CS230

# Applying Conditional Generative Adversarial Neural Networks (cGANs) to Generate Realistic Microarray Gene Expression Data

**Lawrence Bai**
Immunology Program, Stanford University School of Medicine, Stanford, California, USA
lawrence.bai@stanford.edu

**Madeleine Scott**
Biophysics Program, Stanford University School of Medicine, Stanford, California, USA
scottmk@stanford.edu

## Abstract

High-throughput technologies have allowed researchers to uncover biological differences between normal and diseased cells. Examining changes in gene expression, in particular, has led to discoveries of novel disease mechanisms and therapeutic targets. However, this data requires patient tissue samples obtained through invasive biopsies. Such biopsies are a financial burden to the researcher, and can lead to physical and data privacy issues for the patient. To address these shortcomings, we used conditional GANs to generate realistic cancer patient sample gene expression data using publicly available microarray data. We demonstrate that generated data can be used identify biologically relevant differences between cancer and healthy tissue that would have otherwise been lost.

## 1 Introduction

Gene expression defines the phenotype of a cell, as it is the consequence of cumulative genetic and epigenetic alterations. The DNA microarray was developed in the late 1990s as a method of measuring the expression of all genes in a sample. The NCBI Gene Expression Omnibus (GEO) is a repository of human microarray datasets that are derived from samples from tissue biopsies or blood [Edgar et al., 2002]. The conditions studied range across all categories of diseases, including chronic diseases, infectious diseases, and cancer. One fundamental problem in biomedical research is the low number of observations available for each disease of interest, mostly due to a lack of available biosamples, high costs, or ethical reasons. Given these barriers, we believe that augmenting real observations with generated *in silico* samples could lead to more robust and reproducible analysis results. We developed conditional single-sample generative adversarial neural networks (cssGANs) for the realistic generation of microarray samples. This work was based on a recently published method that generated single cell RNA data [Marouf et al, 2018]. With our method, it is possible to augment sparse sample populations which in turn can improve downstream analyses such as biomarker detection and improvement of classifiers while reducing the number of patient samples and thus research costs. In particular, the generation of sample-level gene expression data could improve biological analyses for rare diseases, in which samples are scarce.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

## 2 Related work

In practice, *in silico* generation has been successful in computer vision when used for 'data augmentation', where these generated samples can artificially increase the number of observations [Shrivastava et al., 2017]. Current methods of choice for photo-realistic image generation use Deep Learning-based Generative Adversarial Networks (GANs) [Goodfellow et al, 2014; Isola et al, 2017] and Variational Autoencoders (VAEs) [Kingma et al, 2013]. GANs involve a generator that learns to output realistic in silico generated samples, and a discriminator that learns to spot the differences between real samples and generated ones. The 'adversarial' training procedure allows for these two neural networks to compete against each other, ideally until the discriminator cannot distinguish original from generated data. The development and usage of GANs and VAEs for omics data augmentation is scarce. As a proof of concept, Marouf et al. has focused on the generation of single cell RNA (scRNA) sequencing data. Using their method, they were able to successfully generate data for distinct immune cell subsets found in peripheral blood.

## 3 Dataset and Features

We mined the GEO database in order to collect tissue biopsy microarray gene expression samples from cancer patients and healthy tissue controls. In total, we collected over 7000 samples from across more than 50 datasets in 4 different cancers (colorectal, breast, pancreatic, and lung). For our proof-of-concept, we focused on colorectal cancer as it had the most amount of data (2945 samples across 27 datasets). Since these datasets were obtained across a heterogeneous landscape (e.g geographic location, hospital setting, platform technology, etc.), we use a well-established batch-normalization technique, ComBat [Johnson et al, 2006], before proceeding to our model. After intersection across genes, we are left with a little over 3000 genes for data augmentation.

## 4 Methods

*Conditional Single-Sample Adversarial Neural Network (cssGAN)*
We adopted the model from the original authors' paper [Marouf et al, 2018] and modified their open-sourced code to fit our data. In their paper, their best-performing model was a GAN that minimized the Wasserstein distance [Arjovsky et al, 2017], using two fully-connected (FC) neural networks with batch normalization (Figure 1). The generator is an FC network with three hidden layers of increasing size (128 to 1024 nodes), each layer containing batch normalization and ReLU activation; the output is a library-size (number of genes) normalization layer. The discriminator is also an FC network with three hidden layers, but of decreasing size (1024 nodes to 1 node) with a final output of 1 node.

