TRANSLATION BETWEEN PAINTING AND PHOTO USING CONDITIONAL GENERATIVE ADVERSARIAL NETWORKS WITHOUT EXAMPLES FROM INPUT SPACE

Yihao Liu[1]*, Mariya Kazachkova*[2]*, and Channing Kimble-Brown*[2]

[1]Dept. of Electrical and Computer Engineering, The Johns Hopkins University, Baltimore, MD 21218
[2]Dept. of Computer Science, The Johns Hopkins University, Baltimore, MD 21218

ABSTRACT

Fabricating images using the semantic content of one image and the style of another is a difficult task. The quality of the resulting image is often far from the desired outcome; although the style of the painting is clear, the finished product does not look the part of being a real painting. Here, we set out to describe, implement, and test a process that will yield result images with the content of a real photo and the style of a Monet painting. We use a Conditional Generative Adversarial Network, as well as various methods for extracting the content from the images. In this paper, we introduce an intermediate space to pair with our output space while training the cGAN to compensate for the lack of paired data at the outset of this project. The belief is that the adversarial process will allow us to get closer to a realistic painting than existing versions of style transfer are capable of. We demonstrate our approach using a dataset of 400 Monet paintings. Our results show the potential for using Conditional GANs in style transfer, as well as illustrate a process for making fake data as realistic (or, in our case, as Monet-esque) as possible.

1. INTRODUCTION

Style transfer has become a popular topic in recent years. As the presence of digital art grows, so does the need for tools that allow artists to create new works while preserving classicized styles. With the recent work of Gatys *et al*. [4], we have all seen technology's capability in recreating a photo in the style of a famous artwork. However, as often happens when technology gets involved in human activities that require creativity and subjectivity, the results are not always truly reminiscent of human artwork (think uncanny valley). In order to combat this we believe a conditional generative adversarial network (cGAN) could be employed.

Conditional adversarial networks have been used extensively in image-to-image translation problems and have produced remarkable results in various tasks [1] [2]. The generator learns the mapping from input space to output space while the discrimintor learns to discriminate between real images from output space and generated images. The conditional image input allows the model to condition on the input content, this makes cGANs suitable for image-to-image translation tasks.

The work by Gatys *et al*. in context of style transfer presented a novel approach of extract the semantic information from the target image using a pretrained convolutional neural network [4]. When convolutional neural networks are trained for object recognition tasks, the content becomes increasingly explicit along the layers and relatively invariant to its precise appearance [5]. Therefore, a content image can be reconstructed from noise by gradient descent of the perceptual losses at higher layers. In contrast, reconstructions from the lower layers simply reproduce the exact pixel values of the original image.

Recently proposed models including cycleGAN [3], dualGAN [6] and discoGAN [7] aim to discover the mapping between input space and output space with unpaired datasets using coupled GANs. Thus, the generative model can be applied to problems like translating a photo into a Monet painting, in which paired data is unavailable. What if we wanted to translate a painting from an unknown artist to a photo? Humans can imagine the scene depicted in a painting without other paintings from the same artist, as long as the scene in the painting "makes sense" to the observer. Similar to the way humans imagine the translation, we can push these generative models one step further by finding a proper content extractor. Our conditional generative adversarial network is trained with generated paired examples: examples from the output space and their corresponding content images generated by the content extractor. For an input image from an arbitrary domain, we extract its semantic content and then use this content image as conditional input to the generator as shown in Fig. 1.

2. METHODS

The essential component of our method is finding the content representation of an input image using a content extractor. The ideal content extractor preserves necessary information to sufficiently represent the content in the image while discarding unnecessary information like style. The first part of this section will discuss three content extraction approaches.

2.1. Using Edge-Image as Content Image

A naive way of representing content is simply the creation of an edge-image for each of the paintings that we wanted to train on. We believed this would be a good approach because using just the edges of an image would leave the structural information of the image intact while flat regions, which have less information, will be discarded. We used PIL's edge image filter to create our content images. See Fig. 2 for an example of what an output image created using this process looks like.

(a) Actual (b) Content

Fig. 2: Image b is the edge-image of Image a

2.2. Style Transfer to Create Content Image

Our next attempt at creating content representations of our images was to transfer a style to them that would remove the original image style. The idea is to transfer all of the input images to a intermediate domain. We chose to transfer the style of a pencil drawing to each of our Monet paintings for a structural representation of the content. To perform this style transfer we used an implementation of the algorithm developed by Gatys *et al.* [4] This implementation uses a 19-layer VGG network and a loss function specifically designed to simultaneously minimize both the content loss and style loss. The content loss is computed at layer 5 and the style image is a representation of the consistency between convolutional layers 1 through 5. See Fig. 3 for an example of what an output image created using this process looks like.

