

# SketchyGAN: Towards Diverse and Realistic Sketch to Image Synthesis

Wengling Chen

Georgia Institute of Technology

wchen342@gatech.edu

James Hays

Georgia Institute of Technology, Argo AI

hays@gatech.edu

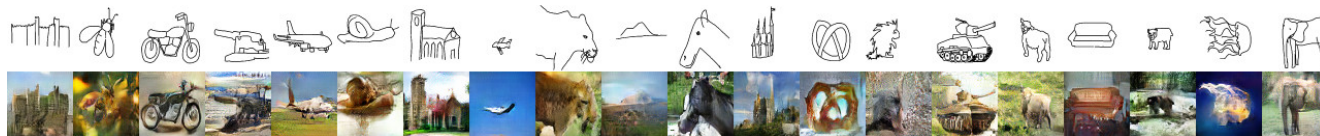


Figure 1: A sample of sketch-to-photo synthesis results from our 50 categories. Best viewed in color.

## Abstract

*Synthesizing realistic images from human drawn sketches is a challenging problem in computer graphics and vision. Existing approaches either need exact edge maps, or rely on retrieval of existing photographs. In this work, we propose a novel Generative Adversarial Network (GAN) approach that synthesizes plausible images from 50 categories including motorcycles, horses and couches. We demonstrate a data augmentation technique for sketches which is fully automatic, and we show that the augmented data is helpful to our task. We introduce a new network building block suitable for both the generator and discriminator which improves the information flow by injecting the input image at multiple scales. Compared to state-of-the-art image translation methods, our approach generates more realistic images and achieves significantly higher Inception Scores.<sup>1</sup>*

## 1. Introduction

How can we visualize a scene or object quickly? One of the easiest ways is to draw a sketch. Compared to photography, drawing a sketch does not require any capture devices and is not limited to faithfully sampling reality. However, sketches are often simple and imperfect, so it is challenging to synthesize realistic images from novice sketches. Sketch-based image synthesis enables non-artists to create realistic images without significant artistic skill or domain expertise in image synthesis. It is generally hard because sketches are sparse, and novice human artists cannot draw sketches that precisely reflect object boundaries. A real-looking image synthesized from a sketch should respect the intent of the artist *as much as possible*, but might need to deviate from

the coarse strokes in order to stay on the natural image manifold. In the past 30 years, the most popular sketch-based image synthesis techniques are driven by image retrieval methods such as Photosketcher [13] and Sketch2photo [5]. Such approaches often require carefully designed feature representations which are invariant between sketches and photos. They also involve complicated post-processing procedures like graph cut compositing and gradient domain blending in order to make the synthesized images realistic.

The recent emergence of deep convolutional neural networks [33, 32, 18] has provided enticing methods for image synthesis, among which Generative Adversarial Networks (GANs) [14] have shown great potential. A GAN frames its training as a zero-sum game between the generator and the discriminator. The goal of the discriminator is to decide whether a given image is fake or real, while the generator tries to generate realistic images so the discriminator will misclassify them as real. Sketch-based image synthesis can be formulated as an image translation problem conditioned on an input sketch. There exist several methods that use GANs to translate images from one domain to another [25, 62]. However, none of them is specifically designed for image synthesis from sketches.

In this paper, we propose SketchyGAN, a GAN-based, end-to-end trainable sketch to image synthesis approach that can generate objects from 50 classes. The input is a sketch illustrating an object and the output is a realistic image containing that object in a similar pose. This is challenging because: (i) paired photos and sketches are difficult to acquire so there is **no massive database to learn from**. (ii) There is **no established neural network method for sketch to image synthesis for diverse categories**. Previous works train models for single or few categories [28, 50].

We resolve the first challenge by **augmenting the Sketchy database** [49], which contains nearly 75,000 actual human sketches paired with photos, with a larger dataset of

<sup>1</sup>Code can be found at <https://github.com/wchen342/SketchyGAN>

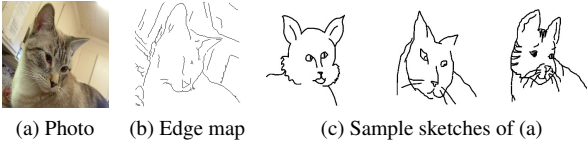


Figure 2: Comparison between an edge map and sketches of the same image. The photo and sketches are from the Sketchy Database. Compared to sketches, the edge map contains more background information. The sketches, in contrast, do not precisely reflect actual object boundaries and are not spatially aligned with the object.

paired *edge maps* and photos. This augmentation dataset is obtained by collecting 2,299,144 Flickr images from 50 categories and synthesizing edge maps from them. During training, we adjust the ratio between edge map-image and sketch-image pairs so that the network can transfer its knowledge gradually from edge-image synthesis to sketch-image synthesis. For the second challenge, we build a GAN-based model, conditioned on an input sketch, with several additional loss terms which improve synthesis quality. We also introduce a new building block called **Masked Residual Unit (MRU)** which helps generate higher quality images. This block **takes an extra image input and utilizes its internal mask to dynamically decide the information flow of the network**. By chaining these blocks we are able to input a pyramid of images at different scales. We show that this structure outperforms naive convolutional approaches and ResNet blocks on our sketch to image synthesis tasks. Our main contributions are:

- We present SketchyGAN, a deep learning approach to sketch to image synthesis. Unlike previous non-parametric approaches, we do not do image retrieval at test time. Unlike previous deep image translation methods, our network does not learn to directly copy input edges (effectively colorizing instead of converting sketches to photos). Our method is capable of generating plausible objects from 50 diverse categories. Sketch-based image synthesis is very challenging and our results are not generally *photorealistic*, but we demonstrate an increase in quality compared to existing deep generative models.
- We demonstrate a data augmentation technique for sketch data that address the lack of sufficient human-annotated training data.
- We formulate a GAN model with additional objective functions and a new network building block. We show that all of them are beneficial for our task, and lacking any of them will reduce the quality of our results.

## 2. Related Work

**Sketch-Based Image Retrieval and Synthesis.** There exist numerous works on sketch-based image retrieval [11, 12, 21, 2, 3, 55, 23, 22, 26, 54, 38, 56, 34]. Most methods use bag of words representations and edge detection to build features that are (ideally) invariant across both domains. Common shortcomings include the inability to perform fine-grained retrieval and the inability to map from badly drawn sketch edges to photo boundaries. To address these problems, Yu *et al.* [60] and Sangkloy *et al.* [49] train deep convolutional neural networks (CNNs) to relate sketches and photos, treating the sketch-based image retrieval as a search in the learned feature embedding space. They show that using CNNs greatly improves performance and they are able to do fine-grained and instance-level retrieval. Beyond the task of retrieval, Sketch2Photo [5] and PhotoSketcher [13] synthesize realistic images by compositing objects and backgrounds retrieved from a given sketch. PoseShop [6] composites images of people by letting users input an additional 2D skeleton into the query so that the retrieval will be more precise.

**Sketch-Based Datasets.** There are only a few datasets of human-drawn sketches and they are generally small due to the effort needed to collect drawings. One of the most commonly used sketch dataset is the TU-Berlin dataset [10] which contains 20,000 human sketches spanning 250 categories. Yu *et al.* [60] introduced a new dataset with paired sketches and images, but there are only two categories – shoes and chairs. There is also the CUHK Face Sketches [57] containing 606 face sketches drawn by artists. The newly published QuickDraw dataset [16] has an impressive 50 million sketches. However, the sketches are particularly crude because of a 10 second time limit. The sketches lack detail and tend to be iconic or canonical views. The Sketchy database [49], in contrast, has more detailed drawings in a greater variety of poses. It spans 125 categories with a total of 75,471 sketches of 12,500 objects. Critically, it is the only substantial dataset of *paired* sketches and photographs spanning diverse categories so we choose to use this dataset.

**Image-to-Image Translation with GANs.** Generative Adversarial Networks (GANs) have shown great potential in generating natural, realistic images [15, 43]. Instead of directly optimizing per pixel reconstruction error, which often leads to blurry and conservative results, GANs use a discriminator to distinguish unrealistic images from real ones thus forcing the generator to produce sharper images. The “pix2pix” work of Isola *et al.* [25] demonstrates a straightforward approach to translate one image to another using conditional GANs. Conditional settings are also adapted in other image translation tasks, including sketch coloring [50], style transformation [59] and domain adaptation [1] tasks. In contrast with using conditional GANs and paired data, Liu *et al.* [39] introduce an unsupervised image trans-

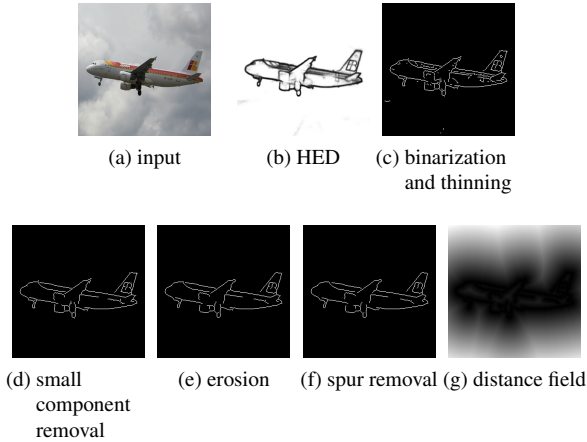


Figure 3: Pipeline of edge map creation. Images from intermediate steps show that each step helps remove some artifacts and make the edge maps more sketch-like.

lation framework consists of CoupledGAN [40] and a pair of variational autoencoders [30]. More recently, CycleGAN [62] shows promising results on unsupervised image translation by enforcing cycle-consistency losses.

