Image Completion using Generative Models

Kevin Joseph
University of Massachusetts Amherst
kevjoseph@cs.umass.edu

Aditya Arcot Srinivasan
University of Massachusetts Amherst
asrinivasan@cs.umass.edu

Abstract

Image completion, also known as image in-painting is an active computer-vision research problem that aims to automatically fill in a missing portion of an image in a content-aware way. Existing image completion algorithms that seek to perform this task, based on just the information from the given incomplete image do not produce satisfactory results. In this project we make use of a Deep Convolutional Generative Adversarial Network as proposed by Radford et al. *[11] to complete images. Therefore, using our trained generative model, we closely follow the method presented by Raymond Yeh* et al. *[13] for semantic image in-painting which generates missing content by conditioning on the uncorrupted pixel information. The core of the image in-painting task is performed by searching for the nearest encoding of an incomplete image in the latent image manifold via defined loss functions; the generative model then uses this encoding to infer missing content.*

1. Introduction

Visual phenomena in the human world is extremely intricate and structured in complex ways; yet humans are able to make sense of this structure. In this course project, we show that deep learning models can do the same when performing the task of image in-painting. Consider the image shown in Figure 1. Although a chunk of the image has been blacked out in the middle, many of us would be able to infer how the missing piece should be filled up. This implies that there must exist some underlying structure in the way natural images manifest themselves. In this project, we train a Deep Convolutional Generative Adversarial Network (DCGAN) to learn and predict this structure.

Image completion, is an active computer vision research problem that aims to automatically fill in a missing portion of an image in a content-aware way. It is very closely related to in- painting which is the application of sophisticated algorithms to replace lost or corrupted parts of the image data (mainly small regions or to remove small defects).

In this project we will be exploring Generative adversarial networks(GAN) [5] to perform the task of image completion in a context aware manner. We exploit the ability of these class of models to generate realistic "articles" or more specifically realistic images. To complete the task of image completion we require the generative model to not only produce a realistic image but to also make use of the contextual information contained in the incomplete image, such that the final completed image is one that the discriminative model would be unable to distinguish as either 'real' or 'fake'.

Figure 1. An ideal image completion model will fill in the undefined region based on the information given in the defined region.

The datasets we used the were the Large-scale CelebFaces Attributes (CelebA) Dataset [10] and Labeled Faces in the Wild [6]. For both these face-image datasets the training images were cropped to size 64×64, such that the face was centered.

The rest of this paper is structured as follows: section 2 discusses related work of image completion problem; section 3 elaborates on the image data set we chose to use for this project; section 4 and 5 discusses the details of the algorithms we chose for this task and actual evaluation of their performances; section 6 contains a discussion about GANs and concludes our project work.

2. Related Work

The research in Image Completion has spanned across multiple techniques including traditional image processing techniques that do not necessarily have a machine learning component to modern deep learning techniques like the one we present in this project.

The image inpainting problem was formalized initially by Bertalmio et al[3]. A number of techniques were then proposed to address this problem. Criminisi *et al*. [4]. introduced a milestone algorithm that formed the basis of the Exemplar-Based Inpainting algorithm. Sun et al. [8] proposed an image completion algorithm based on emphasizing the underlying structures in the images by looking at the entire image rather than nearby pixels around filling regions. All of these methods compute the missing pixels in the image by inference based on individual pixels.

A technique called seam carving was formulated by Avidan *et al*. [2] . Seam carving is a technique that extracts image content information from an optimal 8-connected path of pixels from top to bottom, or left to right. All the aforementioned algorithms are based on traditional image processing techniques with no machine learning involved.

2.1. Image Completion with Deep Learning

Our project builds upon the concept of a GAN network concept first introduced by Goodfellow *et al*. [5]. The idea is to simultaneously train two neural nets. The first one, called the Discriminator let's denote it as D(Y) takes an input (e.g. an image) and outputs a scalar that indicates whether the image Y looks "natural" or not. In one instance of adversarial training, D(Y) can be seen as some sort of energy function that takes a low value (e.g. close to 0) when Y is a real sample (e.g. an image from a database) and a positive value when it is not (e.g. if its a noisy or strange looking image). The second network is called the generator, denoted G(Z), where Z is generally a vector randomly sampled from a simple distribution (e.g. Gaussian or Uniform). The role of the generator is to produce images so as to fool the discriminator into "thinking" that its output is sampled from the real data distribution. During training D is shown a real image, and adjusts its parameters to make its output lower. Then D is shown an image produced from G and adjusts its parameters to make its output D(G(Z)) larger (following the gradient of some objective predefined function). But G(Z) will train itself to produce images so as to fool D into thinking they are real. It does this by getting the gradient of D with respect to Y for each sample it produces. In other words, its trying to minimize the output of D while D is trying to maximize it. The loss function that we are optimizing is:

$$\min_G \max_D V(G,D) = \mathbb{E}_{h\sim p_{data}(h)}[\log(D(h))] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))], \quad (1)$$

