Measuring and Incorporating Correlations in Generative Adversarial Networks

Stanford CS229 Project
Mentor: Zigfried Hampel-Arias

Vishesh Gupta
Department of Computer Science
Stanford University
vgupta22@stanford.edu

Abstract

As generative models continue to become a more prevalent method in data creation, it is important for us to make sure these models generate data that is most similar to the input data. This issue becomes especially relevant datasets with a very limited number of training examples, where the output from the GAN may contribute a decent amount to training proper models to classify the input data. Given that, we test how well these generative models are able to approach the task of recreating the distributions presented in a dataset of Gaussian noise, specifically checking for the introduction of different correlations in the data. We found that these models, since they perform linear transformations on data, develop correlations that reflect the features which impact the output the most, which typically result in straight lines across the correlation matrix rather than the Gaussian noise expected from the input dataset.

1 Motivation

Accurate machine learning models are no better than the data they are trained on, and as machine learning models have become more advanced and complex, they have also become more data ravenous. This poses an issue in scenarios where researchers are unable to collect a suitable amount of data. For example, in the healthcare industry, data privacy concerns limits the size of many potential datasets. Thus one area of interest in the machine learning community concerns data augmentation and data generation.

Generative Adversarial Networks, or GANs, are a machine learning framework that are used to generate new data that has the same statistics as the training data set. GANs involve two competing neural networks, a generator and a discriminator. The generator learns to map from a latent space to the data's distribution. The discriminator learns to distinguish generated data from the parent data. In this way, the critic network forces the generator to produce increasing convincing data samples.

The GAN model was created in 2014 and is one of the most exciting recent innovation in the field of machine learning. GANs have been able to complete impressive image related tasks, such as recreating human faces (creation of Deep Fakes). However, most of the work done with GANs focuses on application. Not much investigation has been done into the limitations of how GANs simulate data.

To determine how well a GAN performs, we compare the data generated from a GAN to the parent data by measuring two values:

1) the generated data's fidelity in representing the parent data and

2) the intersample correlations introduced by the model to the data.

In this project we will investigate the intersample correlations introduced by the model. Our goal is to identify the correlations introduced into the generated data.

2 Methods

The first step in the project is to identify the correlations introduced into the generated data. We can represent both the parent and generated data as multivariate distributions. We will calculate the correlation in each

dataset via an analysis of correlation matrices. We will then compare the results to identify the introduced correlations.

2.1 Correlation Matrices

A correlation matrix is a matrix of the pairwise correlations between each pair of random variables in the multivariate distribution. The correlation matrix is similar to the covariance matrix in that they both measure the relationship and dependency between each pair of variables. The difference between the arises from the fact that the covariance matrix only indicates the direction of the linear relationship between each pair of variables whereas the correlation matrix indicates both the strength and direction of the linear relationship. The correlation is a function of the covariance and represents the standardized covariance scores. It is obtained by dividing the covariance by the product of the standard deviations of each variaible.

2.2 Dataset

We chose to use 2 datasets in this project: 1) A multivariate Gaussian distribution based on a varying number of features and batch sizes from which we sampled 50000 data points as our training set, and 2) a dataset used to predict wine quality based on 11 chemical features which had 5000 training examples. The purposes of these 2 datasets is to test how correlations are changed from data that is normally and predictably distributed, and one that is not explicitly normally distributed. This allows us to see firstly, if the GAN is able to replicate the distribution of data it is given, and secondly, the differences in correlations of the data itself.

3 Model

3.1 GAN Model

The model we used was provided by our mentor Zigfried Hampel-Arias, and included a four-layer dense network that was trained to be the discriminator with a Leaky ReLU activation function between layers, and a three-layer dense network that was trained as the discriminator. The generator's architecture had 3 Linear layers followed by a batch normalization layer and the same Leaky ReLU activation layer. We also conducted an experiment where we doubled the number of layers in the generator to see if it would have any effect on the correlations we trained on. Both nets ran an Adam Optimzed version of SGD with batch size of either 128 or 256 examples per batch. These models were then trained on the input data for 20 epochs. The following is an example of the loss that was observed from training the GAN on our input Gaussian dataset:

