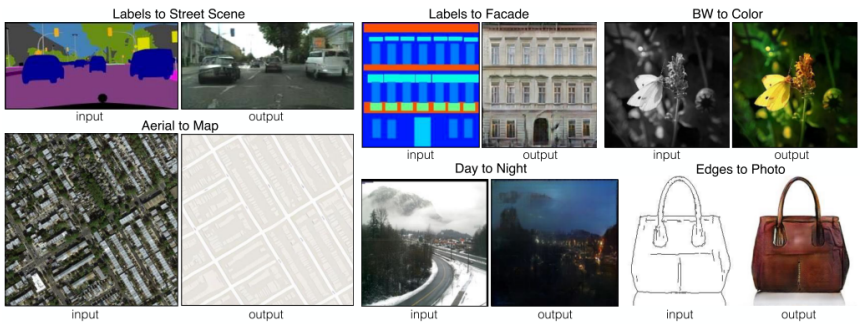
Arthur Juliani Follow
Deep Learning @Unity3D & Cognitive Neuroscience PhD student.
Jan 11 · 8 min read

Remastering Classic Films in Tensorflow with Pix2Pix

Happy New Year! In this first post of 2017 I wanted to do something fun and a little different, and momentarily switch gears away from RL to generative networks. I have been working with a Generative Adversarial Network called Pix2Pix for the past few days, and want to share the fruits of the project. This framework comes from the paper "*Image-to-Image Translation with Conditional Adversarial Networks*" recently out of Berkeley. Unlike vanilla GANs, which take noise inputs and produce images, Pix2Pix learns to take an image and translate it into another image using an adversarial framework. Examples of this include turning street maps into aerial photography, drawings into photographs, and day photos into night photos. Theoretically, translation is possible between any two images which maintain the same structure.



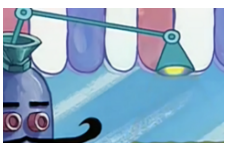
Examples of translated images using Pix2Pix.

What struck me as a possible and exciting usage was the capacity to colorize black and white photos, as well as fill in the missing gaps in images. Taken together these two capacities could be used to perform a sort of remastering of films from the 1950s and earlier. Films shot in black and white, and at a 4:3 aspect ratio could particularly benefit from this process. This "remastering" would both colorize and extend the aspect ratio to the more familiar 16:9.

Top: input. Middle: My Pix2Pix remaster. Bottom: Original. Taken from Rear Window.

In this post I am going to walk through the design of Pix2Pix, provide a Tensorflow implementation, as well as show off some results of the film remastering process. If you are new to GANs, I recommend reading my two previous posts on them in order to get a good sense of how they work. Pix2Pix is really just a GAN plus a few extra architectural changes.

Generative Adversarial Networks Explained with a Classic Spongebob Squarepants Episode



Plus a Tensorflow tutorial for implementing your own GAN

medium.com

Pix2Pix

Pix2Pix builds off of the GAN architecture in a pretty intuitive way. In the GAN formulation we have a generator `G` and discriminator `D`, which are trained adversarially. The generator is trained to generate realistic images from noise input `z`, and the discriminator is trained to differentiate between the real images `x` and those produced by the generator `G(x)`. Using the feedback from the discriminator, the generator can improve its process to produce images more likely to fool the discriminator in the future. Doing so produces more realistic images.

The most apparent change when going from a GAN to Pix2Pix is that instead of noise `z` the generator is fed an actual image `x` that we want to "translate" into another structurally similar image `y`. Our generator now produces `G(x)`, which we want to become indistinguishable from `y`.