### 3. Sketchy Database Augmentation

In this section, we discuss how we augment the Sketchy database [49] with Flickr images and synthesize edge maps which we hope approximate human sketches. The dataset is publicly available. Section 3.2 describes image collection, image content filtering, and category selection. Section 3.3 describes our edge map synthesis. Section 3.4 describes the way we use the augmented dataset.

#### 3.1. Edges vs Sketches

Figure 2 visualizes the difference between image *edges* and *sketches*. A sketch is set of human-drawn strokes mimicking the approximate boundary and internal contours of an object, and an edge map is machine-generated array of pixels that precisely correspond to photo intensity boundaries. Generating photos from *sketches* is considerably harder than from *edges*. Unlike edge maps, sketches are not precisely aligned to object boundaries, so a generative model needs to learn spatial transformations to correct deformed strokes. Second, edge maps usually contain more information about backgrounds and details, while sketches do not, so a generative model must insert more information itself. Finally, sketches may contain caricatured or iconic features, like the “tiger” stripes on the cat’s face in Figure 2c, which a model must learn to handle. Despite these considerable differences, edge maps are still a valuable augmentation to the limited Sketchy database.



Figure 4: Images synthesized from the same input sketch with different noise vectors. The network learned to change a significant portion of the image (the flower), which is not conditioned by the input sketch. In each case, the bee remains plausible.

#### 3.2. Data Collection

Learning the mapping between edges or sketches to photos requires significant training data. We want thousands of images per category. ImageNet only has around 1,000 images per class, and photos in COCO tend to be cluttered and thus not ideal as object sketch exemplars. Ideally we want photographs with one dominant object as is the case for the Sketchy database photographs. Accordingly, we collect images directly from Flickr through the Flickr API by querying category names as keywords. 100,000 images are gathered for each category, sorted by “relevance”. Two different models are used for filtering out unrelated images. We use an Inception-ResNet-v2 network [52] to filter images from the 38 ImageNet [47] categories that overlap with Sketchy, and a Single Shot MultiBox Detector [41] to detect whether an image contains an object in the 18 COCO [37] categories that overlap with Sketchy. For SSD, the bounding box of a detected object must cover more than 5% of the image area or the image is discarded. After filtering, we obtain a dataset with an average of 46,265 images per ImageNet category and 61,365 images per COCO category. For the remainder of the paper, we use 50 out of the 56 available categories after excluding six categories that often have a human as a main object. The excluded classes are harp, violin, umbrella, saxophone, racket, and trumpet.

#### 3.3. Edge Map Creation

We use edge detection and several post-processing steps to obtain sketch-like edge maps. The pipeline is illustrated in Figure 3. The first step is to **detect edges with Holistically-nested edge detection (HED)** [58] as in Isola *et al.* [25]. After **binarizing the output and thinning all edges** [61], we clean isolated pixels and **remove small connected components**. Next we perform **erosion** with a threshold on all edges, further decreasing number of edge fragments. **Remaining spurs are then removed**. Because edges are very sparse, we **calculate an unsigned euclidean distance field for each edge map to obtain a dense representation** (see Figure 3g). Similar distance-field representations are used in recent works on 3D shape recovery [53, 17]. We also calculate distance fields for sketches in the Sketchy database.

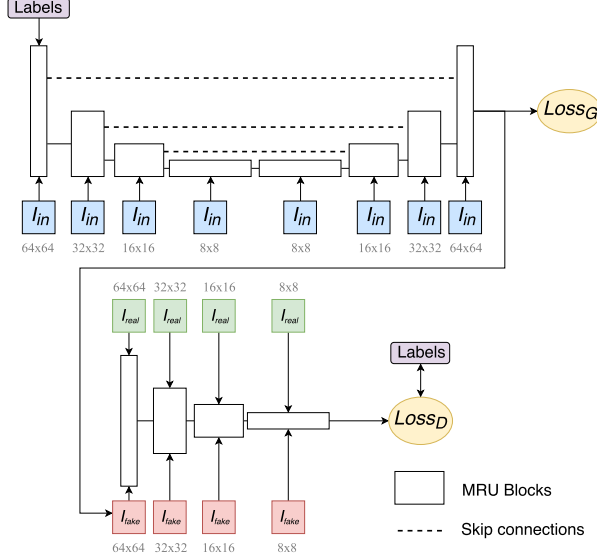


Figure 5: Complete structure of our network. Since we are using MRU blocks, both the generator and the discriminator can take multi-scale inputs.

### 3.4. Training Adaptation from Edges to Sketches

Because our final goal is a network that generates images from sketches, it is necessary to train the network on both edge maps and sketches. To simplify training process, we use a strategy that gradually shifts the inputs from edge maps to sketches: at the beginning of training, the training data are mostly pairs of images and edge maps. During training, we slowly increase the proportion of sketch-image pairs. Let  $i_{max}$  be the maximum number of training iterations,  $i_{cur}$  be the number of current iteration, then the proportion of sketches and edge maps at current iteration is given by:

$$P_{sk} = 0.1 + \min(0.8, (\frac{i_{cur}}{i_{max}})^\lambda) \quad (1)$$

$$P_{edge} = 1 - P_{sk} \quad (2)$$

respectively, where  $\lambda$  is an adjustable hyperparameter indicating how fast the portion of sketches grows. We use  $\lambda = 1$  in our experiments. It is easy to see that  $P_{sk}$  grows from 0.1 slowly to 0.9. Using this training schedule, we eliminate the need of separate pre-training on edge maps, so the whole training process is unified. We compare this method to training on edge maps first then fine-tuning on sketches. We find that discrete pre-training and then fine-tuning leads to lower inception scores on the test set compared to a gradual ramp from edges to sketches (6.73 vs 7.90).

## 4. SketchyGAN

In this section we present a Generative Adversarial Network framework that transforms input sketches into images.

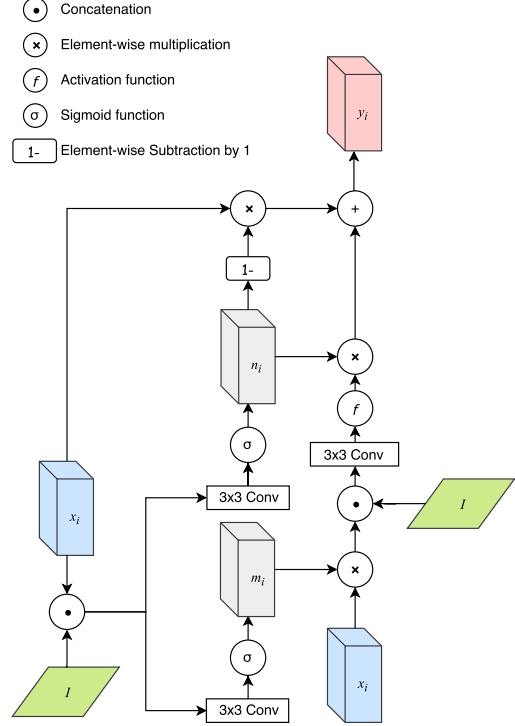


Figure 6: Structure of a Masked Residual Unit (MRU). It takes in feature maps  $x_i$  and an extra image  $I$ , then outputs new feature maps  $y_i$ .

Our GAN learns a mapping from an input sketch  $x$  to an output image  $y$ , so that  $G : x \rightarrow y$ . The GAN has two parts, a generator  $G$  and a discriminator  $D$ . Section 4.1 introduces the Masked Residual Unit (MRU), Section 4.2 illustrates the network structure, and Section 4.3 discusses the objective functions.

### 4.1. Masked Residual Unit (MRU)

We introduce a network module which allows a ConvNet to be repeatedly conditioned on an input image. The module uses a learned internal mask to selectively extract new features from the input images to combine with feature maps computed by the network thus far. We call this module the *Masked Residual Unit* or **MRU**.

Figure 6 shows the structure of Masked Residual Unit (MRU). Qualitative and quantitative comparison to DC-GAN [46] and ResNet generative architectures can be found in Section 5.3. An MRU block takes two inputs: input feature maps  $x_i$  and an image  $I$ , and outputs feature maps  $y_i$ . For convenience we only discuss the case in which inputs and outputs have the same spacial dimension. Let  $[\cdot, \cdot]$  denote concatenation,  $Conv(x)$  denote convolution on  $x$ , and  $f(x)$  be an activation function. We want to first merge the information in input image  $I$  into input feature maps  $x_i$ . A naive approach will be concatenating them along the feature



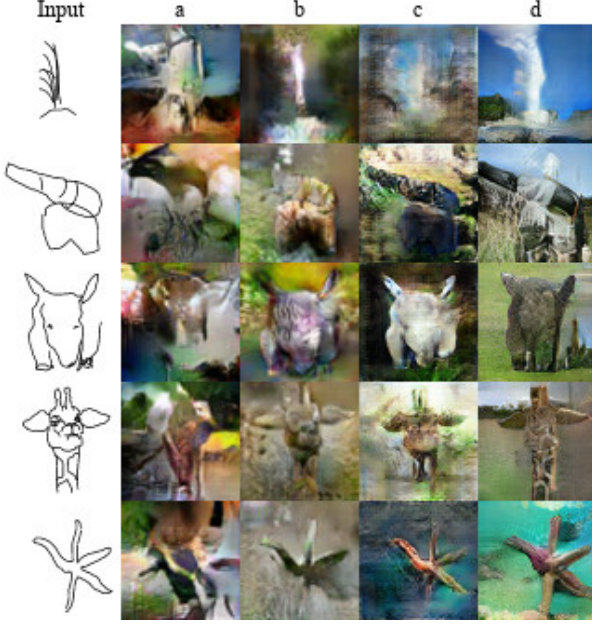


Figure 7: Image generated by pix2pix variations and our method. The four columns labeled by *a* to *d* are: (a) pix2pix on Sketchy (b) pix2pix on Augmented Sketchy (c) Label-supervised pix2pix on Augmented Sketchy and (d) our method. Comparing to our method, pix2pix results are blurry and noisy, often containing color patches and unwanted artifacts.