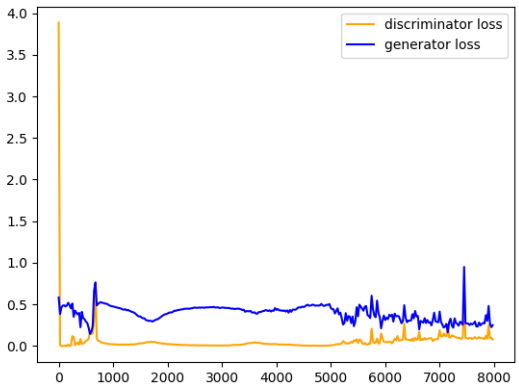

Figure 1: Example of model convergence after about 8000 iterations of batches with 128 datapoints per batch. The y-axis depicts the numerical loss calculated using least-squares variant of the GAN loss function.

3.2 Evaluation

The Evaluation metric from this model is a comparison of the correlation matrices of the data sampled form the trained generator, and the original data distribution. This will allow us to understand and verify the extent to which GANs, and neural networks in general introduce unintended correlations to the dataset. We will also be checking the shape of the distribution based on the L2 norm of each datapoint to see how well our model is

simply able to replicate data. This process will be based on a sample of 10000 datapoints from our model, where we feed uniform noise into the trained generator, and use the output data given by the generator.

4 Results

4.1 Gaussian Generated Distribution

The following contains plots that depict the distribution of the input data and the distribution of the generated data. This analysis is based on the L2 distance of each training point:

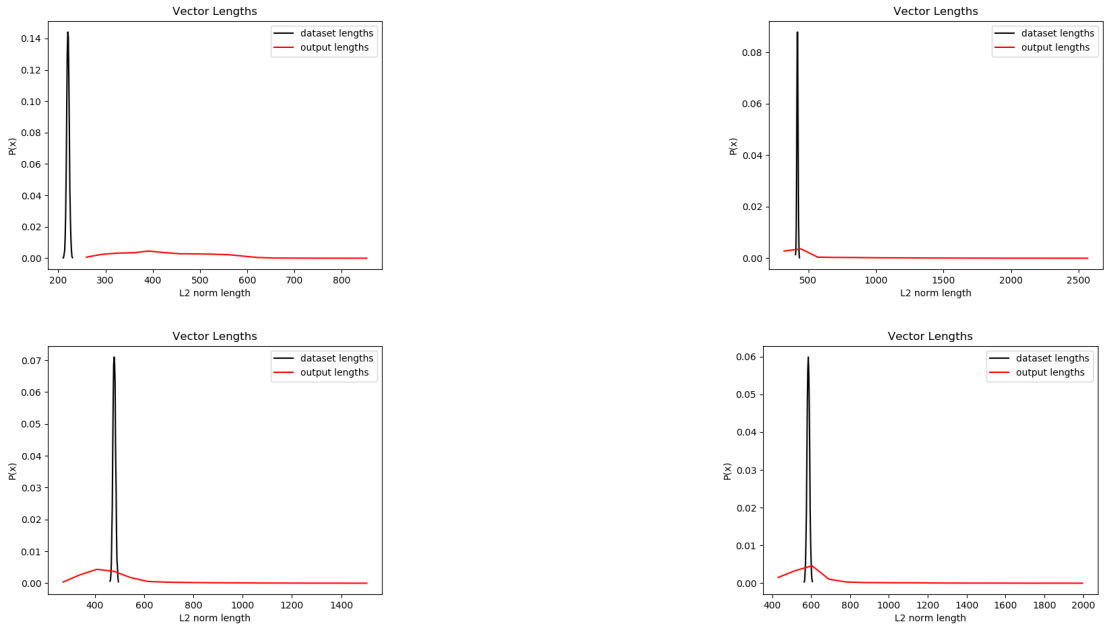

Figure 2: Starting from the top left, we see a massive error in the generated distribution, where the model we are running is a 20 feature model with a 128 batch size. The top right is a 50 feature model with 128 batch size. The bottom left is a 75 feature model with 128 batch size, and the bottom right model has 100 features, and has a 128 batch size with 6 layers in the generator instead of 3.

As we can see here, it appears as though the more features we have, the more accurately we are able to match the mean of the sampled distribution, but we clearly do not perform very well on smaller feature datasets. While this may be cause for concern, Hampel-Arias appears to have gotten similar results with a larger variance but a similar mean length on the distribution. We will be providing an explanation for why this may be in the next section.