Radford *et al*. [11] extended the idea of GAN and proposed DCGAN by replacing fully connected layers with convolutional layers, making generator and discriminator convolutional neural networks instead. Work by Schuler *et al*. [8] proposes a method to learn direct mapping from masked regions to their completed counterparts. Finally, Yeh *et al*. [13] proposed a method to use GANs to complete masked regions of images, and their method of image completion is elaborated in our method section as we formed our GAN based approach image completion algorithm following this work. Specifically, the work by Yeh *et al*. perfectly only considers masking information in the test time as opposed to other techniques that require mask related information like size and location during the training phase, thus making the trained networks more generalizable.

3. Training Data and Pre-processing steps

The GAN formulation does not explicitly require the entire training data distribution to have minimal variation. However conventional wisdom has been that there is a trade-off between image quality and variation. Class variation and quality of generated images has been studied in detail by Karras *et al*. [7]. Thus for our task we have used just human face images from the Large-scale CelebFaces Attributes (CelebA) Dataset [10] and Labeled Faces in the Wild(LFW) [6]. These images were cropped to size 64×64 and the faces were aligned such that the eyes and nose were in the center of the frame. This was done using the OpenFace framework [1].

In section 5 we also look at completion of images when the generative model has been trained on data with high class variation. For this we make use of the CIFAR 10 dataset [9]. These images were of dimension 32×32 and were used in its raw form and did not undergo any sort of preprocessing.

4. Approach

4.1. Problem Statement

To define the problem statement, we first define some notation for completing images. We use a binary mask M (of the same spatial dimensions as that of the target image)that has values 0 or 1. A value of 1 represents the parts of the image we want to keep and a value of 0 represents the parts of the image we want to complete. We can now define how to complete an image x given the binary mask M. Next, suppose we've found an image from the generator $G(\hat{z})$ for

Figure 2. Some of the images in the Celeb-A dataset. (Image credits:http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html)

some z that gives a reasonable reconstruction of the missing portions. The completed pixels $(1 - M) \odot G(\hat{z})$ can be added to the original pixels to create the reconstructed image:

$$x_{\text{reconstructed}} = M \odot x + (1 - M) \odot G(\hat{z}) \qquad (2)$$

The final task is to train a generator $G(\hat{z})$ which is able to produce a realistic $x_{\text{reconstructed}}$. To do this we will have to define a loss function that incorporates the contextual information as well as output an image that's of good perceptual quality.

4.2. Adversarial training of the generator $G(\hat{z})$

As the first step in image completion, we need a generator network that has been trained in tandem with a discriminator network according to the adversarial training procedure given in [5] wherein the optimization objective is given in (1).

The generator network takes in a hidden state vector z of dimension 100 and whose elements are drawn from a uniform distribution. This vector is then transformed into a tensor of dimension 4×4×512 by a matrix multiplication to form the start of the convolutional stack. This stack is then filtered with kernels of size 5×5 and the number of filters progressively decreases by a scalar factor until there are only 3 channels. The spatial dimension of each stack increases by virtue of fractionally strided convolutions. For the non-linearity we apply a ReLU activation and for the final layer a tanh activation is applied in the generator.

The discriminator inputs the 64×64 sized images from either the generator or from the training set. The discriminator uses a stride of 2. This down-samples the activations at every stage. Finally after downsampling at every stage, the last convolution layer is flattened and then fed into a single sigmoid output. Leaky ReLU activation was used as the non-linearity in the discriminator. Batch normalization was used in both the discriminator and generator.

