, we define an auxiliary distribution $Q(c|x)$ to approximate the true posterior distribution $P(c|x)$, where $x \sim G(z, c)$ and z denotes the noise vector. To maximize the mutual information $I(c; G(z, c))$, we maximize its variational lower bound, $L_I(G, Q)$, which is defined as follows:

$$L_I(G, Q) = \mathbb{E}_{x\sim G(z,c)}[\mathbb{E}_{c'\sim P(c|x)}[\log Q(c'|x)]] + H(c)$$

Denote the minimax game of GAN's generator and discriminator as $\min_G \max_D V(D, G)$, then with information regularization, the objective is as follows:

$$\min_{G,Q} \max_D V_I(D, G, Q) = V(D, G) - \lambda L_I(G, Q)$$

5 Experiment

Implementation. In the original light-weight GAN model, the task of the discriminator is three-fold. A big encoder E_{big} with SLE modules predicts a 5×5 matrix indicating patch-wise realness based on input images of target size (larger than 128^2 for all experiments). A small encoder E_{small} predicts a 5×5 matrix indicating realness based on input images that are down-sized to 128^2. Then, for the real images, three decoders should reconstruct the original images or one-fourth of the original images, given the the two 8×8 feature maps from E_{big} and E_{small}, or the 8×8 feature map that is one-fourth of the second-to-last intermediate feature map from E_{big}. Since we want the latent space to capture high-level features, we let Q and E_{small} share a large part of convolutional layers. Taken the 8×8 feature map from E_{small}, we use the network architecture summarized in Table 1 to predict the mean and log variance of the Gaussian distribution that is used to model $Q(c|x)$.

We use a learning rate of 2e-4 for D and G, and a learning rate of 1e-4 for Q. We use 8 continuous latent variables from Unif$(-1, 1)$ distribution, and assume a simple factored joint distribution, given by $P(c_1, \ldots, c_8) = \prod_{i=1}^{8} P(c_i)$. Similar to InfoGAN's experiments, we treat $Q(c_j|x)$ as a factored Gaussian distribution.

Input $256 \times 8 \times 8$
Conv2d(256, 256, 4, 2, 1, bias=False) - Batchnorm - LeakyReLU(0.2)
Conv2d(256, 128, 4, 1, 0, bias=False) - Batchnorm - LeakyReLU(0.2)
Conv2d(128, $2 \times$ latent_dim, 1, 1, 0, bias=False)

Table 1: The CNN architecture used for Q network

Risky Experiments Given the nature of collage data, it is desirable to separately control different part of the generated images, by disentangling and capturing the factors of variations independently in different locations. Thus, besides the original InfoGAN objective, we follow a PatchGAN[17] approach for an additional Q_{sp} network, design a spatial latent code, and enforce spatial mutual information maximization. Specifically, we predict $Q_{sp}(c_k|x_{\text{part k}}, k)$ to enforce each subset of the spatial latent code c_k to capture factors of variations within a corresponding patch of the image $x_{\text{part k}}$. Each ground-truth spatial latent code is sampled independently.

There are two versions of implementation:

- **Version 1**: For each fake sample, predict any $Q_{sp}(c_k|x_{\text{part k}})$ for $k \in \{0, 1, 2, 3\}$, where $x_{\text{part k}}$ is the receptive field of a 8×8 corner of the intermediate 16×16 feature maps from E_{big}. Q_{sp} and E_{big} share the convolutional layers until outputting the 16×16 feature maps.
- **Version 2**: For each fake sample, predict all $Q_{sp}(c_k|x_{\text{part k}})$ for $k \in \{0, 1, 2, 3\}$, where $x_{\text{part k}}$ is a 64×64 corner of the original image that is resized to 128×128, concatenated with a channel of the same-size padding of the index for that corner. The spatial mutual information loss is the average of the mutual information losses for the four corners.

For the two experiments, we use 4 continuous global latent variables. For each of the four corners, we also use 4 continuous latent variables with the same setup.

Metrics. To measure the models' synthesis performance, we use Frechet Inception Distance (FID)[18], which measures the overall semantic realism of the synthesized images. We let the trained generator generates 2048 images and compute FID between the synthesized images and the whole training/dev set.

Quantitative Results. Table 2 summarizes the setups and results of two experiments with baseline light-weight GANs, three experiments with the addition of the mutual information objective, and two experiments with the addition of spatial mutual information objective. We use a batch size of 16. As expected, FID scores increase with more regularization on latent space.

