Improved Logo Generation Using Super Resolution And Progressive Growing of GANs

Chad Maycumber
University of Central Florida
4000 Central Florida Blvd, Orlando, FL
Chad.maycumber@knights.ucf.edu

Jay Patel
University of Central Florida
4000 Central Florida Blvd, Orlando, FL
jaypatel09@knights.ucf.edu

Shubham Sethi
University of Central Florida
4000 Central Florida Blvd, Orlando, FL
shubhamsethi1010@knights.ucf.edu

Abstract

Logo designing is a very cumbersome job which requires hiring a graphic designer to design logos and sometimes these designed logos can be quite different from what client expects, due to which designer has to backtrack and restart from scratch. We plan to explore the possibility of creating an improved process for generating logos using Generative Adversarial Networks. To improve the logo generation process we are building off of previous work using GANs to create logos of a latent vector space to explore the wide variety of potential logo combinations and allow for an end user to have more options. We want to improve upon this idea by employing a new GAN technique known as Progressive Growing GANs to see if it can improve the overall quality of the generated images. We also test the idea of using Super Resolution to up-scale lower resolution images to help improve the training time in GANs without losing image quality. Using Using PGANs we were able to improve logo quality and show that Super Resolution can improve training time without loss in image quality.

1. Introduction

Improvements to Generative Adversarial Networks [4] have started to give rise to a wide range of applicable applications for them. Although other methods have been proposed for image generation including Variational Autoencoders and PixelCNNs, GANs have been seen to produce superior results in terms of both detail and image sharpness. This improved image quality has made GANs the go to in terms of producing images and new advances has bridged the gap even further between this technology and applicability in the real world.

Building upon this idea we set out to see if it is possible to generate high quality logos using the Progressive Growing of GANs [10] algorithm created by researchers at NVidia. The original goal of this paper was to create a method of using GANs to produce high resolution images with improved training speed and stability. There algorithm produces state of the art results on the CIFAR-10 [7] data set with inception scores [9] of 8.80 which is the best results so far for any unsupervised learning method. We chose to build on this algorithm for this reason because it improved upon the inception score generated in previous work using Improved Wassertien GAN [5] and DCGAN on logo generation. Another feature that is essential for logo generation with GANs is the ability to move over the latent space to transition between different logos as pointed out by the authors of ”Logo Synthesis and Manipulation with Clustered Generative Adversarial Networks” [**?**], PGANs [10] also includes this ability making it a perfect algorithm to explore the idea of logo generation.

Taking advantage of the advanced nature of the PGANs and it’s ability to produce photo realistic images we strive to create superior quality logos that could be used for real world purposes. We plan to test the hypothesis that using the PGANs we can produce logos of superior quality with the same data set compared to other methods using the iWGAN [5] and the DCGAN.

We also chose to explore the idea of adding a Super Resolution algorithm [11] at the end of our GANs model to help us reduce the resolution we need to train the network on and produce images with similar quality at our desired resolution. In this regard we explore the ability of the Progressive Growing of GANs to produce high quality images at a lower resolution and then upscale these images to a higher resolution using the super resolution algorithm, essentially creating higher resolution images with a far less amount of training expense by leveraging two seperate networks to carry out each individual aspect of the goal. The GAN is primarily used for the generation of the image, and rather than training at a high resolution we use the Super Resolution algorithm to upscale the results and produce quality image on par with the same images produced with the GAN originally at the desired resolution.

2. Related Work

Recent research shows that the GANs can have different architecture based on the flow of data throught the neural network. Based on that, some of the most famous GAN architectures are DCGAN, iWGAN, etc.

The paper Logo Synthesis and Manipulation using Clustered Generative Adversarial Networks by Alexander *et al.* [1], shows that DCGAN and iWGAN can be trained on a clustered Large Logo Dataset(LLD) [2]. Training GANs on the LLD can be a bit tricky or unstable without clustering the dataset available. So, we use the provided clustered dataset from the site: `https://data.vision.ee.ethz.ch/cvl/lld/` [2].

2.1. Dataset

As mentioned above, for all our experiments we use the LLD dataset made available at `https://data.vision.ee.ethz.ch/cvl/lld/` . This website hosts 2 datasets for GAN training.

