World Wide Technology

Obscuring and Analyzing Sensitive Information with Generative Adversarial Networks

WWT Artificial Intelligence Research & Development

MARCH | 2019

World Wide Technology
wwt.com

Table of Contents

Abstract

Much of the data collected by corporations and public institutions is too sensitive to share publicly or with a third party. Strict rules govern who may access medical records, financial information and other confidential data. However, there is a great potential for this data to be analyzed in new ways, if only it could be shared with the right researchers or business analysts. Generative Adversarial Networks (GANs) are an advance in artificial intelligence which may provide a solution to this problem. A well-trained GAN will create new data that is representative of the original data. This output could be analyzed by a third party while obscuring any sensitive or confidential information from the original data. In this paper, we assess the potential of using GANs to generate representative data and build insightful models without the original data.

Business Justification

In this paper, we explore a method to obfuscate sensitive data in a novel way using artificial intelligence. Rather than obfuscating records by deleting personal identifiers or adding noise, we seek to generate new records entirely using a GAN. The new sample of data will fall within the same distribution of the original data but will not correspond directly to any unique record from the original data set. Ideally, this will be done such a way that the structure of correspondence between variables reflects the original data. We further aim to prove that useful models can be built using the generated data of a GAN.

Many sources of sensitive data are not allowed to be shared with third-party researchers. Healthcare organizations have strict limitations in how they may share patient data. Financial records are kept secret by banks and the IRS. Other personal information is guarded by governments and businesses at the trust of citizens and customers.

These sources of data could be used in a variety of ways if they were accessible to a third party. Medical images could be more easily analyzed by outside researchers. Financial records or credit card statements could be used by security analysts developing techniques to detect fraud or other illegal activities. Network logs that were GAN-generated could mock the activity of viruses or malware (E. David) and be used to test the performance of security software without the danger of installing the virus on a computer.

Traditional methods of obfuscation have not always proven reliable, and erasing names and ID numbers is not always enough. In some cases, only a small amount of data is needed for personal identification (Sweeney). Geolocations alone can be used to identify an individual at times. With the recent developments in generative adversarial neural networks, new data can be generated which is not tied to any individual record. GANs potentially provide a more secure way to release data to researchers.

Generative adversarial networks were first devised in 2014 and have most commonly been used to generate images of people and objects (I. Goodfellow). They work as two competing neural networks, a generator and discriminator: The generator seeks create a realistic image; the discriminator attempts to discern between real-world images and images created by the generator. As each network trains in parallel, both are meant to improve in their ability. Eventually, the generator creates images that look quite realistic, although the generator never directly sees what a true image looks like. GANs have been used to create images of numbers that have never been drawn and faces of humans who have never lived (T. Karras).

Sensitive information can come in many forms apart from images; it will be especially valuable to extend the scope of GANs beyond images. Most prior research into GANs focuses on images. This not only provides impressive visual results, but also allows humans to quickly evaluate the quality of the results. However, many sources of data in industrial applications are values in a table or time series. With this type of data, it can be more difficult for a human to evaluate whether the data is accurate or artificially generated.

In this paper, we explore the feasibility of generating representative data for two types of data: binary input from medical records and real-valued sensor data from industrial mining trucks. Sample data is generated for each data set using a GAN. We evaluate how well the generated data preserves correlations found in the original dataset. Finally, we test the feasibility of building a predictive model using the generated data. This test is to prove that data generated from a GAN could be supplied to outside researchers without releasing the sensitive original data. Furthermore, the models built using generated data provide a secondary method of determining the quality of the GAN data generation.

Experimental Setup

This study involves two distinct sets of data:

1. **Medical records**

   A plain text file of US hospital discharge records from 2010 was obtained from the National Hospital Discharge Survey. After extracting the data into a usable format, a binary dataset was created from basic demographic information and medical diagnosis codes. The presence or absence of ICD-9 classification codes relating to drugs of abuse and other comorbidities in each discharge record were used to populate the columns indicated in Table 1. The final dataset included 19 binary variables and approximately 150,000 rows corresponding to unique patient discharges.

2. **Mining Haul Truck sensor data**

   This is a proprietary dataset used with permission. The data set consists of approximately two weeks of one-second resolution sensor measurements from a large capacity haul truck in a surface mining operation. The six unique variables selected for this study were: *Incline, Ground Speed, Engine Speed, Engine Load, Boost Pressure and Throttle Position*.