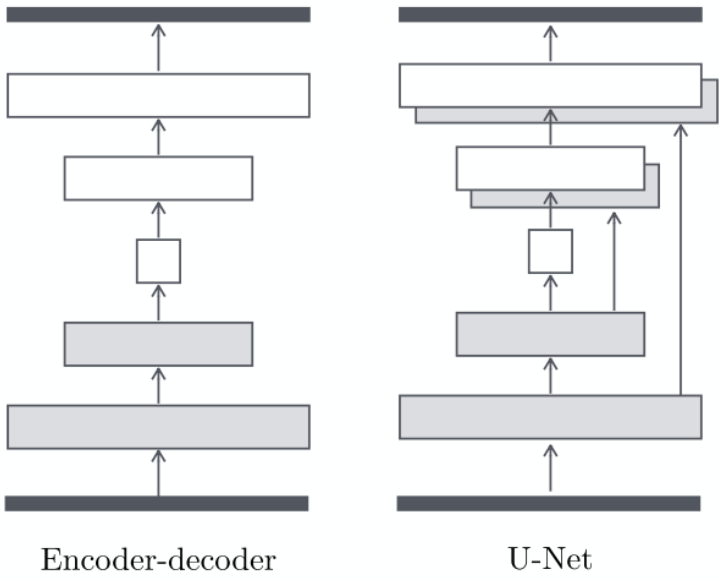
In addition to the original GAN losses, we also utilize an `L1` loss, which is just a pixel-wise absolute value loss on the generated images. In this case, we force the generator to approximate `G(x) = y` with the additional loss:

```
L1 = |G(x) - y|
```

In a traditional GAN we would never utilize such a loss because it would prevent the generator from producing novel images. In the case of image translation however, we care about accurate image translations rather than novel ones. This desire for accurate images is also why we don't entirely do away with the GAN aspect of our network. An `L1` loss by itself would produce blurry or washed-out images by virtue of attempting to generate images that are "on average" correct. By keeping the GAN losses, we encourage the network to produce crisp images that are visually indistinguishable from the real things.

Left: image translation with only L1 loss. Right: Image translation with L1 and GAN losses.

Along the lines of ensuring accurate images, the third change to Pix2Pix is the utilization of a U-Net architecture in the generator. Put simply, the U-Net is an auto-encoder in which the outputs from the encoder-half of the network are concatenated with their mirrored counterparts in the decoder-half of the network. By including these skip connections, we prevent the middle of the network from becoming a bottleneck on information flow.



U—Net Architecture

In the case of Pix2Pix, our input is the image we want to translate `x` and the output is the image we want it to become `G(x)`. By concatenating mirrored layers, we are able to ensure that the structure of the original image gets carried over to the decoder-half of the network directly. When thinking about the task of colorization, the representations learned at each scale of the encoder are extremely useful for the decoder in terms of providing the structure of the colorized image.

In the Pix2Pix paper, the authors also discuss utilizing a different discriminator architecture referred to as PatchGAN. The idea behind PatchGAN is that instead of producing a single discriminator score for the entire image, a set of separate scores are produced for each patch of the image, and the scores are then averaged together to produce a final score. This approach can improve performance by relying on fewer parameters, and a correctly tuned patch size can lead to improved image quality. A PatchGAN with a patch size the same as the image size is thus equivalent to a traditional discriminator architecture. For the sake of simplicity my Pix2Pix implementation utilizes a traditional discriminator, or PatchGAN over the image.

Remastering Films

Given the long history of film, many of the classics in the medium were created over half a century ago, and as such were made with very different technology than the films of today. Classics such as Casablanca, Citizen Kane, Metropolis, It's A Wonderful Life, etc, were all shot in black and white and at a limited aspect ratio. Despite their visual limitations, these films still evoke complex and fully realized worlds. Watching them, it is easy to close your eyes and imagine what they might look like in full color and a wider field of view. Given enough time, an artist could even colorize and extend the frames, as has sometimes been done for classic films. With the power of the Pix2Pix framework, I wondered if it would be possible to have a neural network learn to do this automatically.

