CPSC 66 Final Report: An Analysis of Synthetic Dataset Creation Using Deep Generative Models

Will Lee WLEE1@SWARTHMORE.EDU
Zach Rothenberg ZROTHEN1@SWARTHMORE.EDU
Jimmy Shah JSHAH1@SWARTHMORE.EDU
Swarthmore College, 500 College Avenue, Swarthmore, PA 19081 USA

Abstract

Recent developments in machine learning over the last decade have led us into an age of data. Everywhere, both in industry and research, vast datasets are being collected to train powerful deep models. However, this data collection comes with a troubling moral quandary: how can privacy still be protected in an age of data?

We propose the use of modern deep generative models for the purpose of bootstrapping new synthetic datasets from existing data. Specifically, we explore Generative Adversarial Networks and Deep Boltzmann Machines for generating new images that can be used in place of original data for training models. Initial results on the MNIST dataset suggest that this is a effective method and could potentially be a powerful tool in privacy sensitive domains such as medicine.

1. Introduction

The meteoric rise of deep learning to the mainstream has spelled the start of an age of data. Companies like Google and Facebook are able to harvest billions of samples from their users, enabling algorithmic analysis on a scale never before seen. However, this revolution does not come without some troubling consequences. The question of privacy has become a major roadblock in the way of further progress in the field, especially in domains like medicine where it remains of critical importance. While researchers can sometimes gain access to impressive datasets, they often cannot spread them to colleagues, resulting in a distinct lack of standardized bench-marking datasets like would be found in areas like general image classification (MNIST and CIFAR are two popular examples). Without anonymous datasets, researchers are unable to share their results with one another, stunting progress.

This issue constitutes a major impediment for recent research, and has therefore become itself the target of recent academic efforts. Most recently, (**?**) released a tool named DataSynthesizer designed to bootstrap new and private datasets from existing sensitive data. The main concerns expressed in this paper relate to two key features of good synthetic data: that they are both "obviously synthetic" (different from the original data) and "offer strong privacy guarantees". Without the first, the original owners of the data will be wary to allow the generated version to be published. This is the key problem in the sharing of synthesized data and, if not solved, completely nullifies any work done in generation. On the other hand, the author of the synthetic dataset owes privacy to the people whose data they are using to bootstrap. A poorly considered algorithm might still leave traces of the original information in the new dataset, allowing a dedicated attacker leto capture sensitive information.

Howe et al. implements several probabilistic models in their paper, including a bayesian network model used to capture the dependencies inherent in the original data and then resample from the learned distribution. While this works well for the tabular data they considered, the greedy construction of the graph on high dimensional feature vectors, such as in image processing, would prove ineffective. Instead, we propose the use of modern generative models, known to scale well to image tasks, for bootstrapping synthetic datasets of images.

We consider two such models: a Generative Adversarial Network (GAN) as proposed in (**?**) and a Deep Boltzmann machine (DBM) from work by (**?**). We chose these models as representing the state of the art for deep generation, both being developed in the last decade as a part of the rise of modern computer vision. Both leverage parallelized training on graphics cards, allowing for rapid learning on high dimensional images (**?**).

CPSC 66 Machine Learning Final Report Proceedings, Swarthmore College, Fall 2017.

Specifically, a GAN utilizes the competitive training of two models against one another. One model attempts to generate from the underlying distribution and the other tries to discriminate between real and generated images. As the generator improves, so does the discriminator, forcing increased accuracy to the generated images. In this way, the pair eventually converges to a position where the generator can be used to draw accurate samples and the discriminator can be used to distinguish any samples that do not appear to be from the original data in additoin to regular classification. The discriminator has interesting applications to fraud detection (**?**), but we do not address them in this paper. Instead, we choose to focus on the generator, which is a honed sampling machine. The convolutional nature of the generator network makes it a fitting candidate for computer vision tasks and a promising model for our experiment.

Our other model, DBMs, take an undirected graphical approach to modeling the image distribution, using a Markov Random Field to capture dependencies in neighboring pixels. The graph can then be used for generation by implementing Gibbs sampling, a convergent algorithm for resolving random noise into accurate synthetic data. While older than GANs, DBMs have recently been shown to perform very well in the synthesis of images, as in (**?**), where they were used to sample new faces from an initial collection. Therefore it stands to reason that they are also a competitive model for our experiment.