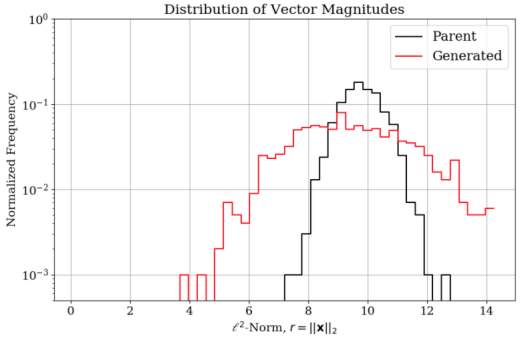

Figure 3: Example of L2 norm plot provided by Hampel-Arias, where we can clearly see a wider generated distribution

4.2 Gaussian Generated Correlation Matrix

The following plots depict the difference in the correlation matrices between the sampled dataset and the generated data. This analysis was done by calculating the covariance matrix of both datasets, and multiplying it by the inverse of its diagonal matrix on the left and right.

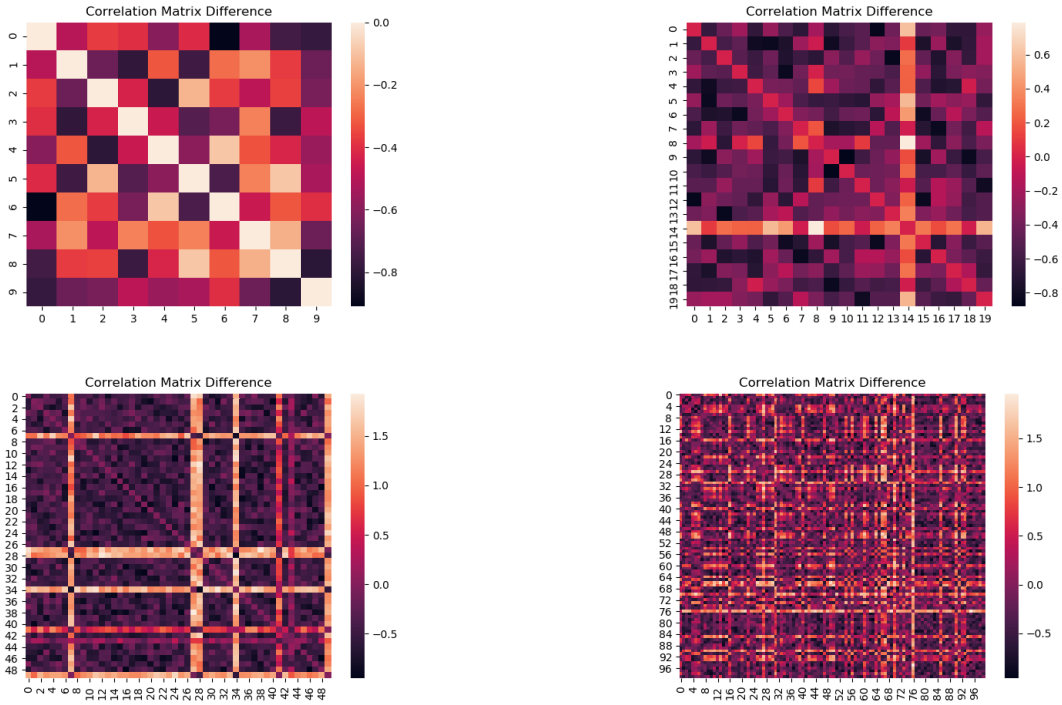

Figure 4: Here we have depicted the differences between correlation matrices from the original and generated datasets. All models had a 128 batch size, with the top left trained on 10 features, top right trained on 20 features, bottom left trained on 50 features, and the bottom right trained on 100 features.