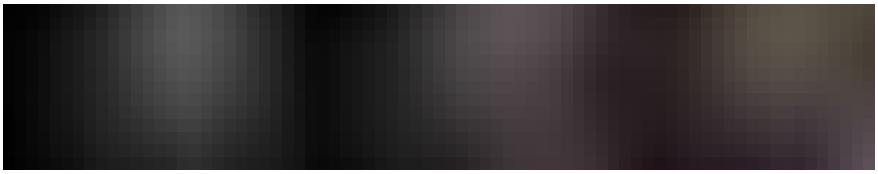
In order to accomplish this I transformed my GAN implementation into a Pix2Pix architecture as described above and began training it on a set of classic films. The set-up was simple: I decided to utilize a collection of Alfred Hitchcock films (*Vertigo*, *Rear Window*, *North by Northwest*) as the training set. I choose the films because they were made in color and wide-screen, but were produced at a point where many films were still in black and white. As such, much of the clothing, lighting, and architecture would be similar to the even older films which I wanted to remaster. I also chose them for being some of my favorites from the era.

Each video was resized to 256x144, and image frames were extracted at 2fps. From that point each frame was pre-processed by being converted to grayscale, and cropped to impose a 4:3 aspect ratio. This set of frames composed the training inputs `x`. The original frames were used as the desired outputs `y`.

Top: input x. Middle: output G(x). Bottom: original y. Taken from North by Northwest.

After a day of training, there are impressive results! On the training films themselves the network is able to pretty accurately reproduce the original images (as seen above).

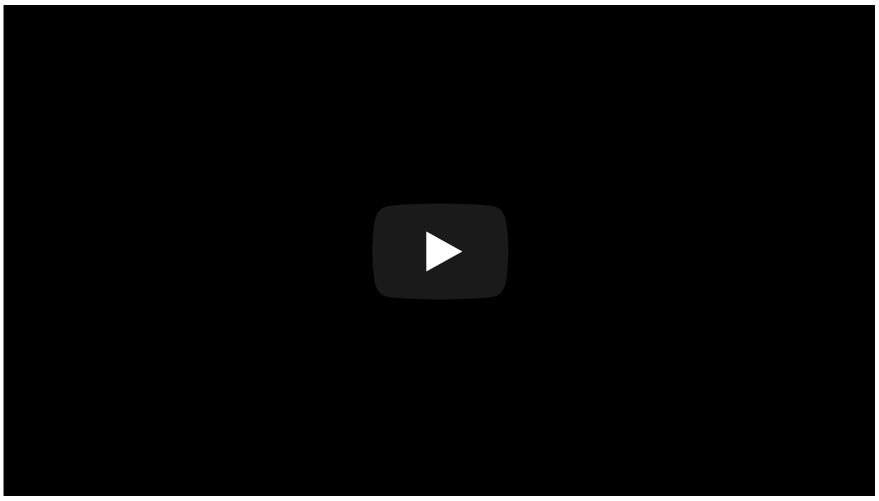
Of course in the field of ML this is basically cheating. A separate test set is needed to get a true sense of performance. Below is an example from *The Birds* (another Hitchcock classic) recreated using the network. Critically, the network was never trained using any of these frames, or any other frames from the film.



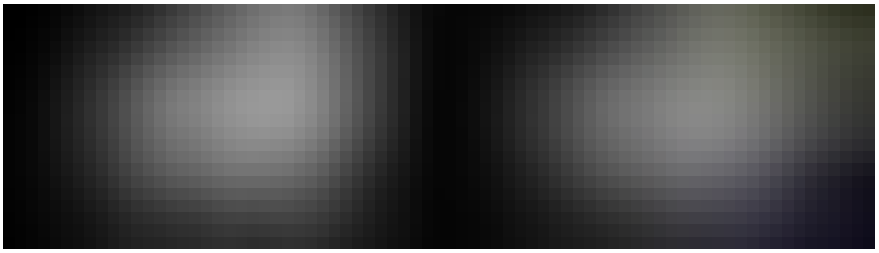
Left: input x. Middle: output G(x). Right: original y. Taken from The Birds.