Our work is a proof of concept that these models can be used to generate new representative data from a "toy" dataset. We hope future work can elaborate on our results, either by attempting similar experiments using different models or by finding tactics to generate on more complex datasets. Our main results are summarized in Table **??**. We conclude this paper with a discussion of possible ramifications of synthetic data.

2. Approach

2.1. GAN

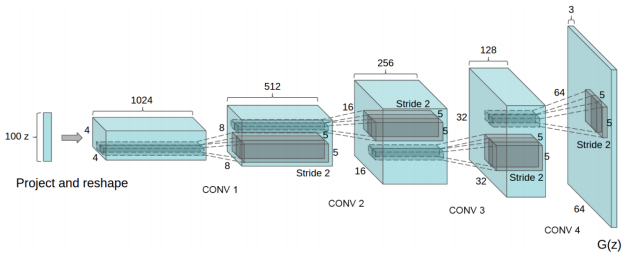

Figure 1. Graphical representation of the DCGAN generator network (**?**)

The specific GAN implementation we used is the DCGAN described in (**?**). This model takes advantage of several modifications to the initial GAN formulation in order to achieve greater stability, a notorious problem for the original GAN formulation. The DCGAN consists of two adversarial Convolutional Neural Networks (CNN) that compete against one another. This is different from previous formulations, which were only able to use multi-layer perceptrons (MLP) for the competing networks. These convolutional models allow the network to scale well to computer vision tasks, on which unmodified MLP are poorly equipped. However, these convolutional layers come with a tradeoff: the already tenuous stability of GANs is made even more apparent. To combat this, Radford et al. employs several techniques to improve the algorithm's chance of useful convergence. We will focus on two of them as important to our understanding of the model's performance.

The first of these constraints is the disallowing of any pooling layers in the network. Instead, the architecture uses high stride length convolutions to reduce the dimensionality of its input. This allows the networks to learn what type of upsampling or downsampling is most effective, giving the models increased flexibility. This adaptability is what allows DCGANs to adapt in tandem, reaching a more useful equilibrium.

The second is that DCGANs, like most modern networks, use batch normalization. This is a topic advanced enough to deserve a paper of its own (which it has in (**?**)), but we will cover only the aspects that pertain to DCGAN stability. Batch normalization is an extension of the minibatch training common to CNNs. In this sort of training, the network is trained on a subset of the training set each iteration, allowing learning from a series of examples at once. This is especially important in DCGANs, where the generator is made to generate a batch of examples at once. This way, the discriminator is able to pick up on the entropy of the generated data, stopping the generator from converging on similar accurate samples. Without it, a generator could learn to sample one label accurately, ignoring all others. The normalization factor of batch normalization pertains to the statistical standardization of the data between every layer in the network. After each activation, the resulting tensor is shifted in accordance to both its mean and variance. This helps the network with the vanishing gradients problem and poor local minima for the cost function. Combined with high stride length convolutions, batch normalization allows the normally incredibly unstable GAN algorithm to be extended to CNNs.

2.2. DBM

We used the DBM implementation from (**?**), a GitHub repository with several implementations of Boltzmann Ma-

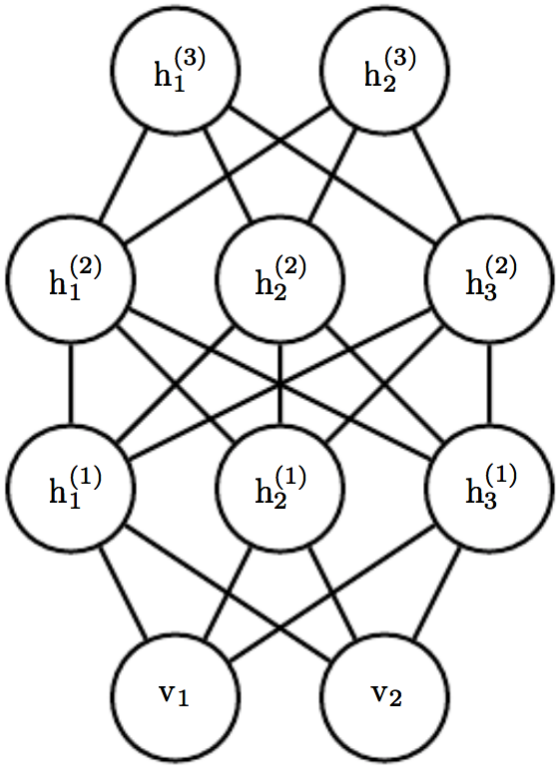

Figure 2. Graphical representation of a small DBM. The model in our experiment is significantly larger. (**?**)

