# Using GANs for Sketch-to-Image Translation

**Hongtao Liu**
Department of Computer Science
New York University
New York, NY 10012
hl3635@nyu.edu

**Projjol Banerji**
Department of Computer Science
New York University
New York, NY 10012
projjol@nyu.edu

## Abstract

We look at the edges2shoes datatset and the usage of pix2pix and Bicycle GANs to generate fully colored and textured shoe images. We model a distribution of possible outputs in a conditional generative modeling setting. The mapping is distilled in a latent vector, randomly sampled at test time. Our generator learns to map the given input and latent code to the output. We explore several variants to reach this goal by employing different training objectives and network architectures.

## 1 Introduction

Our initial goal was to create a GAN derived architecture that could take sketches of human faces and convert them into photo-realistic images. We used facial dataset like celeba and colorferet. We had tried approaches like adding condition data (e.g labels) to the GAN and also creating symmetrical network. The first set of papers that we found in this area where not easy to follow along for us and those that we could make some progress on in theory did not have clear indications of where they derived their dataset from. In this process we found the pix2px paper by Isola et al. In their paper, they had shown that via the usage of sketches across the `edges2shoes`, `edges2handbags` and `facades` datasets photo realistic images of these items was possible. With that, we changed our approach, hoping to learn from the pix2pix paper and implement a GAN that can derive photo-realistic images of shoes. We feel that this cna be a powerful tool for designers who can simply provide the engine sketches and receive various iterations of how their creation can look like in the real world. In the following sections we will discuss the pix2pix paper, the challenge we faced with it, corresponding results and our approach to implement a Bicycle GAN for edges2shoes derived from our learning from the pix2pix paper.

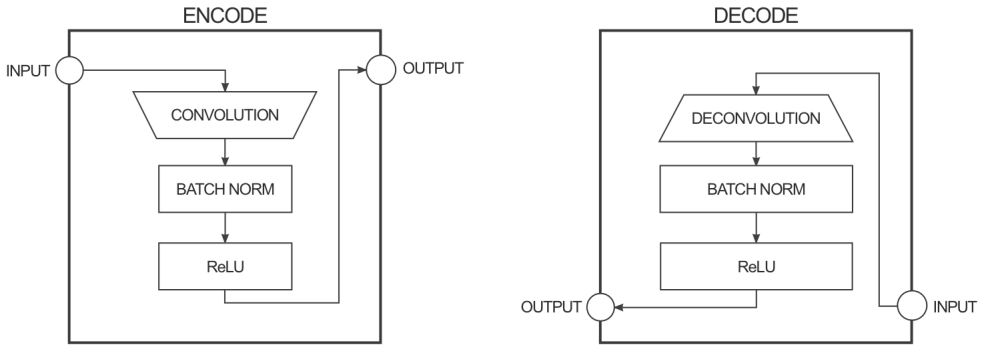
### 1.1 pix2pix

pix2pix uses a conditional generative adversarial network (cGAN) to learn a mapping from an input image to an output image. One of the innovations of this paper is that the discriminator provides a loss function for training the generator.