(a) Actual (b) Content

Fig. 3: Image b is the image produced after transferring the style of a pencil drawing to Image a

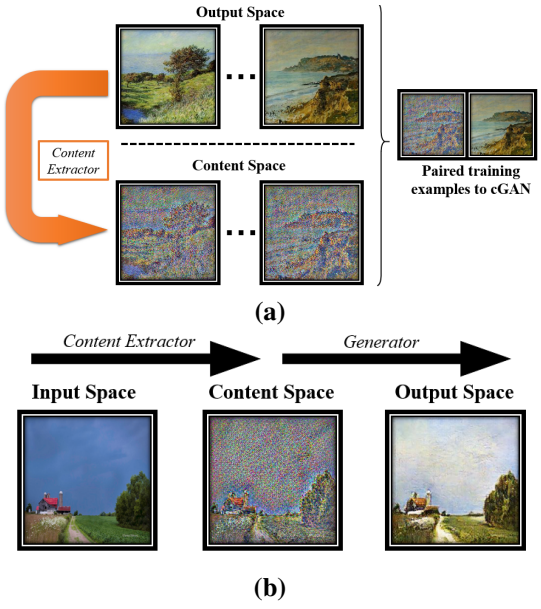
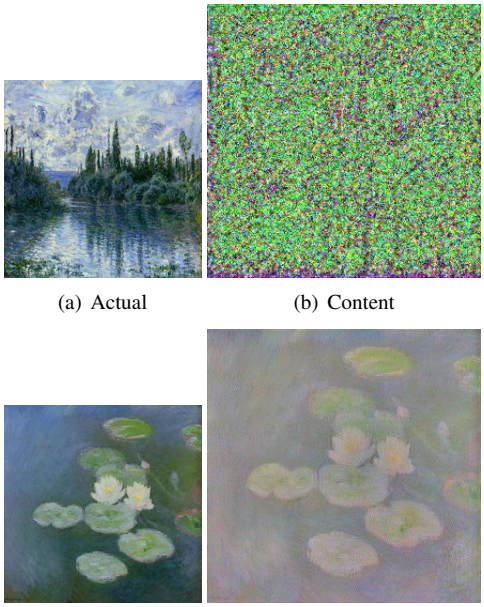


Fig. 1: The process of training is shown in **(a)**:a content extractor is used to fabricate training examples in the intermediate content space, then paired with the original painting to train the cGAN. While during testing, the same content extractor extract the content from test image and input to generator as shown in**(b)**



(a) Actual (b) Content

(c) Actual (d) Content

Fig. 4: Both content images were produced by extracting content from the 16th layer. While Image c produces a recognizable content image, Image a yields a content image that is almost entirely noise.

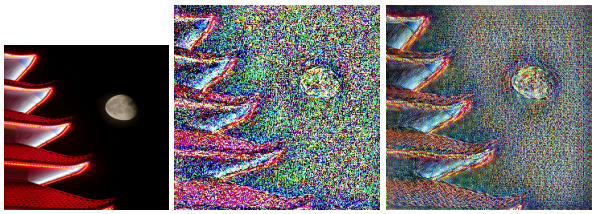


Fig. 5: Content extracted from 8th layer with noise and without noise, respectively.

2.3. Extracting Content Image During Style Transfer

The last procedure we implemented to create our content images involved modification of the style transfer algorithm developed by Gatys *et al.* In the aforementioned algorithm, both content loss and style loss are computed in order to create the resulting image. The paper discusses the ability to visualize encoded information at different layers in the network by performing gradient descent over a white noise image. We decided to use this idea to create our content images.

We first attempted to do this content reconstruction on the 16th layer of the network. Our content images just looked like noise. We hypothesized that this was because we were getting stuck in a local minimum duration optimization. We tried to counter this by making our initial input image gray-scale prior to running it through the network, but we found that this simply produced a content image that looked exactly like the gray-scale image that we started with. This problem arose because we used perceptual loss as our content loss. We tried to coax change by adding noise to the input image every 50 epochs. While this did cause our final content image to look different from our initial input image, it was unstable. Roughly 1 in every 20 images would produce a content image that looked like noise (see Fig. 4). This could be a result of the accumulation of noise being too big for reconstruction. Because of this issue, we chose to make some modifications to this process of extracting content from an image.

The modifications we made were as follows: we decided against using the gray-scale image because it caused little to no change to be made to our resulting content image. We also experimented with reconstructing content from a lower layer in the network to prevent our content image from looking like noise. After some trials we saw that we could extract content from the 8th layer in our network stably (we began experiencing unstable results in the form of noise images at the 9th layer). Instead of adding noise to the input image, we switched to using noise as the style image. The idea is that if we simply use the content loss to reconstruct the image, some unnecessary content information will be reconstructed. However, by adding a noise image as style input, our method minimizes both content loss and style loss. This will replace unnecessary content information with noise, see Fig. 5.

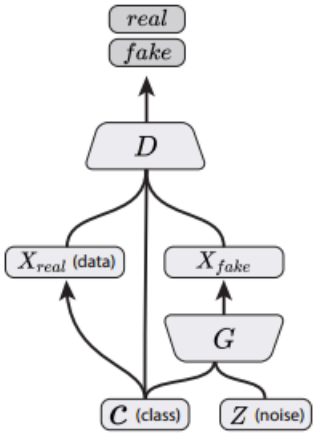


Fig. 6: Conditional GAN Diagram. During training C is a Monet painting, Z is the content image of that painting, X_{fake} is the generated painting, X_{real} is a real Monet painting, and D is the discriminator. During testing C becomes a set of photos and Z becomes the content images of those photos and X_{fake} is the desired output with Monet's style transferred onto the input photo.