Neural network training was performed on an NVIDIA DGX-1 using one Tesla V100 GPU. Eight GPUs are available in the DGX-1; however, only one was needed for training due to the smaller sizes of the data sets (less than 1 GB). Docker containers were loaded on the DGX-1 to run Python 3 and the packages listed below. Much of the code development was done using Jupyter notebook environments.

We used the following Python packages to perform the training and testing:

- Pandas
- Numpy
- SciKit-Learn
- Matplotlib
- Tensorflow
- Keras

We based our neural network architectures and open-source code from the Medical GAN git repository for the medical data. A novel network architecture (Figure 6) was developed for the truck sensor GAN.

Methodology

AN INTRODUCTION TO GENERATIVE ADVERSARIAL NETWORKS

A generative adversarial network consists of two neural networks: a generator and a discriminator. The generator produces fake data, and the discriminator tries to differentiate between the fake and real data. The two train against each other, connected in the structure in Figure 1.

A key feature of this structure is that the generator never sees the real data. Instead, it learns how to produce similar-looking data through feedback from the discriminator. Thus, in situations involving confidential data, one can train the full network in a secure environment and then release only the generator to outside researchers. Then the generator can be used to produce arbitrary quantities of data for analysis.

Training a GAN is a little bit different than training a usual neural network. The strategy is to alternate between the two tasks: training the discriminator and training the generator.

Training the discriminator:

This works as with any other neural network but with the extra step of producing a current batch of fake data from the generator prior to each training iteration. One feeds the real and generated data through the neural network and train it to output the correct real/fake labels.

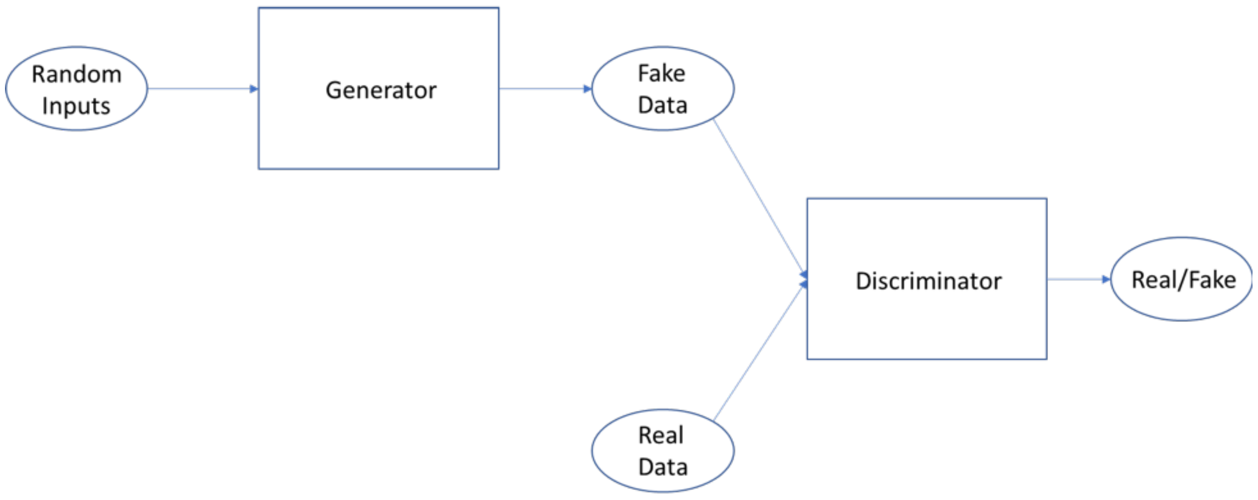


FIGURE 1: Architecture for training a Generative Adversarial Network. The Generator creates a sample of data starting with random inputs. The discriminator takes in both real and fake samples and provides a score from 0 to 1, with higher values indicating samples the Discriminator believes are more likely to be real.

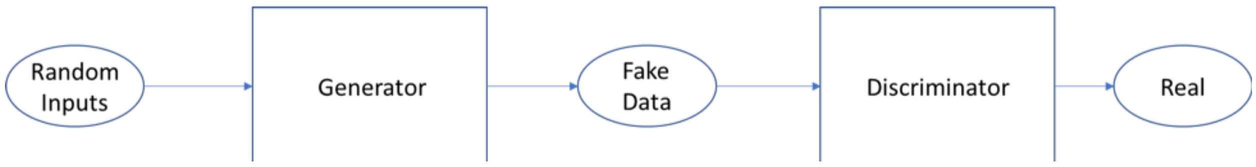


FIGURE 2: Architecture of the used to train the Generator. The Generator takes in random inputs and creates a fake sample of data. The discriminator scores the sample from 0 to 1. The Generator attempts to increase this score. Only the layers in the Generator are trained using backpropagation in this step – the layers in the Discriminator are trained separately.