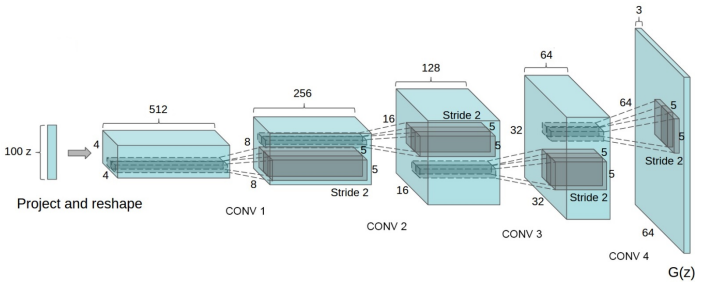

Figure 3. Generator Architecture (Image Credits: Radford *et al.* [11])

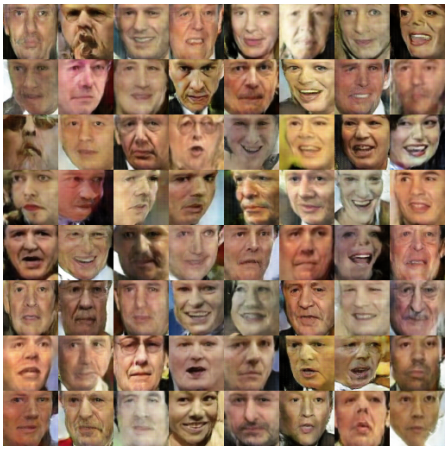

Figure 4. Some of the faces generated after 20 epochs.

These two networks were trained on approximately 200,000 face images for about 20 epochs. Figure 4 shows some faces generated by the generator after this training process. In the ideal case we would want the distribution p_g from which the generator is able to sample images from, to be as close as possible to the original data distribution p_{data}.

4.3. Constrained Generation

The generator G is now able to take a point z drawn from p_z and generate an image mimicking samples from p_{data}. It is hypothesized in [13] that if G is efficient in its representation then an image that is not from p_{data} (incomplete images) should not lie on the learned encoding manifold, z. Therefore, we aim to recover the encoding $\hat{z}$ "closest" to the corrupted image while being constrained to the manifold. Thus we want to find $\hat{z}$-

$$\hat{z} = \arg\min_z \{L_c(z|x, M) + L_p(z)\} \qquad (3)$$

Here L_c is the contextual loss and L_p is the prior loss.

$$L_c(z|x, M) = \|M \odot x + M \odot G(z)\|_1 \qquad (4)$$

M is a binary mask in which the locations of one's denote the locations of uncorrupted pixels, x is the incomplete image and G(z) is the generated image. By enforcing this perceptual loss we are penalizing the generator for generating pixels values that are dissimilar to the ones in that are defined in the incomplete image.

$$L_p(z) = \lambda log(1 - D(G(z))) \tag{5}$$

$$L = L_p + L_c \tag{6}$$

The perceptual loss is identical to the generator's loss during adversarial training.λ is a parameter to balance between the two losses and is set to 0.1. Without to this loss the generated image will converge to something that is perceptually unrealistic.

With these losses set we compute the gradient with respect to the parameters in G and backpropagate the errors into z. z is then updated along the negative of the gradient with respect to the loss. This moves z to the optimal $\hat{z}$. We then generate $G(\hat{z})$ which contains the generated image regions for the incomplete regions. To reconstruct the completed image we use expression (2).

4.4. Issues while Training

We faced a couple of issues while training our GAN. They are unstable and finding the right hyper parameters and architecture details were a bit tedious. The other issue was that one fo the generator or discriminator would effectively become better than the other. One example is when the discriminator gets too strong, the generator is unable to fool the discriminator resulting in the adverserial loss going up. On the flip side, when the generator gets mich better and is always able to fool the discriminator, learning slows down and negatively affects our results.

Based on cues from [5] and [12] we modified out learning procedure as follows:

1. The loss function to optimize G is $\min(\log(1 - D))$, but in practice we used $\max \log(D)$. While both formulations are equivalent, the latter does not suffer from vanishing gradients.

2. Label Smoothing, i.e. if you have two target labels: Real=1 and Fake=0, then for each incoming sample, if it is real, then we replaced the label with a random number between 0.7 and 1.2, and if it was a fake sample, we replaced it with 0.0 and 0.3.

3. Lastly in an attempt to make the discriminator more robust and improve generalization, labels were made noisy for the discriminator: occasionally the labels where flipped when training the discriminator.