*ComBat Normalization*
Each single cell RNA experiment generates data from over 50,000 cells. However, microarray experiments often profile only ten or twenty samples at a time. Therefore, we needed an additional strategy to combine multiple microarray datasets to obtain enough training data. ComBat normalization has been shown to remove batch effects in microarray datasets inherent to heterogeneous conditions such as different platform technologies, geographic location, etc. ComBat uses empirical Bayes to account for adjusting batches with small sizes (since biological samples are difficult to obtain, many groups do not have the luxury to collect thousands of samples). We use ComBat to normalize across each specific tissue, leaving distinct clusters by tissue type, but eliminating sub-batches within a specific cancer.

## 5 Experiments/Results/Discussion

*Hyperparameter tuning*
We first adopted the final learning model from Marouf et al. as described above. We then tuned different hyperparameters to attempt to improve the baseline training model. We changed learning rate logarithmically from 0.1 to 0.00001, and our final model used a learning rate of 0.0001, decaying to 0.00001. We also tried tuning batch size in a log(2) scale (2 -128), and we chose 128 as our batch size. We left our distance metric as Wasserstein distance, given literature on the value of using this compared to traditional GAN loss functions.

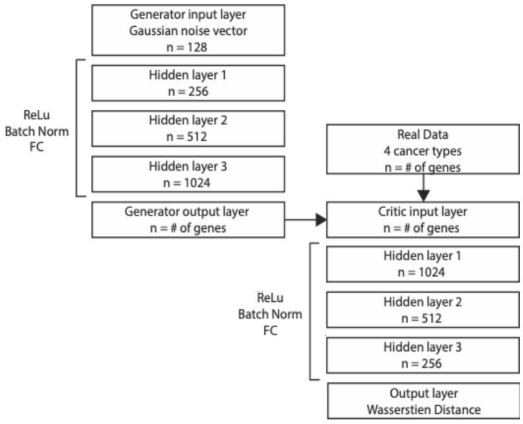


Figure 1. cGAN workflow adopted for microarrays. We use two fully-connected neural networks for our generator and discriminator. In the generator, there are three hidden layers of increasing size and vice-versa for the discriminator.

**Model evaluation**

We recognize that evaluating the quality of generative models can be difficult, especially in biomedical data in which we hope to preserve the underlying biological architecture (i.e. gene-gene interactions) in generated data. Thus, we used a variety of qualitative and quantitative measures to evaluate our generated data.

*Model Evaluation: Clustering using tSNE and gene-gene correlation*

One of the first evaluations we made was to qualitatively visualize how well the generated data clustered after training. We use uniform manifold approximation and projection (UMAP) [McInnes, Healy, and Melville 2018] to visualize the real and generated data. In our real data, healthy samples formed a distinct cluster next to diseased samples. Our generated data was able to recapitulate these cluster structures (Figure 2A, top right). More importantly, the clustering was not due to a random generation of features added to distinguish the two clusters; CLCA4, a gene highly expressed in the colon, retained its expression gradient in the generated data (Figure 2B, bottom right). This suggests that gene-gene interactions and networks were learned and retained when generating new data. Inter-gene dependencies and correlations are vital biological architecture that are needed to understand gene-regulatory networks. We first randomly split the real data and plotted correlation values for each gene, resulting in a high positive correlation (R= 0.97; Figure 2B, top). Comparing all generated data versus real data, we retained high correlation (R = 0.94; Figure 2B, bottom), further corroborating that our cssGAN was able to create highly synonymous data.

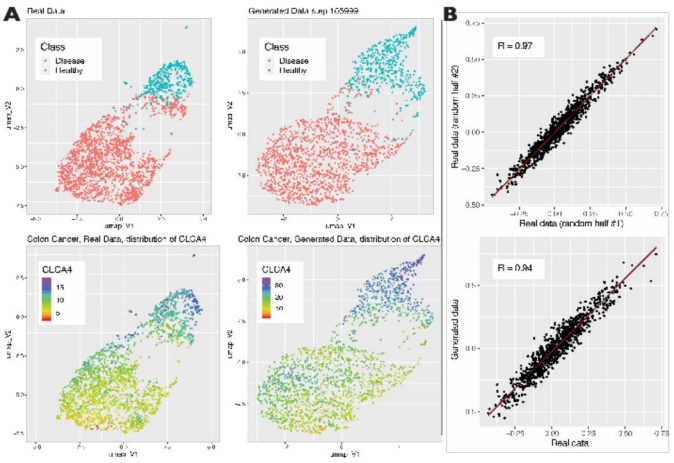


Figure 2. Model evaluation using clustering methods and gene-gene correlation. (A) Using a clustering algorithm, UMAP, we compare the nonlinear combination of features that define disease and healthy groups and compare the clustering pattern of real data with generated data. (B) We look at the correlation values of gene expression between real data and generated data, where one point is a single specific gene.