To train the generator, one uses the combined architecture but trains only the layers belonging to the generator. These layers are updated with backpropagation to achieve labels of "Real" for the generated data, as in Figure 2.

COMMON ISSUES WITH TRAINING GANS

Training networks in this structure requires overcoming some unusual difficulties. First, it is common for the discriminator to overpower the generator during training. Since neural networks learn by incremental improvement, if the discriminator gets too good at detecting the fake data and the generator is left with no direction in which to quickly improve, training breaks down. The methods to resolve this are to disadvantage the discriminator, for example by giving it somewhat noisy answers during training, or even occasionally giving it the wrong answers.

A second problem is that it is hard to tell when to stop training a GAN. When training a normal neural network, the accuracy or loss of the network stabilizes at a certain point and further training is useless. Since the two parts of the GAN are learning against each other, they stay roughly equally accurate over the course of training (if the training is working), and that indication of training completion is lost. Ideally one checks some property of the generated data periodically to see if it is continuing to improve. When generating images this is easy – a human can look at output occasionally – but for other data types, this is more difficult. We take a more quantitative approach to checking the generated output of the GANs for tabular data. We compare the marginal distribution of each variable in the original and generated data, and we compare the bivariate correlations between variables. As the GAN improves, these distributions will be closer to those in the original data.

One more common issue is when the generator begins to create only one or a few distinct outputs. This is called *mode collapse*. The output often looks like something from the real dataset, because the generator has locked onto one output that is sure to fool the discriminator. Ways to mitigate this problem are still being researched. One method is to have the generator submit output in batches. Then the discriminator can decide whether a batch of samples is real rather than a single sample, forcing the generator to vary its generated samples.

OVERVIEW OF USE CASES AND APPROACH

There are two use cases explored in this experiment. Each use case has a different type of data. The medical records have the form of 19 binary variables, with a 19×1 row for each sample. The mining truck data has real-valued data of six variables in a time series forming a 1000×6 table of values for each sample.

For each dataset, we take the following approach.

1. Train a Generative Adversarial Network with the sample dataset.
2. Generate representative samples and compare these with the original dataset.
3. Train a model to predict a feature in the data using the generated samples. Test the model predictions using the original data and compare to a model built with the original data.

In the medical records, we build a model to predict whether there is an indication of *opioid abuse* based on the other 18 factors. For the mining truck data, we predict the *throttle position* based on the other five sensors.

Much of the research with GANs aims to show that data or images can be generated which looks to a human observer like it could be part of the original data set. We aim to go further and examine whether the generated data may be used to build predictive models which perform well even on the original data. This is key for any institution hoping to generate data with a GAN to be released for scientific research.

Results

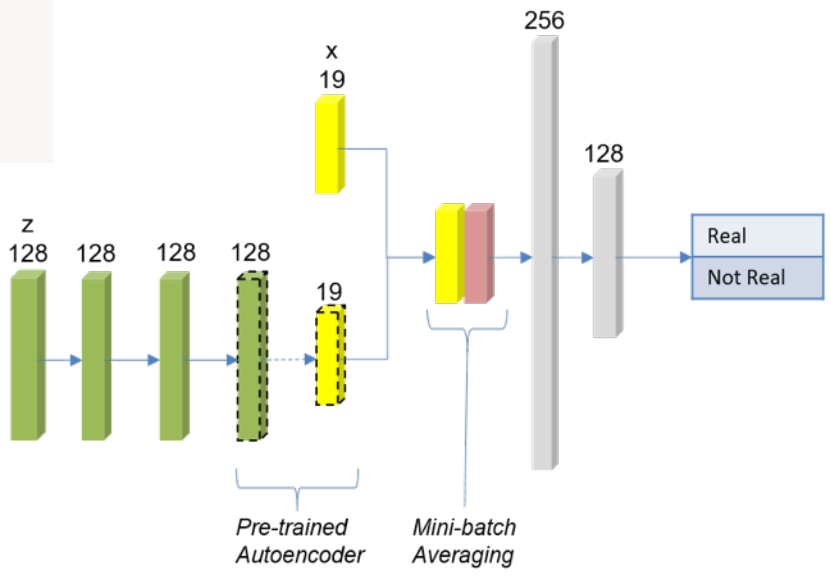
BINARY DATA: HEALTH RECORDS

While much of the research with GANs has involved modeling data in continuous variable space, as in image processing, recent work has demonstrated that the adversarial framework can be adapted to generate realistic samples containing discrete variables (Hjelm and Jacob). These efforts are particularly relevant to the healthcare domain, where salient patient information such as medications, diagnoses and procedures are reported using standardized medical codes.