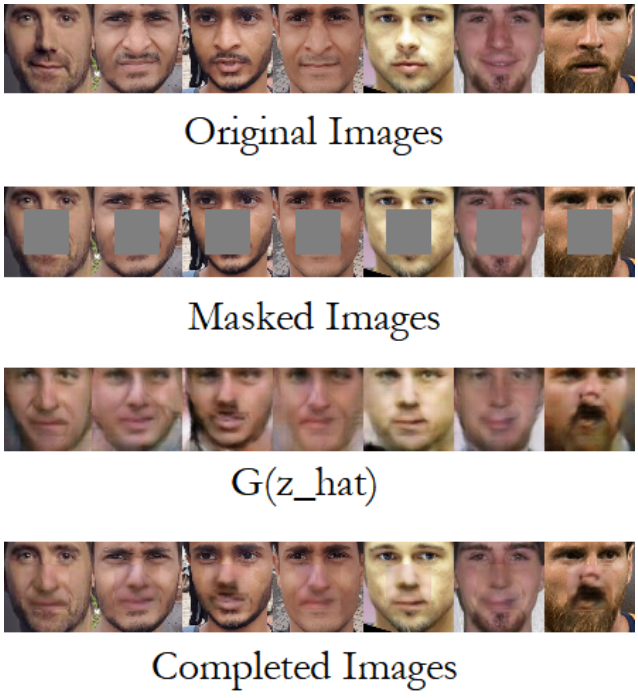

Figure 5. Reconstructed face images from the corresponding incomplete images. The completed images are reconstructed by adding the output of the generator $G(\hat{z})$ along with the the masked image according to equation (2)

Figure 6. The generator network trained on human faces trying to complete dog faces.

Figure 7. The generator network trying to complete images when trained on the CIFAR-10 dataset.

5. Results and additional experiments

From Figure 5 we can see that the generator performs well in completing most face images. It is to be noted that none of the incomplete images shown were taken from the

training data. In Figure 6 we see that if we attempt to complete images that are significantly different from what was in the training set, then the completed images are expectedly of poor perceptual quality.

We also trained the generator-discriminator pair on the CIFAR-10 data-set. As expected the performance of the model to complete faces is significantly worse off than when it was solely trained on face images. Even when trying to complete images taken from from the training data, the model fails to perform well as there was too much variation in the training data. The distribution over the training images is just too complex for the the generator model to learn.

6. Conclusion

This course and the accompanying project provided us a chance to experiment with an area of deep learning; deep convolutional genarative adverserial networks as applied to the task of semantic image inpainting. While the experiments we conducted showed promise, we believe that there exists certain drawbacks to using GANs, the most prominent one being the inference performance of our model depends extensively on the training procedure and the architecture of the generative model. There exist certain cases where the model is unable to find the appropriate $\hat{z}$ in the latent image manifold. Further, while the current GAN model we implemented works well for small images like faces, it may not be capable of representing more complex scenes as seen in the real world without making the model much larger; the training procedure would also require significant tweaking.

This leads us to the question, do GANs always converge? As a result of our experiements, we can say that GANs are difficult to train. However it is our opinion that on small problems, GANs sometimes converge and sometimes do not. The question then revolves around how to effectively train GANs on certain classes of objects(apart from faces) and on large images. This is probably the most important question about GANs, both in terms of theory and practice. In terms of theory, it would be great to derive a set of conditions under which they converge or dont converge. In terms of practice, it would be great to modify them in a way that makes them converge consistently.

7. Acknowledgement

We would like to thank Prof Miller for providing us with this great learning opportunity as well as our TAs Pia and Hang for their guidance.

References

[1] B. Amos, B. Ludwiczuk, and M. Satyanarayanan. Openface: A general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science, 2016.

[2] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. In *ACM Transactions on graphics (TOG)*, volume 26, page 10. ACM, 2007.

[3] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 417–424. ACM Press/Addison-Wesley Publishing Co., 2000.

[4] A. Criminisi, P. Pérez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on image processing*, 13(9):1200–1212, 2004.

[5] 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.

[6] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[7] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[8] R. Köhler, C. Schuler, B. Schölkopf, and S. Harmeling. Mask-specific inpainting with deep neural networks. In *German Conference on Pattern Recognition*, pages 523–534. Springer, 2014.

[9] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009.

[10] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[11] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[12] 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.

[13] R. Yeh, C. Chen, T. Y. Lim, M. Hasegawa-Johnson, and M. N. Do. Semantic image inpainting with perceptual and contextual losses. *arXiv preprint arXiv:1607.07539*, 2016.