depth dimension and performing convolution:

$$z_i = f(\text{Conv}([x_i, I])) \quad (3)$$

However it is better if the block can decide how much information it wants to preserve upon receiving the new image. So instead we use the following approach:

$$z_i = f(\text{Conv}([m_i \odot x_i, I])) \quad (4)$$

where

$$m_i = \sigma(\text{Conv}([x_i, I])) \quad (5)$$

is a mask over the input feature maps. Multiple convolutional layers can be stacked here to increase performance. We then want to dynamically combine the information from the newly convolved feature maps and the original input feature maps, so we use another mask

$$n_i = \sigma(\text{Conv}([x_i, I])) \quad (6)$$

to combine the input feature maps with the new feature maps to get the final output:

$$y_i = (1 - n_i) \odot z_i + n_i \odot x_i \quad (7)$$

The second term in Equation 7 serves as a residual connection. Because there are internal masks to determine information flow, we call this structure masked residual unit. We can stack multiple of these units and input the same image at different scales repetitively so that the network can re-

Model	Inception Score
pix2pix, Sketchy only	3.94
pix2pix, Augmented	4.53
pix2pix, Augmented+Label	5.49
<b>Ours</b>	<b>7.90</b>
Real Image	15.46

Table 1: Comparison of our method to baseline methods. We compared to three variants of pix2pix, and our method shows a much higher score on test images.

trieve information from the input image dynamically on its computation path.

The MRU formulation is similar to that of the Gated Recurrent Unit (GRU) [7]. However, we are driven by different motivations and there are several crucial differences: 1) We are motivated by repetitively inputting the same image to improve the information flow. GRU is designed to address vanishing gradients in recurrent neural networks. 2) GRU cells are recurrent so part of the output is fed back into the same cell, while MRU blocks are cascaded so the outputs of a previous block are fed into the next block. 3) GRU shares weights for each step so it can only receive fixed length inputs. No two MRU blocks share weights, so we can shrink or expand the size of output feature maps like normal convolutional layers.

## 4.2. Network Structure

Our complete network structure is shown in Figure 5. The generator uses an encoder-decoder structure. Both the encoder and the decoder are built with MRU blocks, where the sketches are resized and fed into every MRU block on the path. In our best results in Figure 9, we also apply skip-connections between encoder and decoder blocks, so the output feature maps from encoder blocks will be concatenated to the outputs of corresponding decoder blocks. The discriminator is also built with MRU blocks but will shrink in spatial dimension. At the end of the discriminator, we output two logits, one for the GAN loss and one for classification loss.

## 4.3. Objective Function

Let  $x$ ,  $y$  be either an image or a sketch,  $z$  be a noise vector, and  $c$  be a class label, Our GAN objective function can be expressed as

$$\mathcal{L}_{GAN}(D, G) = \mathbb{E}_{y \sim P_{image}} [\log D(y)] + \mathbb{E}_{x \sim P_{sketch}, z \sim P_z} [\log(1 - D(G(x, z)))] \quad (8)$$

and the objective of generator  $\mathcal{L}_{GAN}(G)$  will be to minimize the second term.

It is shown that giving the model side information will

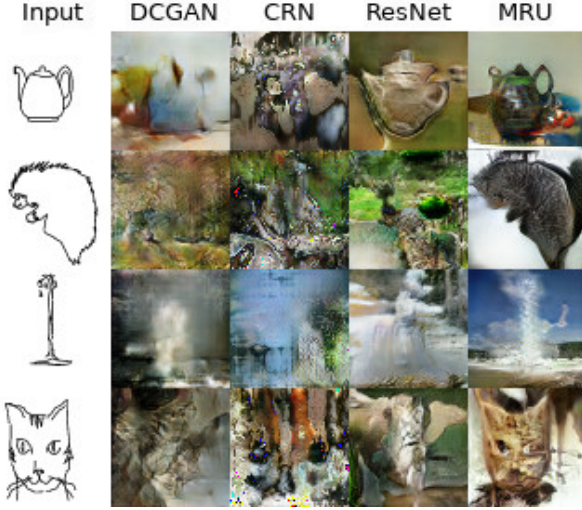


Figure 8: Visual results from DCGAN, CRN, ResNet and MRU. The MRU structure emphasize more on the main object than the other three.

improve the quality of generated images [44], so we use conditional instance normalization [9] in the generator and pass in labels of input sketches. In addition, we let the discriminator predict class labels out of the images it sees. The auxiliary classification loss of discriminator maximize the log-likelihood between predicted and ground-truth labels:

$$\mathcal{L}_{ac}(D) = \mathbb{E}[\log P(C = c|y)] \quad (9)$$

and the generator maximizes the same log-likelihood  $\mathcal{L}_{ac}(G) = \mathcal{L}_{ac}(D)$  with discriminator fixed.

Since we have paired image data, we are able to provide direct supervision to the network with L1-distance between generated images and ground truth images:

$$\mathcal{L}_{sup}(G) = \|G(x, z) - y\|_1 \quad (10)$$

However, directly minimizing L1 loss between generated image and ground truth image discourages diversity, so we add a perceptual loss to encourage the network to generate diverse images [8, 27, 4]. We use four intermediate layers from an Inception-V4 [52] to calculate the perceptual loss. Let  $\phi_i$  be the filter response of a layer in the Inception model. We define perceptual loss on the generator as:

$$\mathcal{L}_p(G) = \sum_i \lambda_p \|\phi_i(G(x, z)) - \phi_i(y)\|_1 \quad (11)$$

To further encourage diversity, we concatenate Gaussian noise to feature maps at the bottleneck of the generator. Previous works reach the conclusion that conditional GANs tend to ignore the noise completely [25] or produce worse results because of noise [45]. A simple diversity loss

$$\mathcal{L}_{div}(G) = -\lambda_{div} \|G(x, z_1) - G(x, z_2)\|_1 \quad (12)$$

will improve both quality and diversity of generated images. The interpretation is straightforward: with a pair of different

Model	Num of params	Inception Score
DCGAN	$G:35.1M D: 4.3M$	4.73
CRN	$G:21.4M D:22.3M$	4.56
Improved ResNet	$G:33.0M D:31.2M$	5.76
MRU (GAN loss only)	$G:28.1M D:29.9M$	8.31
MRU	$G:28.1M D:29.9M$	7.90

Table 2: Comparison of MRU, CRN, ResNet and DCGAN under the same setting. DCGAN structure is included for completeness. Under similar number of parameters, MRU outperforms ResNet block significantly on our generative task.

noise vectors  $z_1$  and  $z_2$  conditioned on the same image, the generator should output a pair of slightly different images.

Our complete discriminator and generator losses are thus

$$\mathcal{L}(D) = \mathcal{L}_{GAN}(D, G) + \mathcal{L}_{ac}(D) \quad (13)$$

$$\mathcal{L}(G) = \mathcal{L}_{GAN}(G) - \mathcal{L}_{ac}(G) + \mathcal{L}_{sup}(G) + \mathcal{L}_p(G) + \mathcal{L}_{div}(G) \quad (14)$$

where the discriminator maximizes Equation 13 and the generator minimizes Equation 14. In practice, we use DRAGAN loss [31] in order to stabilize training and use focal loss [36] as classification loss.

## 5. Experiments

### 5.1. Experiment settings

**Dataset splitting** We use the sketch-image pairs in selected 50 categories from training split of Sketchy as basic training data, and augment them with edge map-image pairs. In the following sections, we call data from Sketchy Database “Sketchy”, and Sketchy augmented with edge maps “Augmented Sketchy”. Since we are only interested in sketch to image synthesis, all models are tested on the test split of Sketchy. All images are resized to  $64 \times 64$  regardless of the original aspect ratio. Both sketches and edge maps are converted into distance fields.

**Implementation Details** In all experiments, we use batch size of 8, except for Figure 9 which uses a batch size of 32. We use random horizontal flipping during training. We use the Adam optimizer [29], and set the initial learning rate of generator at 0.0001 and that of discriminator at 0.0002 [20].

**Evaluation Metrics** For our task of image synthesis, we use Inception Scores [48] to measure the quality of synthesized images. The intuition behind Inception Score is that a good synthesized image should have easily recognizable objects by an off-the-shelf recognition system. Beyond Inception Scores, we also perform a perceptual study evaluating how realistic the generated images are and how faithful they are to the input sketches.

Model	Input correctly identified?
Sketchy 1-NN retrieval	35.3%
pix2pix, Augmented+Label	65.9%
Ours	47.4%

Table 3: Faithfulness test on three models. Models for which participants could pick the input sketch are considered more “faithful”.

Model	Picked as more realistic?
pix2pix, Sketchy only	6.03%
pix2pix, Augmented	18.4%
pix2pix, Augmented+Label	21.8%
Ours	53.7%

Table 4: Realism test on four generative models. We report how often results from each model were chosen by participants to be more “realistic” than a competing model.