Qualitative Results. Fig. 2 present generated samples from Exp. 1 and Exp. 5. Though the two models have different in FID scores, there is not an obvious visual difference. Fig. 3, 4, and 5 display sample images or results of traversing the latent codes from these experiments. For the risky experiment (specifically Exp. 7), Version 2 of spatial information maximization achieves similar FID scores as the baseline model, while also showing some meaningful continuous variations in colors and brightness in the global latent space and the spatial latent space, as shown in Fig. 6 (left, middle). However, the model fails to achieve the desired effect; varying one spatial latent code still affects the entire images. Since it might be a issue with not having enough spatial information regularization, we also try increasing λ_{sp} for spatial information loss to 4 in Version 2 experiment. The model also fails as the spatial latent code stops to capture enough variations, as shown in Fig. 6 (right). The reason that this approach fails might be that there is no explicit penalty for disentangling different spatial latent codes, and there is a nature conflict between disentangling spatial latent codes and also preserving enough factors of variations in them. We need more regularization on spatial information so that changing one code will not affect unrelated parts of the image. However, this would diminish the capability of the spatial latent code to capture variations.

Exp. ID	Model	Res.	Steps	$\lambda(, \lambda_{sp})$	Train FID	Dev FID
1	Baseline	512^2	50k	N/A	85.44	129.11
2	Baseline	256^2	30k	N/A	88.27	131.02
3	w/ Info.	256^2	30k	0.1	89.34	135.01
4	w/ Info.	256^2	30k	1	100.50	141.54
5	w/ Info.	256^2	30k	10	103.08	143.98
6	w/ Sp. Info. v1	256^2	30k	1, 1*	112.66	154.12
7	w/ Sp. Info. v2	256^2	30k	1, 1	92.29	134.07

Table 2: Experiments with baseline light-weight GAN and light-weight GAN with mutual information maximization. *The two numbers correspond to the weight for the global mutual information loss and the spatial mutual information loss.

6 Analysis

As shown in Fig. 3, increasing λ enforces the latent space to capture more drastic and meaningful variation. Beside, for $\lambda = 0.1$, varying the noise vector with latent code fixed results in slightly different images, with similar overall layouts and colors. As λ increases, varying the noise vector with latent code fixed results in more identical images with very little variation. This tendency indicates that noise vector becomes almost futile in models with large λ, and the 8-dimensional latent space can still capture a decent amount of variations as displayed in Fig. 5 (right).

As shown by Fig. 4 (different latent code, different noise vector), a large enough λ will encourage a same type of variation in any dimension across samples from different noise vectors. Besides, a larger λ results in a more drastic continuous variation in each dimension of the latent space. The corresponding features include darkness, colorfulness, the amount of white space, etc.

All these comparisons indicate that, larger λ forces more variations in the data distribution to be encoded by the latent code; varying the noise vector leads to less changes in the output and noise vector becomes futile. As shown in Fig. 5, with $\lambda = 0.1$, a fixed noise vector leads to similar styles and contents (hue, texture, amount of white space, location of object) in the output with different latent code. However, as λ increases, a fixed noise vector leads to outputs that share less similarities in layouts and colors. Most sample images from $\lambda = 1$ case share a similar rectangular layout, and also display more variety in colors and textures. In $\lambda = 10$ case, it is hard to pinpoint common features from different samples.

We do not include disentanglement metrics in our report, since our data is not generated with any ground-truth labels, and there are no clear dimensions of variations for which we can hand-label our data. Our dataset is fundamentally different from many dataset used for disentanglement evaluation. Some widely tested datasets [11] [19], such as MNIST, CelebA face, 3D Chairs, dSprites, and SVHN, all have factors of variations on a single uniformed entity (such as a face, a digit, or a shape). As our dataset contains images with different components and various compositions, it is essentially hard for the model's latent space to capture variations other than the colors, saturation, darkness, and etc.

7 Conclusion

This paper introduces a method to generate collage-like images with controllable factors of variations. Concluding from the previous results and analysis, for our proposed model with λ around 1, we can generate a series of collage with similar high-level styles by fixing a random noise and sample different latent code. Then for a specific sample, we can also control its styles, such as darkness and the amount of white space, by varying a certain dimension of the latent code. Some future work can be generating photo-realistic details in the final outputs, or combining elements from source images with the generated high-level layouts.