Choi et al. have shown that binarized electronic health records (EHR) can be synthesized using a modified GAN training routine, which they have termed medGAN (E. Choi). To work around the intractability of backpropagation across binary nodes, the authors use an inline autoencoder to map the continuous output of the generator to a discrete binary vector. The decoded vector is then fed to the discriminator along with the real binary samples. This network architecture is shown in Figure 3. This modification allows the generator to sample a fully continuous representation of discrete variable space during training, which results in more effective gradient-based learning compared to approximation of the binary vector with continuous values.

To illustrate the effectiveness of medGAN in generating quasi-authentic binary data, we trained the model on binarized hospital discharge records and examined the marginal probability, variable correlation, and single-variable prediction performance of the generated output. The published medGAN model architecture and hyperparameters were unmodified for our experiment, with the only difference being the dimensionality of the input/generated data. Performance evaluation described herein was based on a set of 10,000 generated records.

FIGURE 3: The medical GAN architecture feeds the generated data through a pre-trained autoencoder to create the sample. It also incorporates mini-batch averaging, concatenating each sample with the average from a batch of samples.



BINARY VARIABLE	REAL	GENERATED
sex	0.595	0.580
opiate abuse	0.010	0.012
hepatitis	0.076	0.071
alcohol	0.052	0.055
chronic pain	0.027	0.025
cannabis	0.021	0.012
amphetamine	0.002	0.002
cocaine	0.009	0.010
antidepressant	0.002	0.001
other drug	0.007	0.005
post traumatic stress	0.007	0.006
psych condition	0.003	0.004
anxiety	0.039	0.039
tobacco	0.165	0.146
unemployment	0.003	0.002
suicidal	0.022	0.017
depressive	0.057	0.052
over 65	0.321	0.363
under 18	0.146	0.142

TABLE 1: Bernoulli success probabilities for binary variables in real and generated hospital discharge records.

The column-wise Bernoulli success probabilities of the authentic and generated records are given in. The positive prevalence of each feature in the generated sample closely matches that of the real data, with an R^2 of 0.994. According to the authors of medGAN, this faithfulness is attributed to the use of minibatch averaging during training, where the means of all samples in the real and generated batches are concatenated onto each respective sample before being evaluated by the discriminator, thereby incentivizing the generator to learn the true binomial distribution. This technique also serves as a guard against mode collapse, as a lack of variety in samples produced by a collapsed generator would be manifested in the minibatch average and consequently penalized during training. It should be noted that the results in Table 1 could be reproduced by simple independent sampling of the known Bernoulli probabilities for each variable, however, any meaningful relationships between variables would be lost.

The preservation of key variable relationships was first assessed by conducting a bivariate correlation analysis on the real and generated samples. The strength and direction of correlation between each of the binary variables in the real and generated hospital discharge data are given in Figure 4. Qualitatively, the predominant correlations observed in the real records (a) are preserved in the generated records (b).