At least within the transfer domain of "films by the same director" the network seems to perform astonishingly well. Another longer

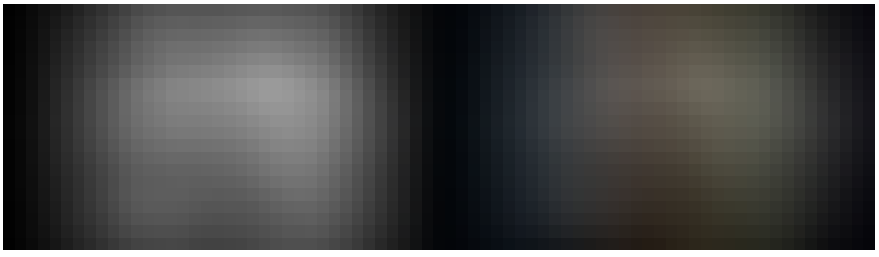
remastered sequence from *The Birds*:



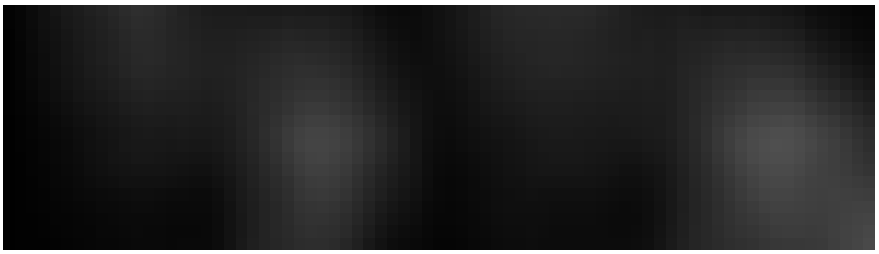
The main reason to create this however was to allow for the remastering of older films for which there exists no color or wide-screen version. Here are some examples of *The Passion of Jean d'Arc*, *Metropolis*, and *It's A Wonderful Life* all 'remastered' using the trained Pix2Pix network:



Left: Original. Right: Remastered. Taken from The Passion of Joan of Arc.



Left: Original. Right: Remastered. Taken from Metropolis.



Left: Original. Right: Remastered. Taken from It's a Wonderful Life.

The results aren't perfect, but I am pretty happy with them, given the relatively short training time, small training set, as well as minimal changes made to the baseline framework. With a longer training time and a larger set of films used, there is no doubt that results would improve. There are also a number of additions to the architecture that could improve performance. One of the biggest shortcomings is that the network doesn't take advantage of the temporal structure of the videos. An object off-screen in one frame may be present in another. As such an RNN or sliding window could be utilized in order to capture this information between frames. I leave this work for anyone interested to pick up.

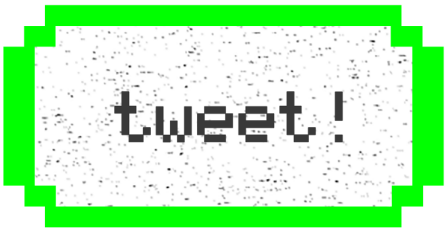
This Github repository contains everything needed if you'd like to train and run Pix2Pix yourself. Additionally, I have uploaded a pre-trained model here, if you'd just like to try remastering your own favorite older film without retraining a model from scratch. Instructions for getting up and running are provided in the repository.

This kind of work is in the early days still, but I am confident that not so many years from now we will be able to sit down and watch Metropolis or countless other classic films with all the visual detail that we can imagine them having. The capacity to breathe life into older media is an exciting potential of Deep Learning which will only grow as the technology matures.

. . .

If you'd like to follow my writing on Deep Learning, AI, and Cognitive Science, follow me on Medium @Arthur Juliani, or on twitter @awjliani.

Please consider *donating* to help support future tutorials, articles, and implementations. Any contribution is greatly appreciated!



> *Hacker Noon is how hackers start their afternoons. We're a part of the @AMI family. We are now accepting submissions and happy to discuss advertising & sponsorship opportunities.*

> *If you enjoyed this story, we recommend reading our latest tech stories and trending tech stories. Until next time, don't take the realities of the world for granted!*