## 5.2. Comparison to Baselines

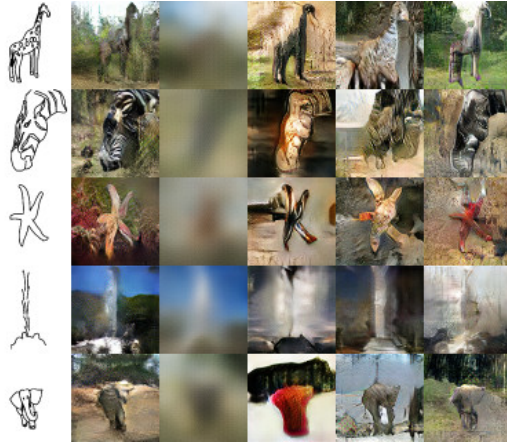
Our comparisons focus on the popular pix2pix and its variations. All models are trained for 300k iterations except for the first model. We include three baselines:

**pix2pix on Sketchy** This is the simplest model. We directly take the authors’ pix2pix code and train it on the 50 categories from Sketchy. Since we find the image quality stops improving after 100k iterations, we stop early at 150k iteration and report the results.

**pix2pix on Augmented Sketchy** In this model, we train pix2pix on both the image-edge map and image-sketch pairs, as we do in our method. The network structure and loss functions remain unchanged.

**Label-Supervised pix2pix on Augmented Sketchy** In this model, we modify pix2pix to pass class labels into the generator using conditional instance normalization, and also add auxiliary classification loss to its discriminator. This is a much stronger baseline, since the label information helps the network decide the object type and in turn improves the generated image quality [15, 44].

The comparison of Inception Scores can be found in Table 1 and visual results can be found in Figure 7. Our observations are as follows: (i) pix2pix trained on Sketchy fails, generating unidentifiable color patches. The model is unable to translate from sketches to images. Since pix2pix has been successful with edge-to-image translations, this implies that sketch-to-image synthesis is more difficult. (ii) pix2pix trained on Augmented Sketchy performs slightly better, starting to produce the general shape of the object. This shows that edge maps help the training. (iii) The label-supervised pix2pix on Augmented Sketchy is better than the previous two baselines. It correctly colors the object more often and starts to generate some meaningful backgrounds. The results are still blurry, and many artifacts can be observed. (iv) Comparing to baselines, our method generates sharper images, gets the object color correct, puts more de-



Input	Full	-GAN	-L-AC	-P	-DIV
None	7.90	1.49	6.64	6.70	7.29

Table 5: Table of Inception scores for models with particular components removed. “Full” is the full model described in this work. “-GAN” means no GAN loss and no discriminator. “-L-AC” means no labels-supervision on generator and no auxiliary loss on discriminator. “-P” means no L1 and no perceptual loss, and “-DIV” means no diversity loss.

tailed textures on the object, and outputs meaningful backgrounds. The whole images are also more colorful.

## 5.3. Component Analysis

Here we analyze which part of our model is more important. We decouple our objective function and analyze the influence of each part of it. All models are trained on Augmented Sketchy with the same set of parameters. Detailed comparison can be found in Table 5. We first remove the GAN loss and the discriminator. The result is surprisingly poor as the images are extremely vague. This observation is consistent with that of Isola *et al.* [25]. Next we remove the auxiliary loss and substitute conditional instance normalization with batch normalization [24]. This leads to a significant decrease in image quality as well as wrong colors and misplaced textures. This indicates that class information helps a lot, which makes sense because we are generating 50 categories from a single model. We then remove the L1 loss and the perceptual loss. We find they also have a large impact on image quality. From sample images we can see the model uses incorrect colors and fails and object boundaries are unrealistic or missing. Finally, we remove the diversity loss, and doing so also decreases image quality slightly. This can be related to how we apply this diversity loss, which forces the generator to generate image pairs that are realistic but different. This encourages generalization because the generator needs to find a solution that when given different noise vectors only makes changes in unconstrained areas (e.g. the background).



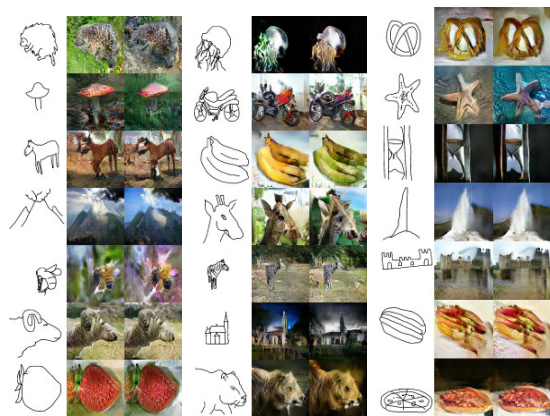


Figure 9: Some of the best output images from our full model. For each input sketch, we show a pair of output images to demonstrate the diversity of our model.

**Comparison between MRU and other structures** To demonstrate the effectiveness of our MRU blocks, we compare the performance of MRU, ResNet, Cascaded Refinement Network (CRN) [4] and DCGAN structures in our image synthesis task. We train several additional models: one uses improved ResNet blocks [19], which is the best variant published [18], in both generator and discriminator; one is a weak baseline, using DCGAN structure; one uses CRN in generator instead of MRU; and one MRU model using only GAN loss and ACGAN loss. We keep the number of parameters of MRU model and that of ResNet model roughly the same by reducing feature depth in MRU. Detailed parameter counts can be found in Table 2. Judging from both visual quality and the Inception Scores, the MRU model generates better images than both ResNet and CRN models, and we show that even using only standard GAN losses, MRU outperforms other structures significantly. From Figure 8, we notice that the MRU model tends to produce higher quality foreground objects. This can be due to the internal masks of MRU serving as an attention mechanism, causing the network to selectively focus on the main object. In our task this is helpful, since we are mainly interested in generating a specific object from sketch.

#### 5.4. Human Evaluation of Realism and Faithfulness

We do two human evaluations to measure how our model compares against baselines in terms of realism and faithfulness to the input sketch. In the “faithfulness” test, a participant sees the output of either pix2pix, SketchyGAN or 1-nearest-neighbor retrieval using the representation learned in the Sketchy Database [49]. With each image, the participant also sees 9 random sketches of the same category, one of which is the actual input/query sketch. The participant is asked to pick the sketch that prompted the output image. We then count how often participants pick the correct input

sketch, so a higher correct selection rate indicates the model produces a more “faithful” output. In the “realism” test, a participant sees the output of pix2pix variants and SketchyGAN compared in pairs, alongside the corresponding input sketch. The participant is asked to pick the image that they think is more realistic. For each model we calculate how often participants think it is more realistic. The image retrieval baseline is not evaluated for realism since it only returns existing, realistic photographs. We conducted 696 trials for the “faithfulness” test and 348 trials for the “realism” test. The results show that SketchyGAN is more faithful than the retrieval model, but is less faithful than pix2pix which often preserves the input edges precisely (Table 3). Meanwhile, SketchyGAN is considered more realistic than pix2pix variants (Table 4). The results are consistent with our goal that our model should respect the intent of input sketches, but at the same time deviate from the strokes if necessary in order to produce realistic images.

## 6. Conclusion

In this work, we presented a novel approach to the sketch-to-image synthesis problem. The problem is challenging given the nature of sketches, and this introduced a deep generative model that is promising in sketch to image synthesis. We introduced a data augmentation technique for sketch-image pairs to encourage research in this direction. The demonstrated GAN framework can synthesize more realistic images than popular generative models, and the generated images are diverse. Currently, the main focus on GANs is to find better probability metrics as objective functions, but there has been very few works searching for better network structures in GANs. We proposed a new network structure for our generative task, and we showed that it performs better than existing structures.

**Limitations.** Ideally, we want our results to be both realistic *and* faithful to the **intent** of the input sketch. For many sketches, we fail to meet one or both of these goals. Results generally aren’t photorealistic, nor are they high enough resolution. Sometimes realism is lost by being *overly* faithful to the sketch – e.g. Skinny horse legs that too closely follow the badly drawn input boundaries (Figure 9). In other cases, we do deviate from the user sketch to make the output more realistic (motorcycle and plane in Figure 1, mushroom, church, geyser, and castle in Figure 9) but still respect the pose and position of the object in the input sketch. This is more desirable. Human **intent** is hard to learn, and SketchyGAN failures that treat the input sketch too literally may be due to lack of sketch-photo training pairs. Despite the fact that our results are not yet *photorealistic*, we think they show a substantial improvement over previous methods.