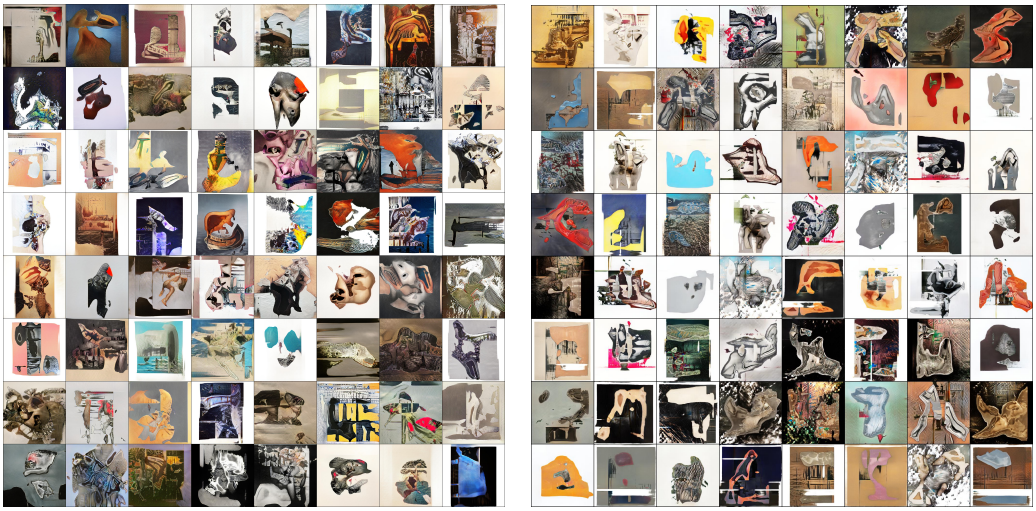

Figure 2: Sample images from generators in Exp. 1 (left) and Exp. 5 (right)

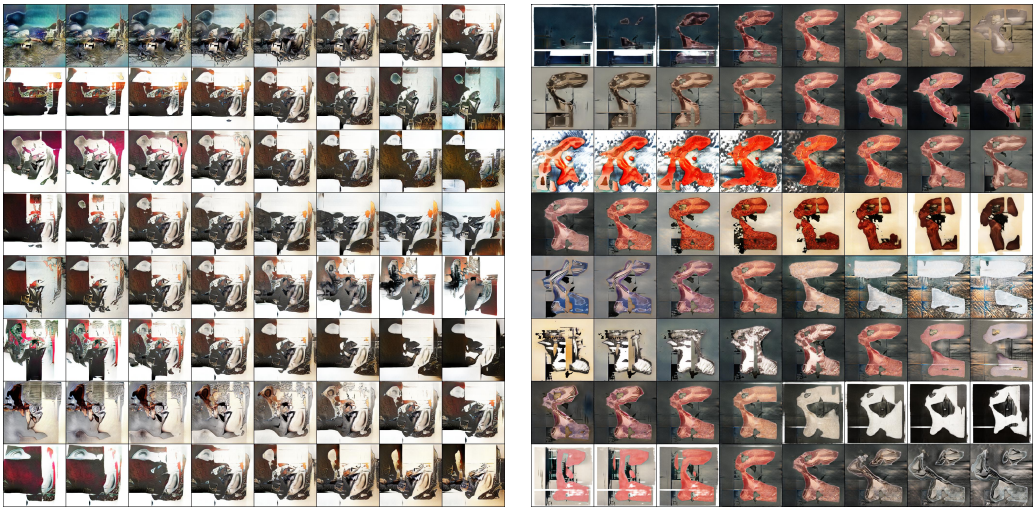

Figure 3: Latent traversal from generator in Exp. 3 (left) and Exp. 5 (right). For each set of 64 images, we fix a random noise z and a random latent code c. Then row i shows the outputs of varying the value of latent code c_i from -2 to 2.

References

[1] `https://www.tate.org.uk/art/art-terms/c/collage`.

[2] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[3] Hanock Kwak and Byoung-Tak Zhang. Generating images part by part with composite generative adversarial networks, 2016.

[4] `https://derivative.works/exhibits`.

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

[6] `https://github.com/davisking/dlib`.

[7] Nikolay Jetchev, Urs Bergmann, and Gökhan Yildirim. Transform the set: Memory attentive generation of guided and unguided image collages, 2019.