chines. This repository also included pre-trained models that we hoped would help expedite the sampling process. The DBM itself is a deep architecture, consisting of several other models called Restricted Boltzmann Machines feeding into one another. A Restricted Boltzmann machine is a more constrained version of the Generalized Boltzmann machine, which is a Markov Random Field model all Boltzmann machines are based upon. A Generalized Boltzmann Machine consists of a simple undirected graph with weighted edges. The nodes of this graph are of two categories, visible and hidden. In a Restricted Boltzmann Machine, nodes of the same category cannot be connected with one another. This allows for efficient generation via Block Gibbs Sampling, described in Algorithm **??**.

The Restricted Boltzmann Machines are independently trained in a greedy fashion before composition together into a DBM. The resultant model can learn deep dependencies, allowing for sophisticated generational modeling.

Algorithm 1 Block Gibbs Sampling on a Restricted Boltzmann Machine

Input: weights matrix W, hidden bias a, visible bias b
H = random sampling from gaussian for each hidden node
repeat
  $V = \text{sigmoid}(WH + b)$
  $H = \text{sigmoid}(W'V + a)$
until convergence

3. Experiment and Results

3.1. Synthetic Dataset Sampling

First, we will discuss sampling from the GAN. The model was first trained for 7 epochs on the MNIST dataset using batches of 64 images at once. Once 7 epochs passed and convergence was deemed to have been reached, the model was stopped. At this point, the generator network was fed samples of Gaussian noise, along with a set of labels for the minibatch. Minibatches were generated of homogeneous labels, and the first item of each minibatch was sampled to get our labelled synthetic image. We sampled 100 images of each label, acquiring 1000 synthetic images in total.

Figure 3. DBM generated images on CIFAR

Unfortunately we were not able to sample a complete dataset from the DBM. The specific implementation we were using did not allow for easy modification of the computational graph created by TensorFlow to allow sampling. Therefore our model was capped to generate the same 100

Table 1. Real vs Generated MNIST Performance

MODEL	REAL	GENERATED
SVM	0.653 ± 0.048	0.742 ± 0.021
CNN	0.610 ± 0.048	0.604 ± 0.038

Table 2. Real vs Generated MNIST Performance (1000 training examples)

MODEL	REAL	GENERATED
SVM	0.903 ± 0.008	0.899

images every time. Since we were unable to generate the varied dataset we required and were under time constraints, we decided not to retrain the DBM to get more images. We were also dissuaded by the images that were generated. Figure **??** displays our samples from CIFAR-10 after 10,000 Gibbs steps. The only subjectively distinguishable labels in these images are the horses; everything else is blurry noise with amorphous shape. The state of the art generation using GANs for CIFAR-10 is much higher, generating much sharper images. Generation by the DBM on MNIST resulted in much better images than CIFAR-10, but the hard cap on 100 images made it impossible to use as a dataset. The rest of our discussion will therefore rest on the results yielded from the GAN's adversarial approach.

3.2. Synthetic Dataset Evaluation

To evaluate our synthetic dataset we tested its performance training two different models, a CNN and a Support Vector Machine (SVM) for classification on the original MNIST test set. First, 10 training sets of 100 images were sampled from both the original and synthetic datasets. These datasets had no overlap with one another, creating 10 completely independent sets. Then, these datasets were used to train both a CNN and an SVM, after which the models were evaluated with the same MNIST test set of 10,000 images.

The CNN we used for classification came from (**?**) and followed a fairly standard architecture of two convolutional layers, a max-pooling layer, multiple dropout and flattening layers, and a final dense layer with softmax activation.