**Acknowledgements.** This work was funded by NSF award 1561968.



## References

- [1] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [2] Y. Cao, C. Wang, L. Zhang, and L. Zhang. Edgel index for large-scale sketch-based image search. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 761–768. IEEE, 2011.
- [3] Y. Cao, H. Wang, C. Wang, Z. Li, L. Zhang, and L. Zhang. Mindfinder: interactive sketch-based image search on millions of images. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1605–1608. ACM, 2010.
- [4] Q. Chen and V. Koltun. Photographic image synthesis with cascaded refinement networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [5] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu. Sketch2photo: Internet image montage. *ACM Transactions on Graphics (TOG)*, 28(5):124, 2009.
- [6] T. Chen, P. Tan, L.-Q. Ma, M.-M. Cheng, A. Shamir, and S.-M. Hu. Poseshop: Human image database construction and personalized content synthesis. *IEEE Transactions on Visualization and Computer Graphics*, 19(5):824–837, 2013.
- [7] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.
- [8] A. Dosovitskiy and T. Brox. Generating images with perceptual similarity metrics based on deep networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, pages 658–666. Curran Associates, Inc., 2016.
- [9] V. Dumoulin, J. Shlens, and M. Kudlur. A learned representation for artistic style. *ICLR*, 2017.
- [10] M. Eitz, J. Hays, and M. Alexa. How do humans sketch objects? *ACM Transactions on Graphics (proceedings of SIGGRAPH)*, 31(4):44:1–44:10, 2012.
- [11] M. Eitz, K. Hildebrand, T. Boubekur, and M. Alexa. An evaluation of descriptors for large-scale image retrieval from sketched feature lines. *Computers & Graphics*, 34(5):482–498, 2010.
- [12] M. Eitz, K. Hildebrand, T. Boubekur, and M. Alexa. Sketch-based image retrieval: Benchmark and bag-of-features descriptors. *IEEE transactions on visualization and computer graphics*, 17(11):1624–1636, 2011.
- [13] M. Eitz, R. Richter, K. Hildebrand, T. Boubekur, and M. Alexa. Photosketcher: Interactive sketch-based image synthesis. *IEEE Computer Graphics and Applications*, 31(6):56–66, Nov 2011.
- [14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680. 2014.
- [15] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5769–5779, 2017.
- [16] D. Ha and D. Eck. A neural representation of sketch drawings. *arXiv preprint arXiv:1704.03477*, 2017.
- [17] X. Han, Z. Li, H. Huang, E. Kalogerakis, and Y. Yu. High-resolution shape completion using deep neural networks for global structure and local geometry inference. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [19] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645, 2016.
- [20] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6629–6640, 2017.
- [21] R. Hu, M. Barnard, and J. Collomosse. Gradient field descriptor for sketch based retrieval and localization. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 1025–1028. IEEE, 2010.
- [22] R. Hu and J. Collomosse. A performance evaluation of gradient field hog descriptor for sketch based image retrieval. *Computer Vision and Image Understanding*, 117(7):790–806, 2013.
- [23] R. Hu, T. Wang, and J. Collomosse. A bag-of-regions approach to sketch-based image retrieval. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 3661–3664. IEEE, 2011.
- [24] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.
- [25] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [26] S. James, M. J. Fonseca, and J. Collomosse. Reenact: Sketch based choreographic design from archival dance footage. In *Proceedings of International Conference on Multimedia Retrieval*, page 313. ACM, 2014.
- [27] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016.
- [28] T. Kim, M. Cha, H. Kim, J. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192*, 2017.
- [29] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [30] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *ICLR*, Apr. 2014.
- [31] N. Kodali, J. Abernethy, J. Hays, and Z. Kira. How to train your dragan. *arXiv preprint arXiv:1705.07215*, 2017.

- [32] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105. Curran Associates, Inc., 2012.
- [33] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
- [34] K. Li, K. Pang, Y. Z. Song, T. Hospedales, H. Zhang, and Y. Hu. Fine-grained sketch-based image retrieval: The role of part-aware attributes. In *The IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9, March 2016.
- [35] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.
- [36] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. Focal loss for dense object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [37] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.
- [38] Y.-L. Lin, C.-Y. Huang, H.-J. Wang, and W. Hsu. 3d subquery expansion for improving sketch-based multi-view image retrieval. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.
- [39] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*, pages 700–708, 2017.
- [40] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 469–477, 2016.
- [41] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *European Conference on Computer Vision*, pages 21–37, 2016.
- [42] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.
- [43] A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski. Plug & play generative networks: Conditional iterative generation of images in latent space. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [44] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier GANs. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2642–2651, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.
- [45] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [46] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
- [47] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.
- [48] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.
- [49] P. Sangkloy, N. Burnell, C. Ham, and J. Hays. The sketchy database: Learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics (proceedings of SIGGRAPH)*, 2016.
- [50] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays. Scribbler: Controlling deep image synthesis with sketch and color. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2017.
- [51] R. K. Srivastava, K. Greff, and J. Schmidhuber. Training very deep networks. In *Advances in Neural Information Processing Systems*, pages 2377–2385, 2015.
- [52] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017.
- [53] D. Thanh Nguyen, B.-S. Hua, K. Tran, Q.-H. Pham, and S.-K. Yeung. A field model for repairing 3d shapes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [54] D. Turmukhambetov, N. D. Campbell, D. B. Goldman, and J. Kautz. Interactive sketch-driven image synthesis. *Comput. Graph. Forum*, 34(8):130–142, Dec. 2015.
- [55] C. Wang, Z. Li, and L. Zhang. Mindfinder: image search by interactive sketching and tagging. In *Proceedings of the 19th international conference on World wide web*, pages 1309–1312. ACM, 2010.
- [56] F. Wang, L. Kang, and Y. Li. Sketch-based 3d shape retrieval using convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [57] X. Wang and X. Tang. Face photo-sketch synthesis and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):1955–1967, 2009.
- [58] S. Xie and Z. Tu. Holistically-nested edge detection. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [59] D. Yoo, N. Kim, S. Park, A. S. Paek, and I. S. Kweon. Pixel-level domain transfer. In *European Conference on Computer Vision*, 2016.
- [60] Q. Yu, F. Liu, Y.-Z. Song, T. Xiang, T. M. Hospedales, and C.-C. Loy. Sketch me that shoe. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [61] T. Zhang and C. Y. Suen. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–239, 1984.
- [62] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

## Supplementary Material Outline

Section 1 lists all categories we used in training our models. Section 2 compares the performance of MRU to some other models on CIFAR-10. Section 3 shows samples of generated images from all 50 categories.

### 1. Category list

Here are the 50 categories we use for training and testing our models: airplane, ant, apple, banana, bear, bee, bell, bench, bicycle, candle, cannon, car, castle, cat, chair, church, couch, cow, cup, dog, elephant, geyser, giraffe, hammer, hedgehog, horse, hotdog, hourglass, jellyfish, knife, lion, motorcycle, mushroom, pig, pineapple, pizza, pretzel, rifle, scissors, scorpion, sheep, snail, spoon, starfish, strawberry, tank, teapot, tiger, volcano, zebra.

### 2. Evaluation of MRU on CIFAR-10

We introduce the Masked Residual Unit (MRU) to improve generative deep networks by giving repeated access to the conditioning signal (in our case, a sketch). But this network building block is also quite useful for classification tasks. We compare the performance of the MRU and other recent architectures on CIFAR-10 and show that the MRU performance is on par with ResNet. Accuracy numbers for other models are obtained from their corresponding papers. For convenience, we call the improved ResNet "ResNet-v2" in the table. In "MRU-108, LeakyReLU gate", we substitute the sigmoid activations in our MRU units with LeakyReLU [42], and normalize obtained masks to the range of [0, 1].

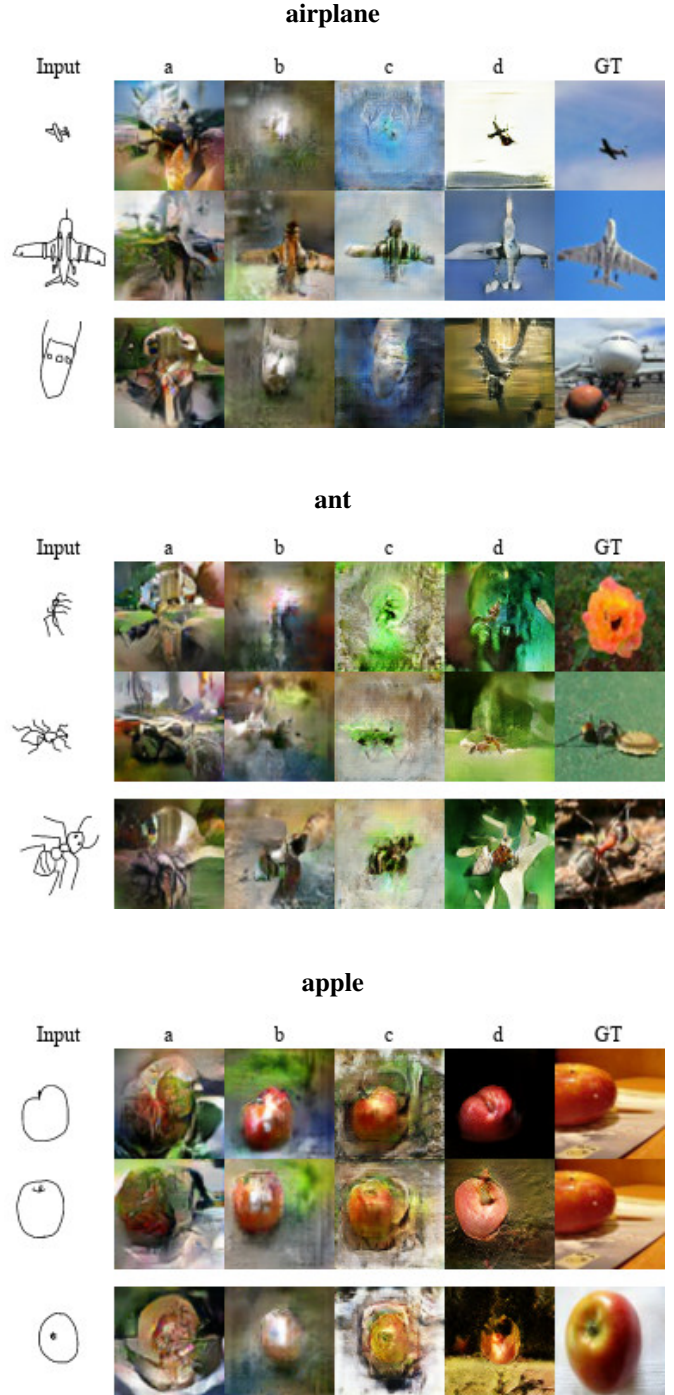
Model	error (%)
NIN [35]	8.81
Highway [51]	7.72
ResNet-110 [18]	6.61
ResNet-1202 [18]	7.93
ResNet-v2-164 [19]	5.46
MRU-108	6.34
<b>MRU-108, LeakyReLU gate</b>	<b>5.83</b>

Table 6: Comparison of error rates on CIFAR-10. Lower is better.