LLD-icon: Favicons This dataset contains approx. 480k images of uniform 32x32 pixel size. This images have been run through ResNet Classifier/ AutoEncoder Classifier for clustering. The clustered datasets are available on the site. This dataset was put together by crawling through Alexa's top 1-million sites for Favicons.

LLD-logo: Twitter This dataset contains approx. 120k images of uniform 400x400 pixel size. This dataset is also made available after clustering. Dataset was put together by crawling through *Twitter* for company logos.

2.2. GAN Architectures

Initially we refered to the paper by Alexander *et al.* [1], to understand how GANs can be used to synthesis logos and how different GANs provide different output based on their architecture.

DCGAN For all of our experiments on Deep Convolutional GAN or DCGAN, we used Taehoon Kim's TensorFlow Implementation [6]. From the paper on Logo Synthesis,we found out DCGAN is unstable for handling large dataset with unsupervised clustering,so we haven't trained any model of DCGAN for our experiments instead used them in understanding GAN architectures.For our interested readers, we have including the reference the github repository for TensorFlow implementation.

iWGAN For all of our experiments on Improved Wasserstein GAN, we used official TensorFlow Repository by Gulrajani *et al* [3].

Training the DCGAN or iWGAN [8] architectures seemed sometimes unstable or took too long. Progressive Growing of GANs (PGANs) helps with lessening the training time and increases the stability of the system exponentially.

PGAN For all of our experiments on PGAN, we used TensorFlow Implementation of Progressive Growing of GANs by Tero Karras *et al.*

3. Progressive Growing of GANs

In the Progressive growing of GANs algorithm the authors explored a revolutionary way to improve the overall image quality of higher resolution GAN images. We wanted to apply this state of the art GAN technique to the logo data set explored in our original paper to improve the quality of the image output, potentially creating an extremely feasible solution to for designers to produce images with across a latent space.

Inception Score	Model
8.80	Progressive Growing Of GANs
8.625	iWGAN-AC with RC clustering
7.799	iWGAN-LC with RC clustering

Table 1. Inception Score of GANs for logo generation

The primary improvements over the normal algorithm of the progressive growing of GANs is the new GAN architecture that the authors employ. Rather than training the generator and discriminator at static resolutions they chose to gradually increase the size of the generator, discriminator and the produced images from low to higher resolutions.

To accomplish this the authors employ a technique that starts the image at extremely low resolutions trains gradually increasing the size of both the generator and the discriminator networks. This gradual increase in image size helps the GAN maintain stability as the resolution of the produced images increases because the image being produced by the generator is quickly recognized by the discriminator and both networks are being train highly effectively at these lower resolutions because the class models are significantly less complex than at higher resolutions. The gradual increase allows the network to gradually learn to increase the resolution and has a solid understanding of each resolution at the previous step allowing it to improve the quality of the produced images.

Employing this technique has a variety of benefits, the most obvious being the noticeable increase in image quality and the GAN stability at higher resolutions. Increasing the stability has allowed them to synthesize images of greater scale with both WGAN-GP loss and LSGAN loss. This was the primary reason that we chose to run this algorithm on the logo dataset collected by the previous authors. Another benefit to this gradual resizing of the produced images is the ability to speed up the overall training time. By gradually move from lower to higher resolutions the authors found a way to improve the training time by avoiding an extremely deep network that has to train at a resolution the same as the output image. This is a result of the network being built upon lower resolution layers that drastically reduce the overall number of hyperparameters in each layer.

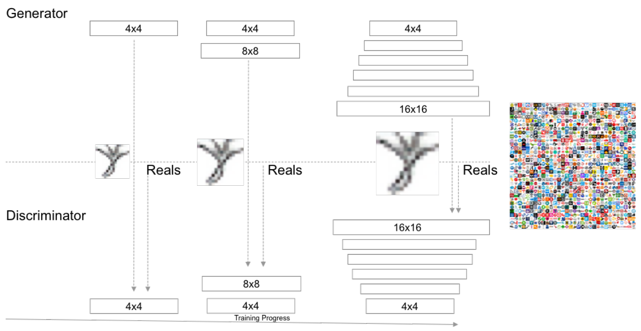

Figure 1. This process starts with both the generator and discriminator having resolutions 4 x 4. As the training progresses they add layers to both the generator and discriminator at the same time leaving all the layers trainable. This gives them the ability to create high resolution images and improve training time.