*Model Evaluation: classification performance*

Next, to more quantitatively evaluate our neural network model, we use a Random Forest classifier using 5-fold cross-validation and three mtry values to classify generated *vs* real data. We measured

evaluated the random forest model with area under the curve (AUC). The ability of the random forest model to distinguish between real and generated degrades over training time (Figure 3). In other words, our discriminator slowly loses the ability to distinguish generated data from real data over time. This is important, as we want our generated data to be representative of the distribution of real-world data (assumed to be based on our real data).

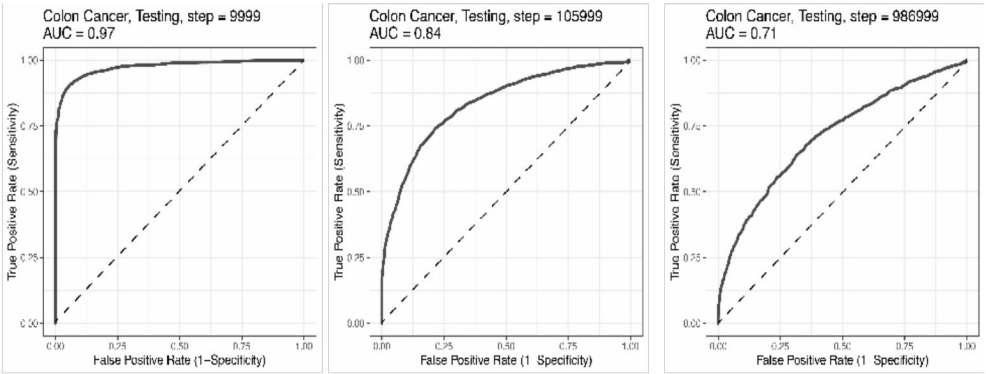


Figure 3. Model evaluation using random forest classifier to predict accuracy of discriminator during training. In the beginning of the model, the discriminator easily distinguishes generated data from real data. Over the course of training the cGAN, it becomes increasingly harder for the discriminator to correctly distinguish generated and real data, as measured by ROC.

*Model Evaluation: Downsampling Analysis*

Finally, we evaluated the biological usefulness of the generated samples when compared to real data. We downsampled the number of healthy samples to recapitulate a potential real-world scenario in which large number of samples are difficult to obtain. We then trained the cssGAN this downsampled dataset. We performed differential gene expression analysis using log-fold change in three scenarios: 1) using all real data (ground truth); 2) data with small number of healthy samples; 3) data from 2) but replenishing healthy samples with data generated from the downsampled data (Figure 4). The downsampled data alone is unable to correctly identify genes that are different in disease and healthy samples. However, using generated healthy samples from the cssGAN trained on the downsampled data, we were able to regain the same list of differentially expressed genes as ground truth. We also qualitatively evaluated the analysis using UMAP clustering (Figure 5). When downsampled, healthy samples no longer maintain their own distinct cluster (Figure 5, middle), but adding generated healthy samples can bring back the two distinct clusters. Interestingly, the model was able to generate healthy data even during extreme class imbalance between the two groups, suggesting that the network is able to learn gene-gene interactions from the entire dataset.

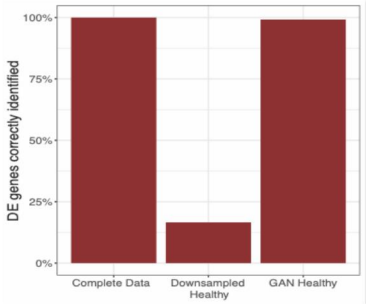


Figure 4. Model evaluation using differentially expressed genes between healthy and disease samples. Using all real data, we first calculate differentially expressed (DE) genes between disease and healthy, at a p-value < 0.00001. Downsampling the healthy samples, we lose significance in most of these genes, whereas replenishing missing healthy samples with generated healthy samples recapitulated most DE genes.