### 3. Samples from all 50 categories

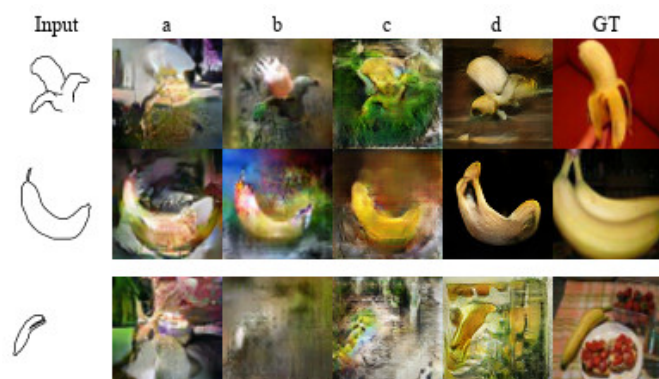
Here we present samples from all 50 categories from pix2pix variants and our methods for comparison. Each category contains three input samples, among which the third sample is a failure case for our method. The six columns in each figure are: (Input) input sketch, (a) pix2pix

on Sketchy, (b) pix2pix on Augmented Sketchy, (c) Label-supervised pix2pix on Augmented Sketchy, (d) our method, (GT) ground truth image.

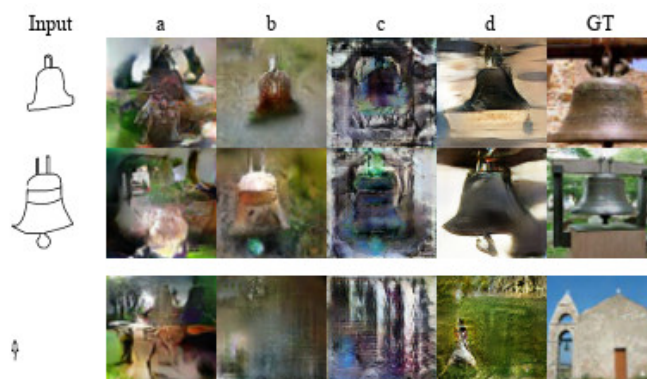




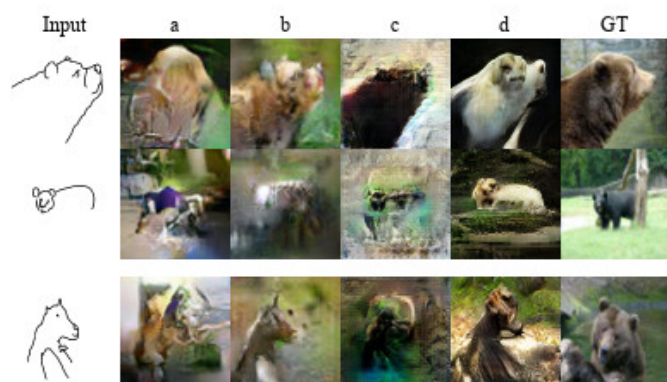
**banana**



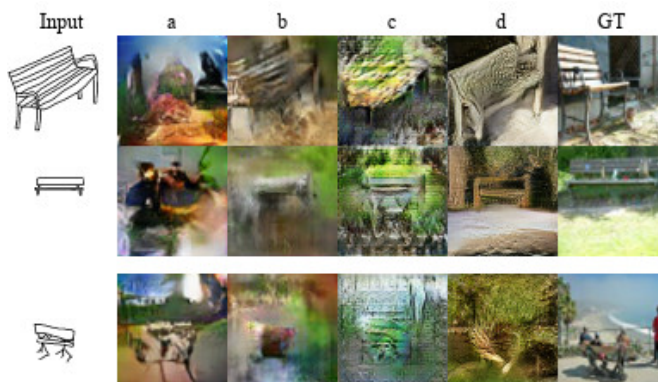
**bell**



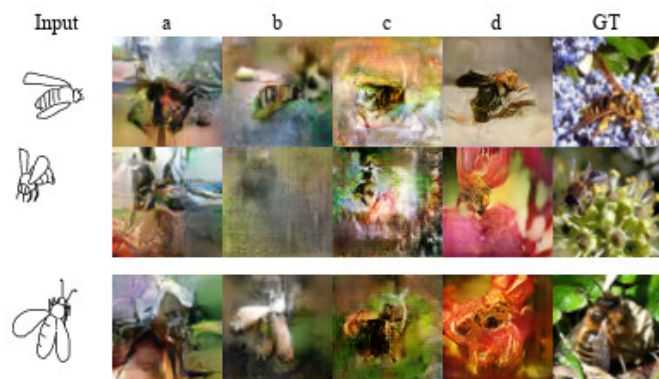
**bear**



**bench**



**bee**



**bicycle**



**candle**



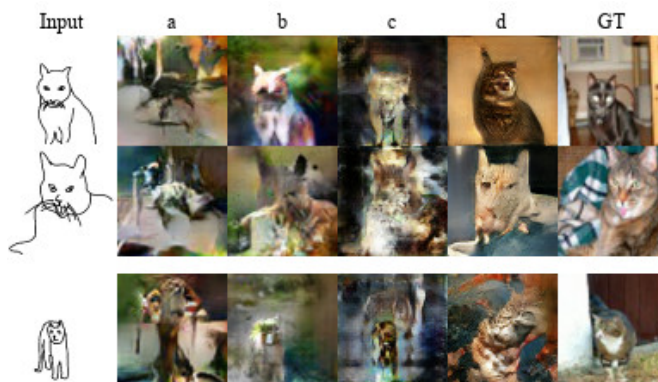
**castle**



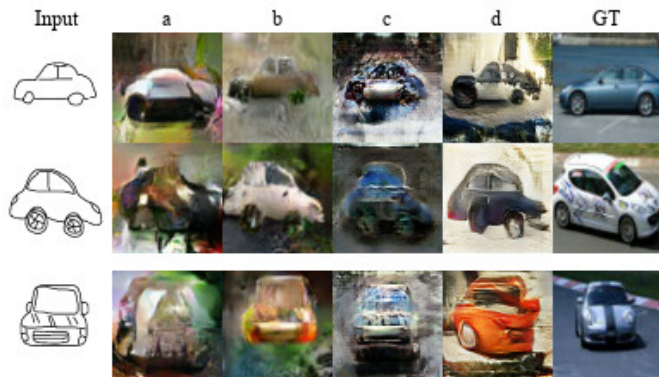