Preprocessing included randomly reshifting the images both vertically and horizontally by about ten percent. Images were also flipped horizontally at random. This was applied to both the real and generated images.

The SVM we used for this experiment was of the soft margin variety with an RBF kernel, using a C value of 5 and gamma value of 0.05.

The results of this experimentation are described in Table **??**. For the CNN, our data fit exactly what we expected. The intervals for our real dataset models and our generated dataset models overlap heavily, suggesting little to no statistical difference between the two. However, for the SVM we observed significant deviation from the hypothesis. The model performed better trained on the generated dataset than on the real one, which was extremely unexpected. The synthetic dataset was expected to yield at best equal performance, not better.

We believe this unexpected result is due to a regularization effect happening due to the generated data. Subjectively, the synthetic dataset has a "fuzzier" feel to it, with the DCGAN capturing the essence of the digits but failing to achieve the same crispness of the handwritten scans. At very small training set sizes, such as the 100 tested here, an SVM could be overfitting the data. The "fuzzy" samples of the generated dataset could prevent the model from learning an overspecified hypothesis. We decided to test this regularization hypothesis by performing a similar experiment, but on all 1000 digits. For the real dataset counterpart we generated 10 different datasets of 100 labels each, obeying the same exclusion with which we selected the earlier data. The results of this second inquiry are summarized in Table **??** and seem to confirm our hypothesis. With a larger training set, the chances of overfitting are much lower and the regularizing effect of the generated dataset has little to no effect.

3.3. CNN Mixed Dataset Learning Curve

In addition to the above analysis, we compared the effect of augmenting a training set with real MNIST images versus augmenting a training set with generated MNIST images (from our GAN). We used a baseline training set size of 10 real MNIST images. In one experiment, this baseline training set was augmented with 10 additional real MNIST images. In the other experiment, the baseline training set was augmented with 10 generated MNIST images. We continued to augment the training set sizes with real and generated MNIST images respectively until the training set reached a total size of 110 images. In both experiments, each training set size was evaluated three times; the CNN was trained three times on three independent training folds, and each fold was evaluated on the MNIST test set of 10000 real MNIST images.

As seen in Figures **??** and **??**, increasing the training set size using images generated from a GAN results in similar accuracy increases as increasing the training set size using real MNIST images. Augmenting with real MNIST data and generated MNIST data ultimately result in very similar peak accuracies of 64.77% and 61.52% respectively. How-

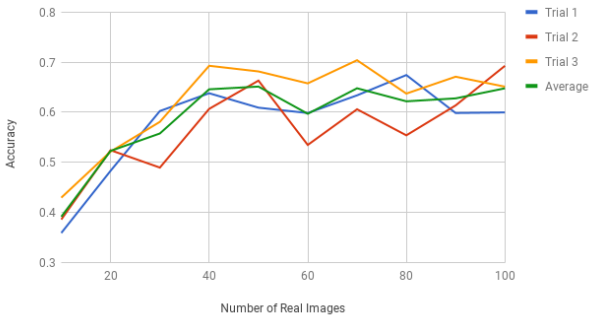

Figure 4. CNN Performance on Varied Sample of Real Images

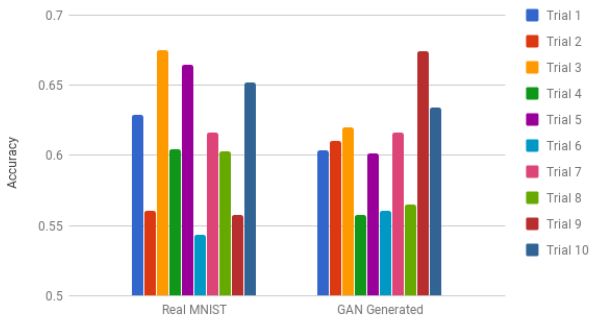

Figure 6. Performance Comparison over 10 Trials

ever, it appears that augmenting with real MNIST images results in convergence to this peak optimum faster than augmenting with generated GAN images. Notably, the increase in accuracy on the interval [0,40] when augmenting with real MNIST images (Figure **??**) is stronger than the same interval when augmenting the training set with generated images (Figure **??**).