We see an extremely interesting pattern emerge from these matrices, where we can clearly see the randomness of the correlation matrix of the original dataset, and the lines that we see are the correlations learned by the generator. This output looks even more pronounced when we look at the plots individually:

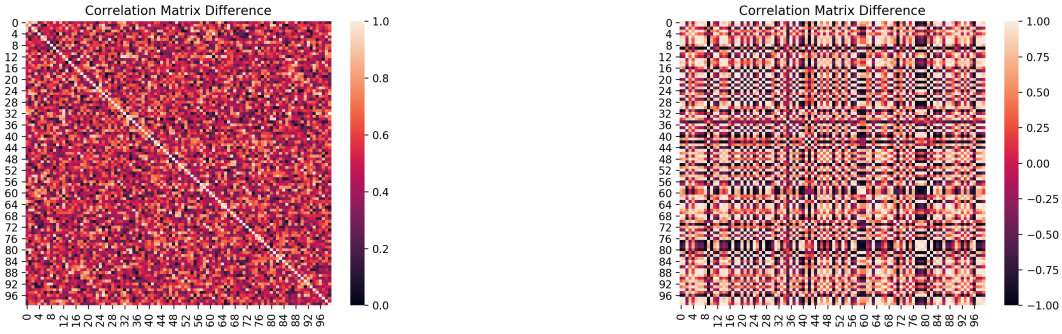

Figure 5: On the left is the correlation matrix of the original dataset for a 100 feature dataset, where we can clearly see the uniform randomness we gave it. The right is the correlation matrix for the generated dataset, where we don't see that same uniform randomness at all, but rather a massive series of lines which tell the model what it should consider important.

4.3 Non-Gaussian Dataset Correlation Matrix

Here, we look at a notable output from the generated dataset based on the wine quality dataset described above:

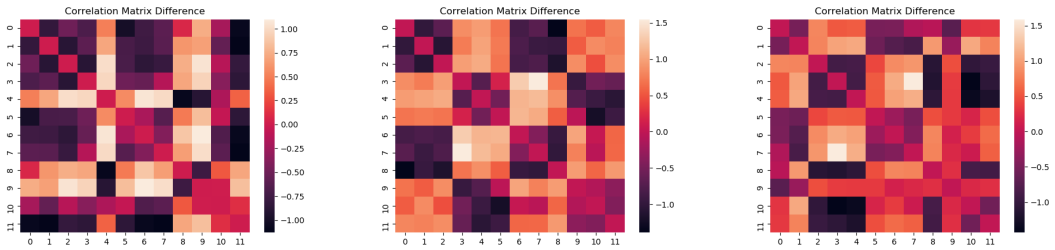

Figure 6: Here, we see the correlation difference for 3 different models run on the wine dataset. The left model is a 128 batch size, the middle model is a 256 batch size, and the right is a 128 batch size with double the layers in the generator.

5 Discussion & Conclusion

Going back to our original question, it is clear as day that the generated data provided by GANs, whether from the Gaussian Noise or a non-normally distributed dataset, results in the introduction of new correlations into the dataset. And this intuitively seems to make sense, where the only way for a neural net to learn is to draw connections where it thinks there are connections, so when trained on Gaussian Noise, it makes sense that it would make connections between certain features having a higher correlation with the output, and some features having less impact on the output. This is ultimately what leads to the lines we see in the generated dataset correlation matrices. This same introduction of correlations likely results in the differences in the distribution we saw above as well, where the shape of our generated distribution is similar to the mean for larger feature spaces, but significantly less for lower feature spaces. Since there are more correlations we can draw for the larger feature space, we likely would be able to see a better output distribution on Gaussian noise.

In terms of ways we could try to mitigate this problem, the output we see from the wine dataset may give some answers, as the total aggregate difference in the correlations we much less in the right 2 runs compared to the standard 128 batch size run. Both of those experiments appear to have resulted in some form of smoothening to the correlation matrix. It is possible that a combination of increased batch size and larger nets may result in data that is more closely correlated to the input dataset.

6 Next Steps

There are many other experiments that can be done to tackle this problem. Our network was based on a model provided by our mentor, so we did not alter it very much, but it would be interesting to see how the correlations change when a dropout layer is added after each linear layer. It would also be nice to experiment with different pre-processing methods such as Factor analysis (not PCA since that essentially separates data based on correlation/covariance) and see how normalization of data can change our output. Additionally, it would be interesting to see how different neural architectures alter this, where something like a DCGAN might have better results given how large it is, allowing it to learn more correlations and potentially end up closer to the source values.

References

Z. Hampel-Arias, "GANomede – Capabilities and Limitations of Generative Models"
C. Esteban et al., "Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs"
M. Arjovsky et el., "Wasserstein GAN"