Another new technique that the authors created to help improve the overall variation of the GANs is the way they applied feature maps produced by finding the standard deviation of each minibatch features to produce an additional feature map which is later applied to the image at the end of the discriminator. They found that this method coupled with adding in a down-sampled version of the original image while moving between the different image resolutions helped gradually move between these different resolutions. Combining these methods they were able to improve the variability and maintain the stability of the GAN with gradual increases between resolutions.

The authors also entertained the idea of improving the variation even further by leveraging other methods suggested by multiple authors. They realized this could have an important impact by either working in parallel to improve, hinder or create results that differ very little from the results they achieved in their paper. However they left the exploration of these separate methods for improving variation for another time.

4. Enhanced Super Resolution GAN

To improve the overall training time and stability of the GAN network even further we have included the idea of adding a pre-trained Super Resolution algorithm as a final step at the end of the GAN to allow for training at lower resolution images and maintain a similar level of quality at higher resolutions.

For our Super Resolution algorithm we chose to use Enhanced Super Resolution GAN which is a super resolution technique using a generative adversarial network that has shown to produce state of the art results in terms of sharpness and detail when scaling. This method is based of SRGAN with some slight improvements to architecture which has produced better results.

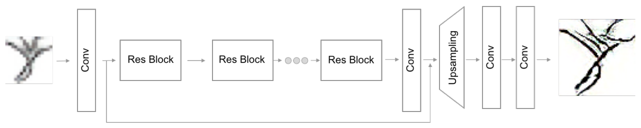

Figure 2. This architecture is based of the basic SRResNet. Most of the computation is computed in the lower resolution feature space, they then select different types of residual blocks to improve the performance of there algorithm. We use there algorithm to up-scale the lower resolution images to higher resolution of similar quality.

There primary improvements to the architecture of the generator include replacing the original blocks with Residual-in-Residual Dense Blocks which combines Residual-in-Residual blocks and dense blocks. This was able to improve the quality of the output from the original SRGAN. They also chose to remove the BN layers to improve the performance of the algorithm finding that these BN layers increased the number of artifacts when applying super resolution to images. These layers were found by the authors to be relatively harmful rather than helpful when analyzing there results because of these artifacts decreasing the final quality of the image.

They also observed that more layers usually lead to increased performance inside of there Residual blocks. Keeping the rest of the SRGAN architecture relatively the same there addition the Residual-in-Residual Dense Blocks were a main reason for there visual increase.

To improve the stability of there deeper network they use two techniques. Residua scaling which applies a constant between 0 and 1 to the end of the residual blocks before adding them to the main path to increase the stability of the network and also using smaller initializations because they help when training deep residual networks by making them easier to train.

Beyond improving upon the original SRGAN generator they also chose to modify the discriminator based on the architecture of the Relativistic GAN. In contrast to the normal discriminator employed in SRGAN where the discriminator determines the probability that a real image is real and natural, the raGAN discriminator tries to determine the probability that a real image is relatively more real than a fake image.

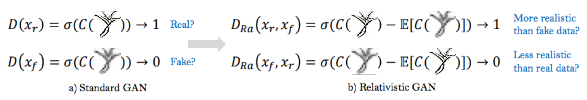
$D(x_r) = \sigma(C(\ ))\rightarrow 1$
Real?
$D(x_f) = \sigma(C(\ ))\rightarrow 0$
Fake?
a) Standard GAN
$D_{Ra}(x_r, x_f) = \sigma(C(\ ) - \mathbb{E}[C(\ )]) \rightarrow 1$
More realistic than fake data?
$D_{Ra}(x_f, x_r) = \sigma(C(\ ) - \mathbb{E}[C(\ )]) \rightarrow 0$
Less realistic than real data?
b) Relativistic GAN

Figure 3. The raGAN compares the relative fakeness compared to a real image.

The benefits of this include being able to train images that benefit from both the gradients of the real images and the fake images. This improves the overall quality of the resulting image improving the sharpness and detail because it tries to more closely match the original image, rather than a straight comparison between the fake image and the real image.