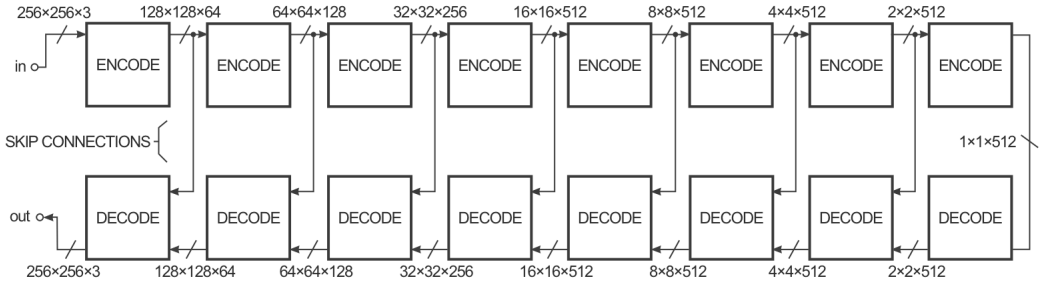
#### 1.1.1 pix2pix's Generator

The generator uses a 'encoder-decoder' structure and applies a transform in the image to get the desired output. The generator takes an input and tries to reduce it with a series of encoders (convolution + activation function) into a much smaller representation. The idea is that by compressing it this way we hopefully have a higher level representation of the data after the final encode layer. The decode layers do the opposite (deconvolution + activation function) and reverse the action of the encoder layers.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.



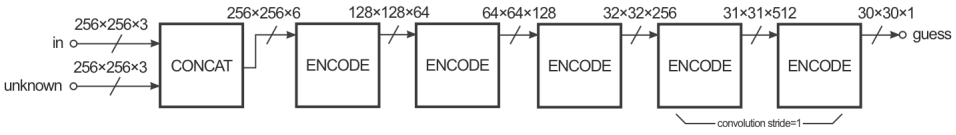

In order to improve the performance of the image-to-image transform in the paper, the authors used a "U-Net". With "skip connections" directly connecting encoder layers to decoder layers:




The skip connections give the network the option of bypassing the encoding/decoding part if it doesn't have a use for it. Please note that these diagrams are a slightly amplified as first and last layers of the network have no batch norm layer.

### 1.1.2 pix2pix's Discriminator

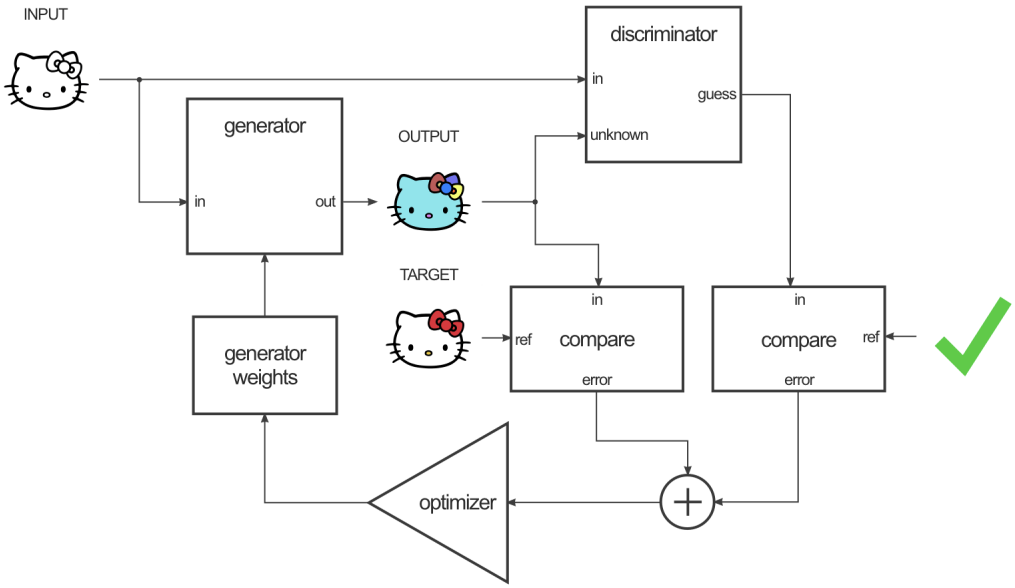
The Discriminator has the job of taking two images, an input image and an unknown image and deciding if the second image was produced by the generator or not.




While the structure (as shown above) looks like the generator, the working is slightly different. The output is a 30x30 image with every pixel value (0 to 1) representing how believable the corresponding section of the unknown image is. In the pix2pix implementation, each pixel from this 30x30 image corresponds to the believabality of a 70x70 patch of the input image.