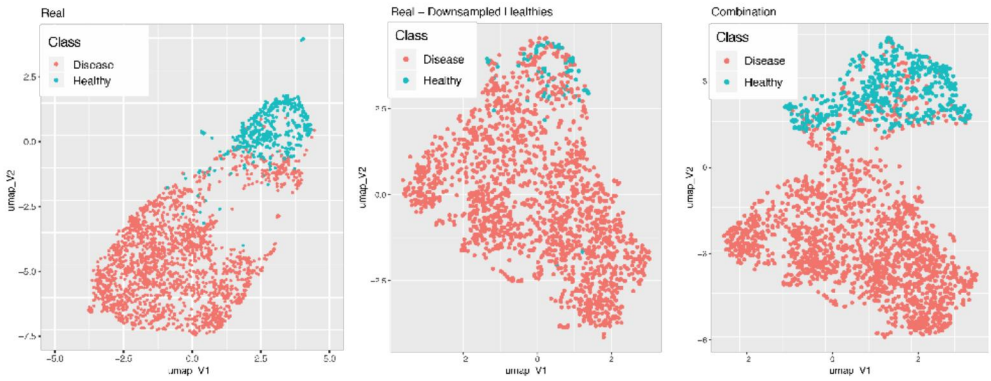


**Figure 5. Model evaluation using downsampling analysis. Using clustering algorithms like UMAP, we show that downsampling of healthy samples results in the disappearance of a distinct cluster between healthy and disease groups. The replenishment of healthy samples using GAN-generated data trained using downsampled data restores clusters.**

## 6 Conclusion/Future Work

Based on our findings, extrapolating beyond single-cell data and using whole-sample gene expression data to generate new data is feasible and could potentially be useful in the short-term for researchers across biological fields. Overall, our model evaluations suggest that our cGAN was able to: 1) retain gene-gene interactions as it created new data; 2) learn to generate better data that becomes less distinguishable from real data; and 3) learn gene architecture of a particular class even after downsampling. While our findings are encouraging, there are still some questions left unanswered. Can this model be applicable across different diseases, especially outside of cancer, in which samples are notoriously difficult to obtain? For rare diseases, how little data do we need in order to successfully train the cGAN? Would it be possible to learn with less data if we had several more disease clusters as anchor points? Ultimately, we would want to validate our findings in a biological setting—we may want to perform wet lab experiments on potential genes of interest that become significant through the help of cGAN.

## 7 Contributions

LB and MS contributed equally to this project. In terms of breakdown, MS had more experience working in GPU space and Python, while LB had more experience in R and grant writing, so MS did more coding while LB gathered the generated data to be visualized and presented via this report and the poster.

Code is uploaded here: `https://github.com/scottmk777/tissueGAN/`

## References

Arjovsky, M., Chintala, S. and Bottou, L. (2017). *Wasserstein GAN.* [online] arXiv.org. Available at: https://arxiv.org/abs/1701.07875.

Edgar, R., Domrachev, M. and Lash, A. (2002). Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Research*, 30(1), pp.207-210.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014). *Generative Adversarial Networks.* [online] arXiv.org. Available at: https://arxiv.org/abs/1406.2661.

Isola, P., Zhu, J., Zhou, T. and Efros, A. (2016). *Image-to-Image Translation with Conditional Adversarial Networks.* [online] arXiv.org. Available at: https://arxiv.org/abs/1611.07004.

Johnson, W., Li, C. and Rabinovic, A. (2006). Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, 8(1), pp.118-127.

Kingma, D. and Welling, M. (2013). *Auto-Encoding Variational Bayes. [online] arXiv.org.* Available at: https://arxiv.org/abs/1312.6114.

Marouf, M., Machart, P., Magruder, D., Bansal, V., Kilian, C., Krebs, C. and Bonn, S. (2018). *Realistic in silico generation and augmentation of single cell RNA-seq data using Generative Adversarial Neural Networks.*

McInnes, L., Healy, J. and Melville, J. (2018). *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction.* [online] arXiv.org. Available at: https://arxiv.org/abs/1802.03426.

Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W. and Webb, R. (2017). *Learning from Simulated and Unsupervised Images through Adversarial Training.*