2.4. Conditional Generative Adversarial Network

The conditional GAN, or cGan, has the same structure structure as a GAN except for the additional feature of a condition on the generator in the interest of mapping an input image and some random noise vector to some output image. The goal is to produce a fake output image that is in the domain of the input image and constrained by the condition, as shown in Fig. 6

For this project we used the same model as Isola et al. [1] to conduct all of our experiments. The goal is to find G^* as shown in equation (1).

$$(1)\ G^* = arg\,min_G\,max_D\,L_{cGAN}(G,D) + \lambda L_{L1}(G)$$

In (1) $L_{cGAN}(G,D)$ is a loss function for the generator's ability to fool the discriminator and the discriminator's ability to successfully distinguish between real and fake input. $\lambda L_{L1}(G)$ is an L1 distance that represents the generator's task of generating output as close to the noise vector as possible.

Isola et al. [1] go on to describe some architectural variations on the traditional cGAN. To prevent a bottleneck of information that Encoder-Decoder generators face, they instead used a "U-Net" model that has skip channels between every layer that concatenates at one layer with those at the layer it is connected to. The discriminator is designed to only penalize image structure at the scale of patches to help the network become better at generating high-frequency structures and produce crisper images.

(e) Real (f) Edge (g) Pencil (h) 8th - no noise (i) 8th - noise

Fig. 7: This shows the results on some of our test images using the condition GAN trained with paired data extracted in four different ways.

3. RESULTS

We test our method on the *Photo* $\rightarrow$ *Monet* dataset. We use only 400 examples from the output space, that is 400 paintings from Monet. We adopt the content extraction approaches described above and results are shown in Fig. 7

Since our generator learns the mapping between the content space and the output space, there is no need to retrain a cGAN for changing the input space. In other words, we train a cGAN to learn a general mapping instead of a fixed input space to output space translation. If a cGAN is trained to learn to translate from the content space to Monet's paintings, then the task of translating van Gogh's paintings to Monet's style is almost trivial: the only thing left to do is extract the content from van Gogh's paintings, see Fig. 8 and Fig. 9. The same content extractor that was used during the training process should be applied to ensure the content image within the content space is recognized by generator.

While our method produced results with high perceptual quality, there are still some limitations to finding a proper content extraction approach. Though content reconstruction yields satisfying content representation of scenes for paintings and photos, its reconstruction results highly depend on the recognition capability of the VGG network. Our method performed poorly for label to photo translation tasks because the content extracted from label image looked significantly different from content extracted from a real photo. We also observed issues with translation tasks involving different objects, like from zebra to horse translation or orange to apple translation. An obvious reason for our method to fail is the VGG network regards horse and zebra as two separate classes. Thus, the pattern from the zebra class remains on the content images.

4. DISCUSSION

It is pretty clear that our content represented as an edge-image produced the worst results. We hypothesize that this may be because we could be losing important aspects of the content while also preserving elements of the style.

Our content represented as a pencil image produced decent results. It does look like style has been transferred; however, the texture of Monet's style does not seem to be incredibly present. As with the edge image, we think that using the pencil image caused elements of the style to be preserved.

Our content reconstruction from the 8th layer without noise yielded results similar to those of the pencil image. Again, we think this is because all of the content was preserved as well as elements of the style (so the result image has all of the correct content but also style elements from the original, making it look more like a modified photo than a painting).

Our content reconstruction from the 8th layer with noise produced the best results. We believe that adding noise was the key to the quality results because the noise allowed us to remove unnecessary content by adding noise to the style. Thus, when our content image went through the cGAN, the style was covered by noise. This means that significantly less style elements were preserved from the original image, allowing us to produce an image that more closely resembles a painting.

(b) Actual (c) Content (d) Result

Fig. 8: Demonstration from a Monet painting to that same painting in the style of van Gough

(b) Actual (c) Content (d) Result

Fig. 9: Demonstration from a van Gough painting to that same painting in the style of Monet

5. REFERENCES

[1] M. Mirza and S. Osindero. Conditional generative adversarial nets. CoRR, abs/1411.1784, 2014

[2] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2016). Image-to-image translation with conditional adversarial networks. arXiv preprint arXiv:1611.07004 .

[3] Jun-Yan Zhu*, Taesung Park*, Phillip Isola, and Alexei A. Efros. "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks", in IEEE International Conference on Computer Vision (ICCV), 2017.

[4] Gatys, L.A., Ecker, A.S., Bethge, M.: A neural algorithm of artistic style. arXiv preprint arXiv:1508.06576 (2015)

[5] Gatys, L.A., Ecker, A.S., Bethge, M.: Texture synthesis using convolutional neural networks. In: Advances in Neural Information Processing Systems 28. (May 2015)

[6] Z. Yi, H. Zhang, and P. Tan. DualGAN: Unsupervised dual learning for image-to-image translation. ICCV, 2017.

[7] 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. 2