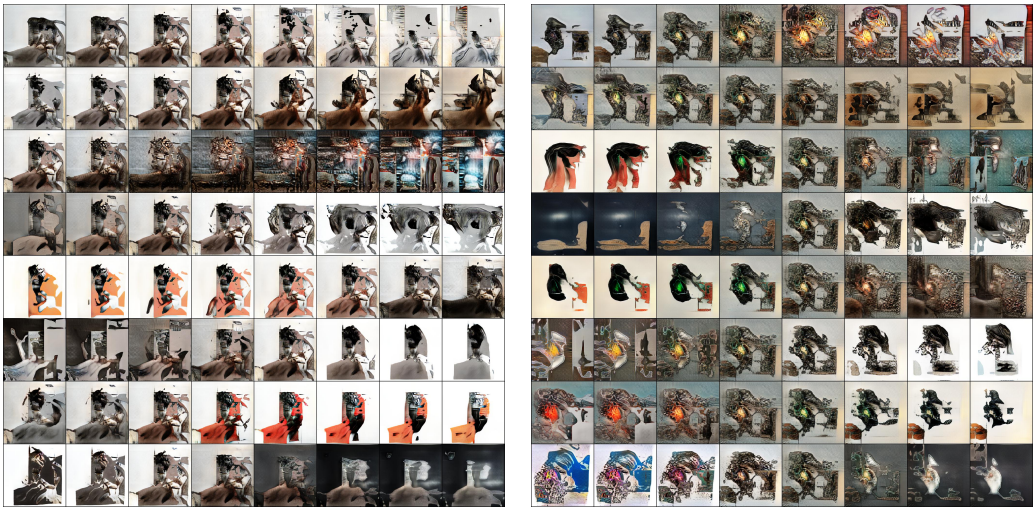

Figure 4: Latent traversal from generator in Exp. 4. For each set of 64 images, we fix a random noise z and a random latent code c. Then row i shows the outputs of varying the value of latent code c_i from -2 to 2.

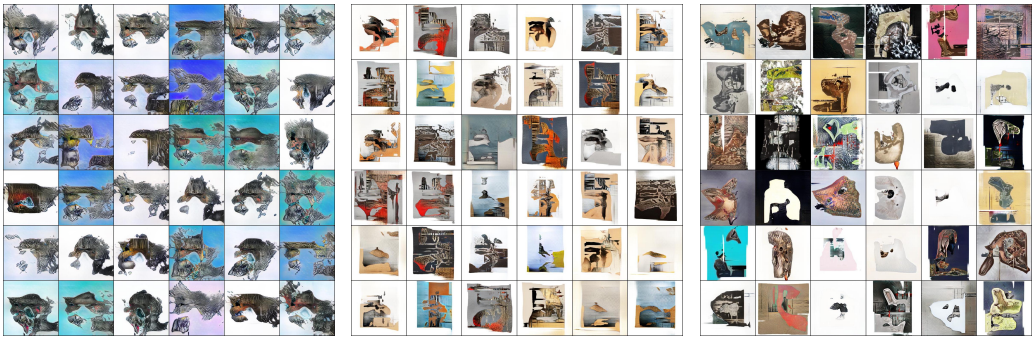

Figure 5: Sample images from generators in Exp. 3 (left), Exp. 4 (middle), and Exp. 5 (right). For each set of images, we fix a noise vector and randomly sample 36 latent codes.

Figure 6: Traversing the global latent space from generator in Exp. 7 (left); traversing the spatial latent space corresponding to the low right corner in Exp. 7 (middle); traversing the spatial latent space corresponding to the low right corner in model with same setup as Exp. 7, but with $\lambda_{sp} = 4$ (right). For each set of images, we fix a noise vector and latent codes.

[8] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks, 2019.

[9] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[10] Hyunjik Kim and Andriy Mnih. Disentangling by factorising, 2019.

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

[12] `https://www.instagram.com/collage_expo/`.

[13] `https://www.instagram.com/thecollageclub/`.

[14] `https://github.com/instaloader/instaloader`.

[15] Bingchen Liu, Yizhe Zhu, Kunpeng Song, and Ahmed Elgammal. Towards faster and stabilized gan training for high-fidelity few-shot image synthesis, 2021.

[16] `https://github.com/odegeasslbc/FastGAN-pytorch`.

[17] Chuan Li and Michael Wand. Combining markov random fields and convolutional neural networks for image synthesis, 2016.

[18] 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, 2018.

[19] Zinan Lin, Kiran Koshy Thekumparampil, Giulia Fanti, and Sewoong Oh. Infogan-cr and modelcentrality: Self-supervised model training and selection for disentangling gans, 2020.