5. Proposed Method

We propose a new method of binding together the Progressive Growing GAN and the Enhanced Super Resolution GAN in an ensemble fashion to produce higher resolution images at a better training speed. This is possible because we are able to train images on a lower resolution and up-scale them to the desired image without any significant loss in quality.

To do this we first train the network on a dataset that is at a scale smaller to the relative super resolution, preferably reducing the original training data by a factor of one-fourth the original images. Then the Progressive Growing GANs is trained on these lower resolution images to produce images of a lower resolution. Finally, when the Progressive Growing GAN is finished training we scale the output of the GAN by a preferred factor of 4x with the ESRGAN which puts the final output of the network back to the original image size.

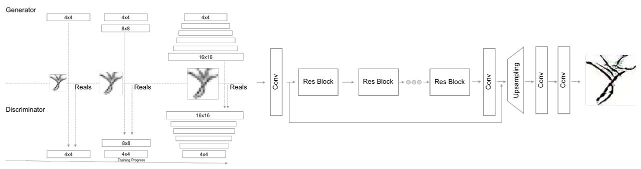

Figure 4. This architecture shows the combination between the Progressive Growing of GANs algorithm and the Enhanced Super Resolution GAN to produce sharp images of similar quality with same training time.

The benefit to this includes theoretical extreme boost to training times because the network is trained at a significant smaller resolutions. With this method we can reduce the total number of pixels in each image being trained by the GANs up to 16x the original image, leading to a drastic reduction in the total number of parameters in the network.

This method would theoretically work better with images that start at a higher resolution to begin with being that the Super Resolution has more items to work with to scale that image more accurately. A problem with extremely small images, such as those 32x32 is that one-fourth the image would create an extremely low resolution that might be hard to scale with Super resolution. Therefore for these lower resolutions we recommend using images that are at a scale one-half the original resolution, scale it 4x with Super Resolution then finally down-scale the image to the original input images resolution.

6. Experiments And Results

To experiment we chose to compare the outputs of the Progressive Growing GAN trained on the LLD-icon dataset which includes images of resolution 32 x 32 with logos trained on the smaller resolution Modified LLD-icon dataset scaled down by a factor 2 to a resolution 16 x 16 up-scaled by a factor of 4x to 64 x 64 with super resolution then downscaled to the original image size of 32 x 32.

To start the experiment we rescaled the original dataset then trained each network individually keeping other factors the same other than the minibatch size which was changed to 128 for the smaller resolutions vs 64 for the higher resolution. Keeping most factors the same helped us to determine if our method is an effective way of producing high quality high resolution GAN output from lower resolution data.

6.1. Training Time

We observed that by decreasing the networks input images by a factor of 2 significantly reduced the total training time of the algorithm. The original LLD-icon dataset took a total time of 23h 59m 57s, while the LLD-icon dataset modified to one-half the original was finished is a time of 4h 59m 30s. This is almost 5x the speed of the original output, therefore a significant improvement in the total training time as should be expected from a smaller resolution input.

Training Speed	Model	Comments
23h 59m 57s	PGANs	32 x 32 Logos
4h 59m 30s	PGANS + Super Res	16 x 16 Logos

Table 2. Training Speed Of GAN Models

6.2. Results Logo LLD-icon Dataset

We then applied our method of combining the Super Resolution algorithm to the produced data. Therefore we scaled and enhanced a random 16 x 16 logo to size of 64 x 64 with ESRGAN, then downscaled this image back to a size of 32 x 32 to compare the image to the a random logo from the original results.

This resulted in an image that is very similar to the random image produced in the 32 x 32 image results. This is very beneficial because of the greatly improved training speed and similarity between the resulting images, showing that very little image quality is lost with our method.

7. Conclusion

Overall we were able to show that there could be some potential for using super resolution to improve GAN results with decreased amounts of training time. Although we do recognize that it has some obvious limitations in terms of up-scaling the image without a complete loss in quality. However using super resolution techniques along with GANs could be a effective method in decreasing training time outside the sole use at the end of the network.