**cannon**



**cat**



**car**



**chair**

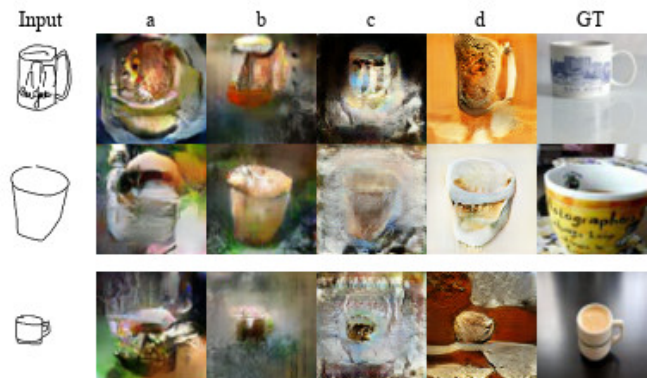




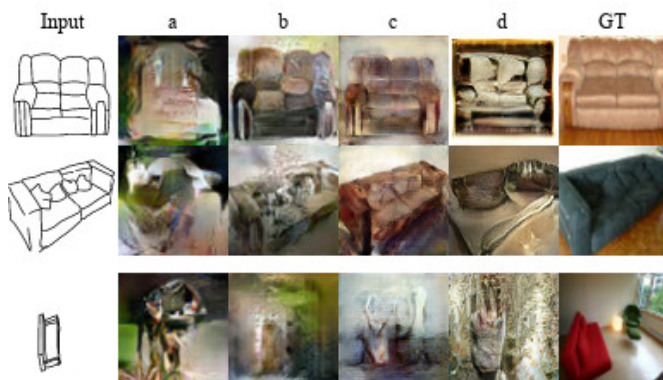
**church**



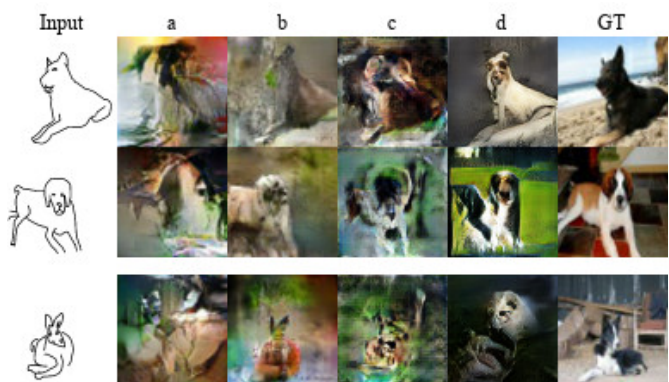
**cup**



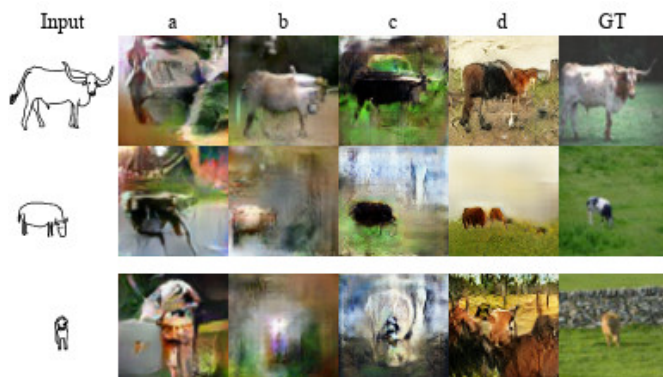
**couch**



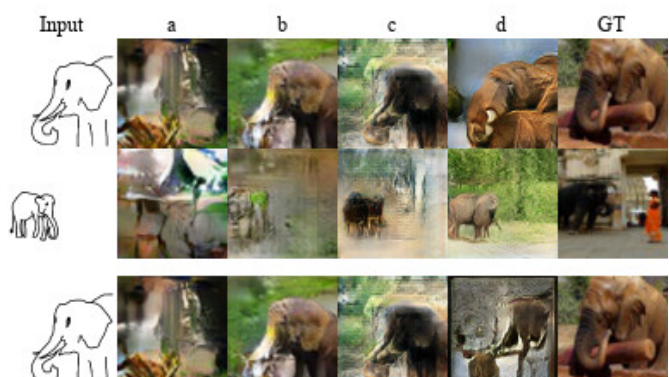
**dog**



**cow**



**elephant**

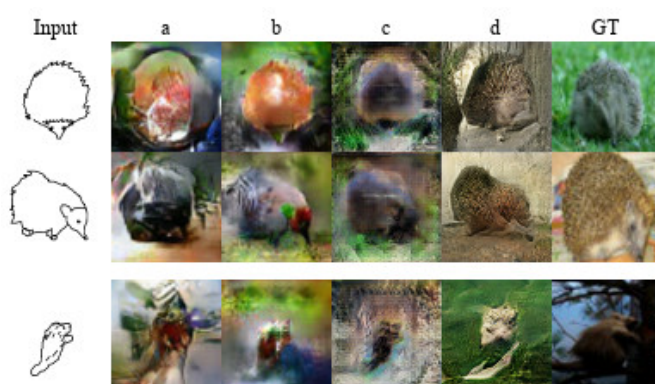




**geyser**



**hedgehog**



**giraffe**



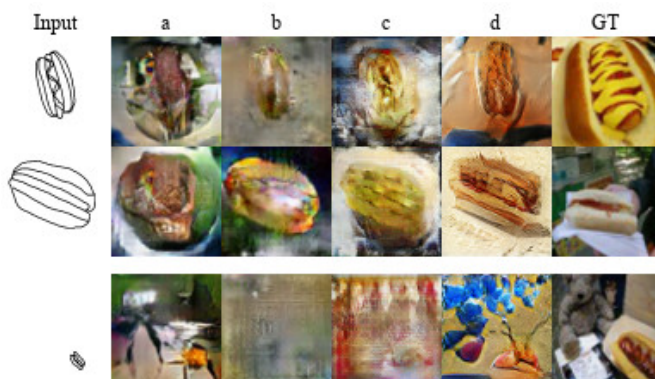
**horse**



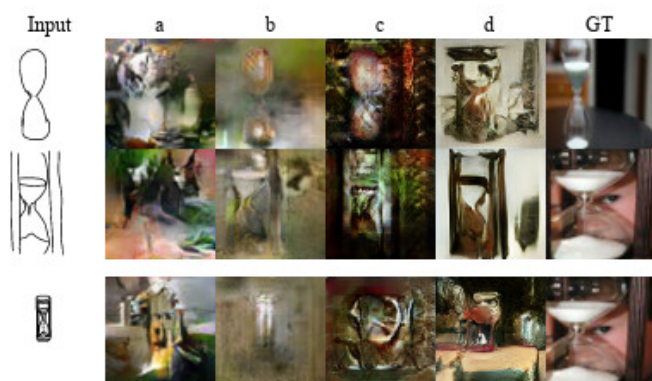
**hammer**



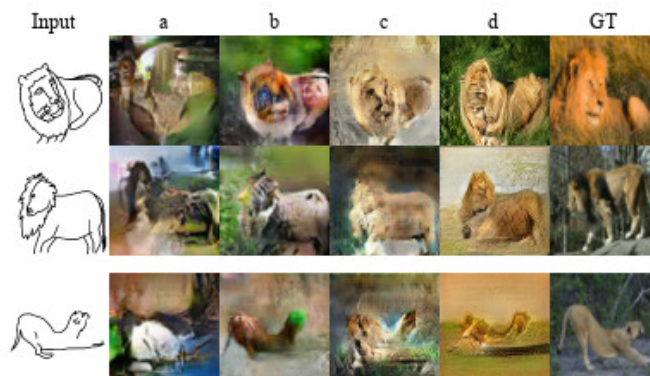
**hotdog**



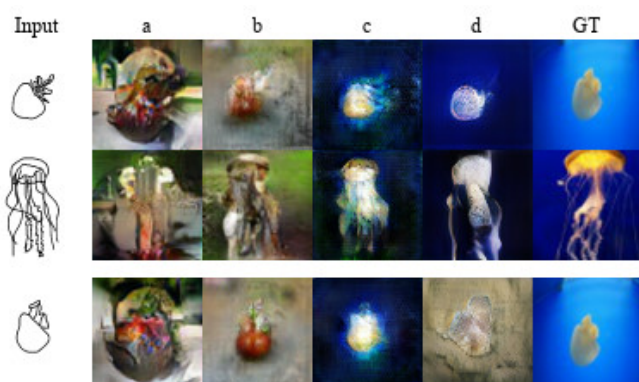
**hourglass**



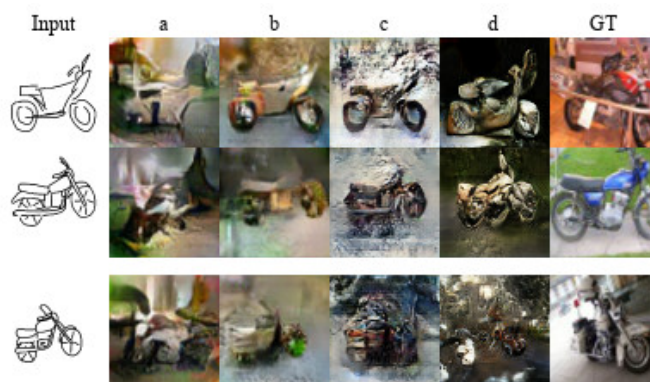
**lion**



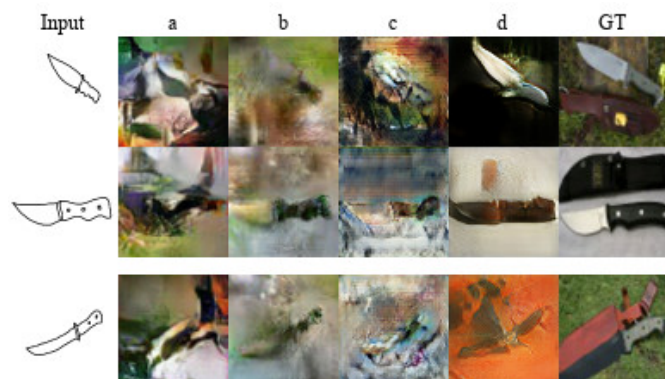
**jellyfish**



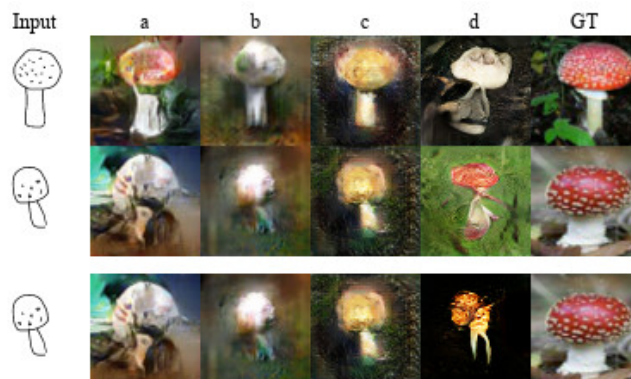