## 2 Training

Training the network is two-fold process: training the generator and training the discriminator. To train the discriminator, first the generator generates an output image. The discriminator looks at the input/target pair and the input/output pair and produces its guess about how realistic they look. The weights of the discriminator are adjusted based on the classification error of the input/output pair and the input/target pair. The generator's weights are adjusted based on the output of the discriminator as well as the difference between the output and target image. `https://affinelayer.com/` has a good visual representation for this process which has been included below:




By training the generator on the output of the discriminator, we find that by improving the discriminator, the generator in turn gets trained to beat it.

## 3 Challenges and Results for pix2pix

Since the paper behind pix2pix was really popular, we found many blogs with simpler and annotated versions of the paper for us to follow along. That helped us get a good headstart when it came to writing the training script. One of the unexpected challenges of running it on Prince was tweaking the parameters being passed to the script. Additionally, the way we wrote our code was probably not the most memory efficient way of going about it as we would end up running out memory even on small batch sizes of 8. The only batch size we could get it to work on was 4 and that did slow us down considerably.

We personally, liked the results of the pix2pix implementation. We've attached some of the results below:


# 4 Implementing a Bicycle GAN

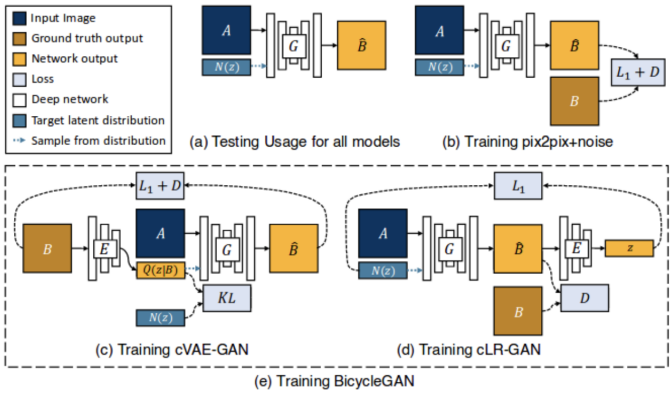
When we were working on the original idea of converting sketches of human faces to images, we found a paper on Bicycle GAN, Toward Multimodal Image-to-Image Translation, by Zhu et al. and were intrigued by it. Upon achieving a certain level of clarity in results we tried to see if we can use the same approach and apply it to the edges2shoes dataset. In the following sections we discuss Bicycle GANs and our results.

## 4.1 Bicycle GAN

In the paper, the authors aim to create a distribution of output images to a given input image by extending the model used in cGANs (seen in the pix2pix implementation). The model for bicycle GANs have two parts, the first of which involves using a conditional Variational Autoencoder GAN. We try to learn a low-dimensional latent representation of target images using an encoder net and try for this distribution to be close to a normal distribution, so as to make our task of sampling easier during inference. A generator is used to map the input image to an output image using the encoded representation z.

For the second part, we use a Conditional Latent Regressor GAN. In this, z is sampled from a normal distribution N(z) which in addition to the input image A is fed to the Generator to get the output image. This output image is then fed to the Encoder net to output z' which we try to be close to N(z).

In this model, the mapping from latent vector(z) to output images and output images to latent vector is bijective. The overall architecture consists of two cycle, B->z->B' and z->B'->z' and thus the name BicycleGAN. We have included an image to represent in clearer terms the architecture of the model:




### 4.2 Results

Our major challenge with Bicycle GANs was the unavailability of an easily readable primer. While we found a few explanations, it took us quite a while to translate our conceptual understanding of the paper to actual working code. With limited time, we could only run this for 35 epochs and we have presented our results as images below. We feel this might not be truly representative of the results this implementation is capable of, but given the low set of epochs it was run for we are happy with the outcome.

## References

[1] Phillip Isola & Jun-Yan Zhu & Tinghui Zhou & Alexei A. Efros (2019) Image-to-Image Translation with Conditional Adversarial Networks

[2] Jun-Yan Zhu et al. (2019) Toward Multimodal Image-to-Image Translation