7.1. Potential Improvements

Some things that we entertained was the ability to improve upon the Progressive GANs architecture by adding in a Super Resolution not only at the end of the network, but resizing each layer with a Super Resolution algorithm to further improve the quality of the output produced by the GANs and potentially lead to a faster convergence. One drawback to this could be a slight impact on performance because it would require for the each image produced at each layer to be based through another network to produce the next results. However this could be worth the extra training time if the images were to converge relatively faster and thus increase the quality and lead to a smaller overall training time even though it takes longer to pass through each individual layers. This idea could be taken even further if the Super Resolution algorithm itself was trained on the same data set used by the GAN because this would produce highly accurate super resolution results over a pretrained model on a different data set, either we believe this could be very impactful for expanding on the progressive growing of GANs architecture.

References

[1] R. T. L. V. G. Alexander Sage, Eirikur Agustsson. *Logo Synthesis and Manipulation with Clustered Generative Adversarial Networks*.

[2] Database. *Large Logo Dataset*. 2007.

[3] I. Gulrajani. iwgan-tensorflow.

[4] M. M. B. X. D. W.-F. S. O. A. C. Y. B. Ian J. Goodfellow, Jean Pouget-Abadie. *Generative Adversal Networks*. 2007.

[5] M. A. V. D. A. C. Ishaan Gulrajani, Faruk Ahmed. *Improved Training of Wasserstein GANs*. 2007.

[6] T. Kim. Dcgan-tensorflow.

[7] A. Krizhevsky. Cifar-10 dataset link.

[8] L. B. Martin Arjovsky. *Towards Principled Methods for Training Generative Adversarial Networks*.

[9] R. S. Shane Barratt. *A Note on the Inception Score*.

[10] S. L. J. L. Tero Karras, Timo Aila. *Progressive Growing of GANs for Improved Quality*.

[11] S. W. J. G. Y. L. C. D.-C. C. L. Y. Q. X. T. Xintao Wang, Ke Yu. *ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks*.

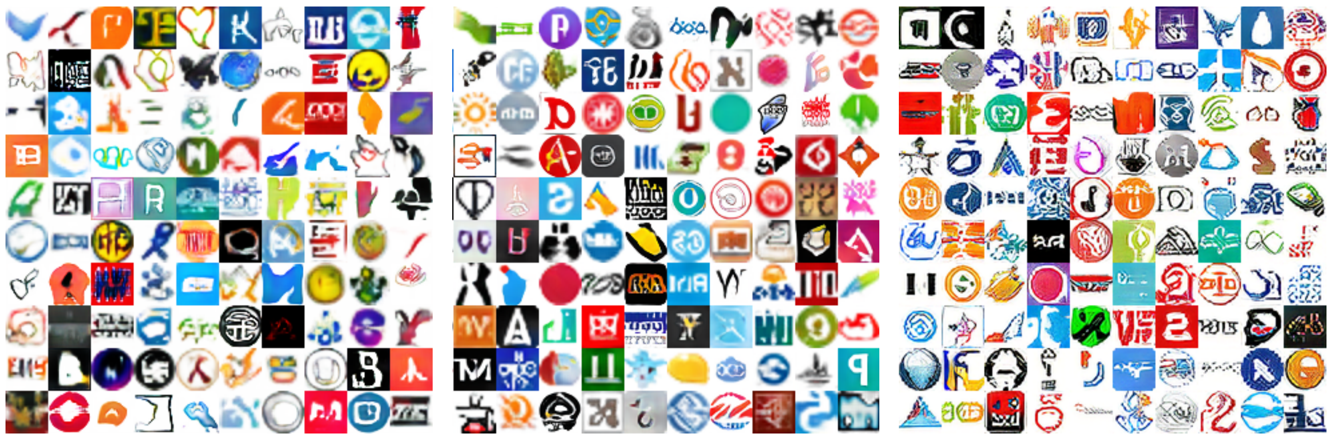

Figure 5. Comparisons of Result from our runs on different GANs. First Image shows our outputs from iWGAN. Second Image shows our output from PGAN. Third Image shows our output from PGAN + SRES architecture. We provide 32x32 pixel images as input for training on iWGAN and PGAN,while we provided 16x16 pixel images for PGAN + SRES and got 32x32 pixel images as output. All the outputs are at 12k iteraions.