**motorcycle**



**knife**

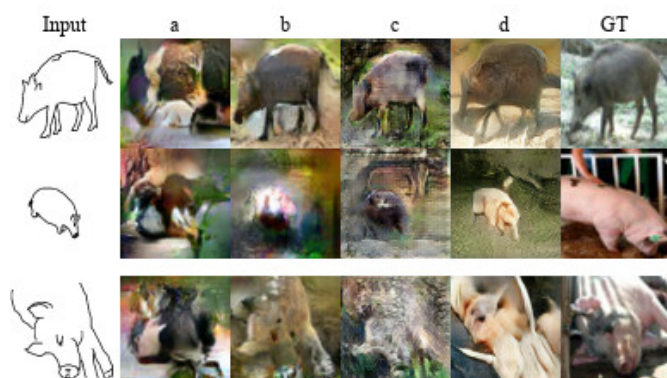


**mushroom**

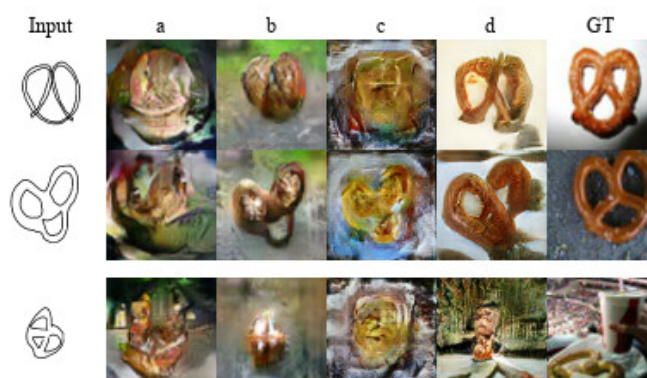




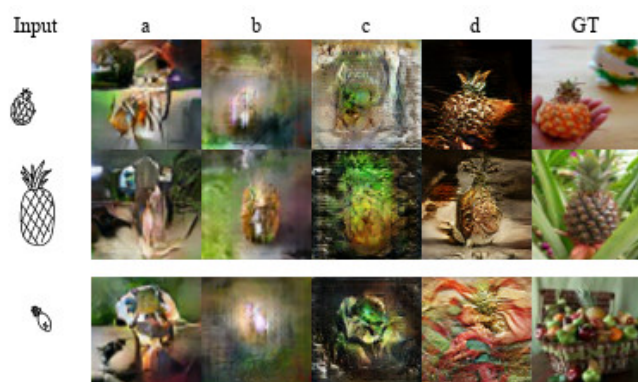
**pig**



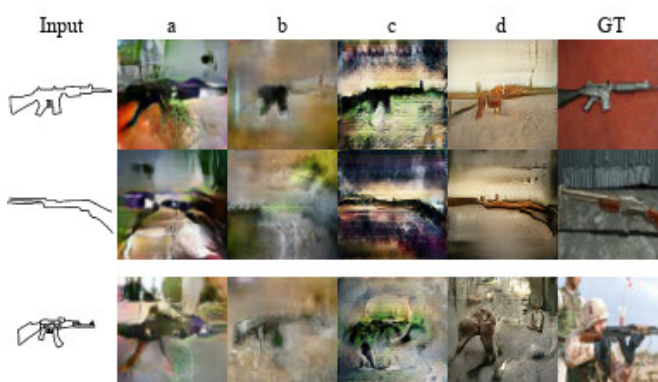
**pretzel**



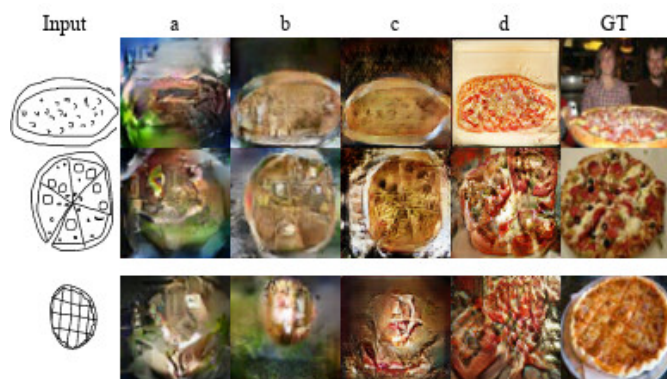
**pineapple**



**rifle**



**pizza**

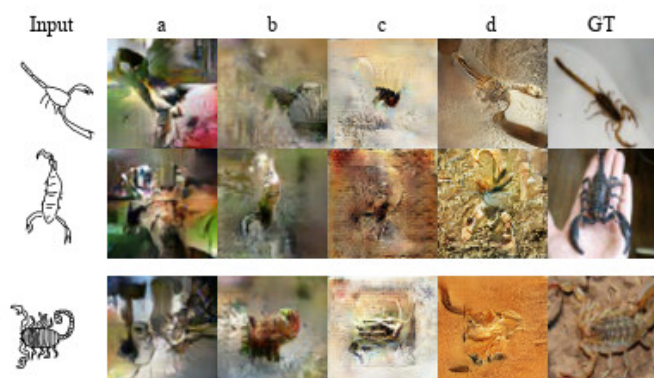


**scissors**

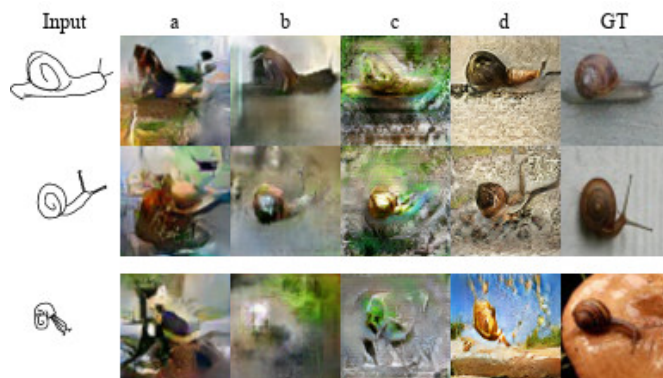




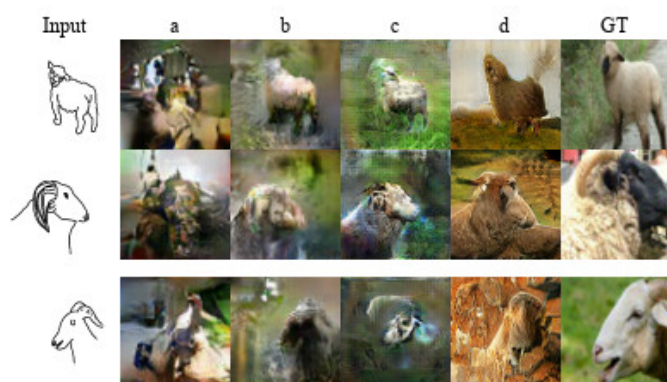
**scorpion**



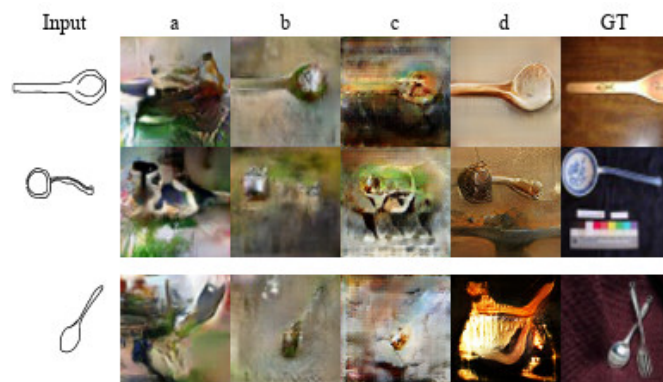
**snail**



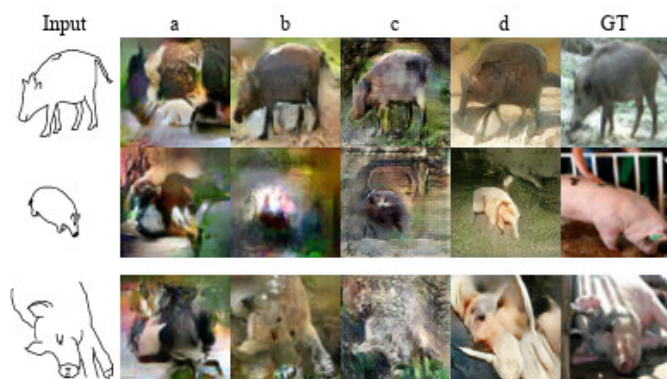
**sheep**



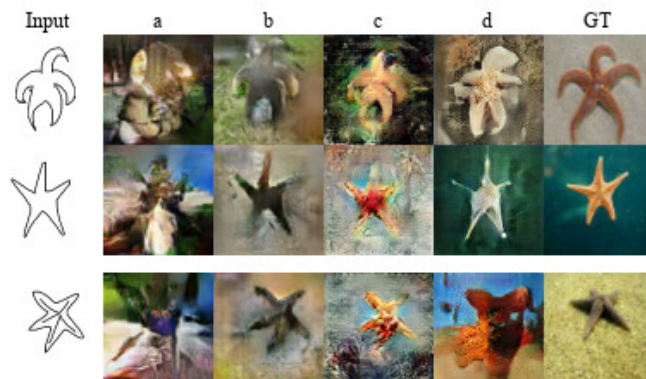
**spoon**



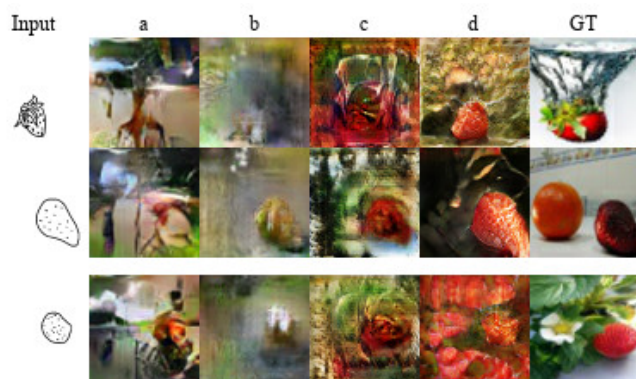
**pig**



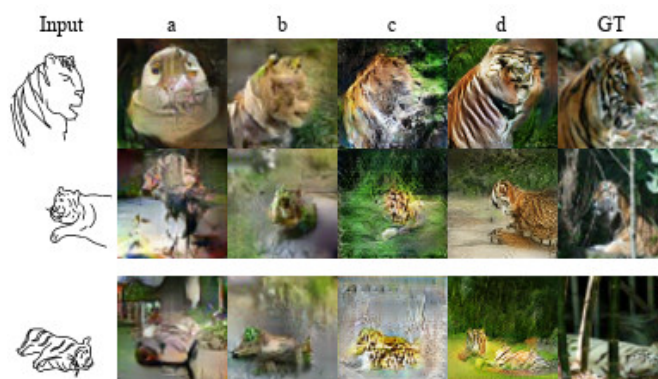
**starfish**



**strawberry**



**tiger**



**tank**



**volcano**



**teapot**



**zebra**