At first glance, this does not fully support our SVM data, where we observed that small training sets composed of generated data performed better than those composed of real MNIST images. Initially, we hypothesized that this was caused by a regularization effect from the "fuzziness" of generated images. We hypothesize that our regularization hypothesis is still correct, and the reason we did not see the regularization effect in this experiment is due to noise/variance from using small training set sizes. This is supported by the fact that augmenting the training set with more than 50 images results in no statistically significant difference between augmenting with real MNIST images and augmenting with generated MNIST images. We thus conclude using generated MNIST images to augment a training set can be an effective replacement for augmenting using real MNIST images.

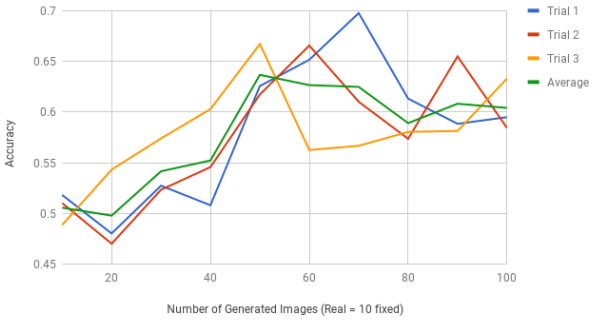

Figure 5. CNN Performance on Varied Generated Data

4. Discussion

Our results show a very promising possibility for the use of synthetic datasets in machine learning. While synthetic approaches have been attempted before, as in (**?**) by using sampling distributions, the possibility of generation through adversarial approaches represents a rich new area of study. Even as we were disappointed with our inability to test DBMs on our dataset, we remain convinced that they are also a potentially effective model for dataset synthesis and would recommend further investigation as an open question for research.

In terms of dataset choice, we expected MNIST to be relatively easy to generate for and were proven right. The dataset is well known to simplify away several of the complicated difficulties of computer vision work, yielding clean results. We wished that our GAN could have generated on CIFAR-10, but in their current stage not enough is known about GAN stability to make them a practical consideration for more complex datasets without significant tuning. While it is possible to get DCGANs working on CIFAR-10, it is a delicate process that requires great amounts of algorithm specific expertise. However, if one is able to get the model working on their data, an adversarial approach yields state of the art generation. While at its current best DCGAN images on CIFAR-10 can be distinguished as fake with $> 95\%$ accuracy by humans that are familiar with the dataset (**?**), our earlier results into overfitting suggest that this ambiguity might actually help augment the dataset and make it more resistant to overfitting.

The societal implications of synthetic dataset generation are discussed at length in (**?**), but we reiterate them here. Too often the datasets that have the real potential to benefit society are held behind strict privacy restrictions, such as FERPA and HIPAA compliance acts. Often times, even if access is granted, rules prohibit the data from being changed in any way, including combination of data. With the ability to synthesize statistically identical datasets,

we gain the ability to introduce this data into the academic community. As given by Howe et al., if the synthetic data is both representative of the true data and provides strong guarantees about privacy, as such in generative model cases, then collaboration can flourish and we can more easily converge on these potentially beneficial insights. This also incentivizes data owners to share, who often are risk-averse to avoid potential findings of malpractice. A benchmarking competition like ImageNet, responsible for massive advancements in general Computer Vision, could be brought into more useful domains. Not only would a benchmarking dataset help the above issues, but they can also bring crucial media presence. The advances in CNN architecture made on ImageNet make headlines and brought popular consciousness to the field. This can bring more funding to issues that have great potential societal impact.

Specifically in image-centered domains, this advance could bring traditionally exclusive data into the mainstream academic world. A collection of brain scans would normally be an incredibly-protected dataset, limited in access to one team by their agreement with a hospital. If an generative approach could create new scans, representative of the originals, the synthetic data could be shared. Different teams could share different insights on the data, eventually leading to a greater understanding of how to analyze brain scan data.

5. Conclusion

In this paper we presented an argument for the use of synthetic data generation in domains populated by sensitive image data. We showed that modern generative methods demonstrated more than adequate performance on the MNIST dataset and could possibly be adapted to new datasets as GAN research continues and our understanding of their stability improves. We also showed that the sometimes inexact nature of synthetic replication can lead to a regularized dataset that implicitly defends against overfitting.

We believe this generative approach will be instrumental in the future of open data science and machine learning research. It will open up new domains to the community and result in the rapid growth of our understanding in these previously unreachable areas. The generation of synthetic datasets show particular promise for areas such as medicine, where patient data privacy is of the utmost importance; mitigating the cost of spreading sensitive datasets like those found in medicine will lead to improvements not only in healthcare, but in all areas of machine learning and computer science where it is currently difficult to compare, share, and evaluate results run on private datasets. Data with restrictive usage licenses can be made into an open source tool for learning.

Our approach of evaluating modern generative techniques on the MNIST data serve as a baseline proof of concept that synthetic datasets can be used to supplement or even replace real datasets. However, it is important to note that the question of generating more complex datasets such as CIFAR-10 images still remain and are not fully addressed in this paper. In our experimentation, we failed to generate high quality CIFAR-10 images using models from open source, community-driven repositories. However, our understanding of generative models is expanding rapidly; for example, Google released an open source library for building GANs, named TFGAN, on December 12th, 2017, two days before the publishing of this paper. Lightweight tooling like TFGAN will allow the community to further experiment with GANs, hopefully leading to the rise of good implementations on more challenging datasets. This would allow our approach to be ported to more practical problems. We hope that others can build on our research using these tools and avoid the repository pitfalls that prevented a more thorough examination.

6. Acknowledgments

We'd like to thank Professor Ameet Soni for guidance on this project and for a great semester in Machine Learning. We'd also like to thank Jeff Knerr for his help running long jobs on lab computers. Lastly, we'd like to thank `anise` and `savory`, the two lab computers that have been running for the past straight week.

7. Full Data

Table 3. CNN MNIST Accuracy on True and Observed Data

TRIAL	REAL MNIST	GENERATED
1	0.6290	0.6039
2	0.5608	0.6106
3	0.6754	0.6200
4	0.6042	0.5578
5	0.6647	0.6015
6	0.5433	0.5605
7	0.6162	0.6160
8	0.6027	0.5647
9	0.5579	0.6745
10	0.6518	0.6338
AVERAGE	0.610	0.604
STANDARD DEVIATION	0.048	0.038

Table 4. SVM MNIST Accuracy on True and Observed Data

TRIAL	REAL MNIST	GENERATED
1	0.6177	0.7458
2	0.7073	0.6967
3	0.6949	0.7446
4	0.6015	0.7762
5	0.7426	0.7478
6	0.5983	0.7405
7	0.6478	0.7585
8	0.6122	0.7378
9	0.6448	0.7558
10	0.6587	0.719
AVERAGE	0.653	0.742
STANDARD DEVIATION	0.048	0.021

Table 5. CNN Performance on Increasing Real Dataset Size (MNIST)

REAL IMAGES	TRIAL 1	TRIAL 2	TRIAL 3	AVG
10	0.359	0.3859	0.4295	0.3914
20	0.4831	0.5242	0.5214	0.5228
30	0.6024	0.4895	0.5808	0.5575
40	0.6379	0.6067	0.6927	0.6457
50	0.6091	0.6631	0.6814	0.6512
60	0.5984	0.5347	0.6576	0.5969
70	0.6337	0.6059	0.7038	0.6478
80	0.6742	0.5539	0.6372	0.6217
90	0.5984	0.6136	0.6709	0.6276
100	0.5996	0.6926	0.651	0.6477

Table 6. CNN Performance on Increasing Generated Dataset Size (MNIST)

GENERATED IMAGES [1]	TRIAL 1	TRIAL 2	TRIAL 3	AVG
0	0.518	0.510	0.488	0.505
10	0.480	0.470	0.543	0.498
20	0.527	0.523	0.574	0.541
30	0.508	0.545	0.603	0.552
40	0.625	0.617	0.667	0.636
50	0.651	0.665	0.562	0.626
60	0.697	0.610	0.566	0.624
70	0.613	0.573	0.580	0.589
80	0.588	0.654	0.581	0.608
90	0.595	0.584	0.633	0.604
100	0.690	0.608	0.546	0.615

[1] In addition to a starting set of 10 real images, one of each label