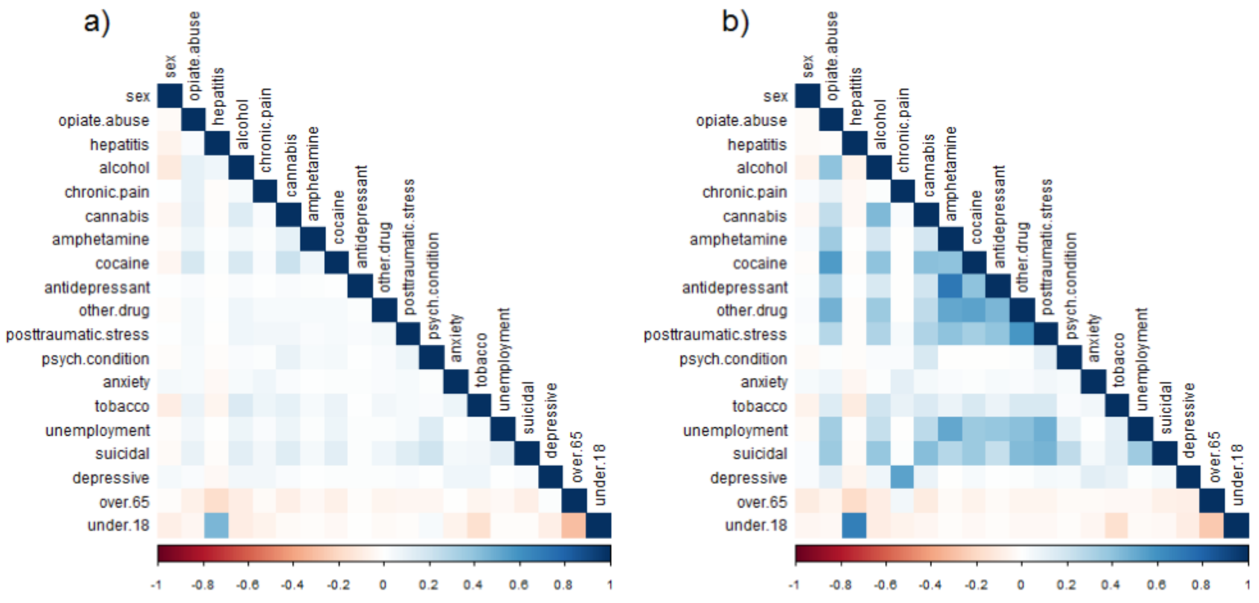


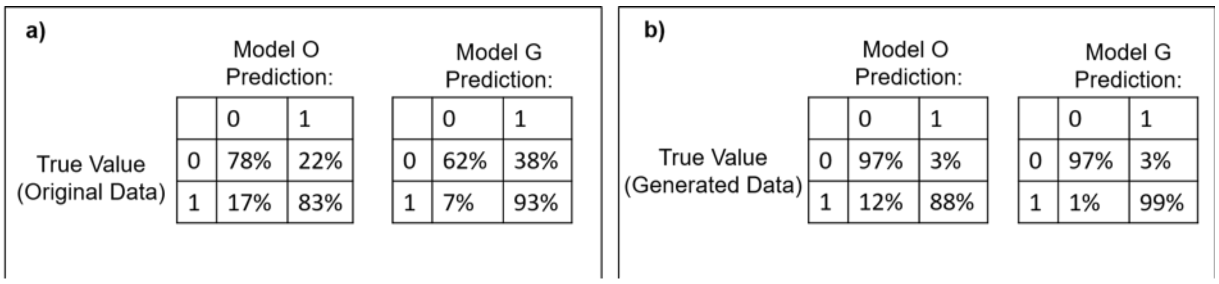
FIGURE 4: Pearson correlation coefficient matrices for real (a) and generated (b) hospital discharge records.

For example, a relatively strong positive correlation exists between hepatitis diagnosis and patients under 18 years of age in both real and generated data, as well as a moderate negative correlation between hepatitis and patients over the age of 65. Additionally, cooccurrences of diagnosis codes related to drugs of abuse in the real data were reproduced in the generated samples, as indicated by positive correlations between cocaine, opiate, alcohol and cannabis use. Interestingly, variable correlations present in the real data appear to be amplified in the generated samples. Furthermore, there are idiosyncratic correlations in the generated data, such as that between depression and chronic pain diagnoses. While the exact cause of this is unknown, one possible explanation could be that many of the diagnoses in our dataset are rare enough that their cooccurrence in generated sample batches is not heavily penalized during training. On the contrary, cooccurrence of the abundant age indicator binaries *under 18* and *over 65* seems to have been successfully down regulated by the discriminator. Of the 4,630 generated records where patient age was less than 18 or greater than 65, only 33 showed positive values for both fields.

A PREDICTIVE MODEL FOR OPIATE USAGE

To demonstrate the potential of doing research using GAN-generated data, we trained a simple random forest model to predict the *Opiate Usage* variable based on the other 18 variables. We trained one model on the generated data (Model G), and we trained another model using a subset of the original data (Model O). Then both models were tested against a separate subset of the original data. The accuracy of each model is shown by the confusion matrices in Figure 5a. Model G shows a higher number of false positives (38% compared to 22% in Model O), but a similar accuracy overall.

Both models were also tested against a separate subset of the generated data. While it is more relevant to test against the real data, the results show that both models have greater accuracy when predicting the *Opiate Usage* variable in the generated data. This suggests that the GAN may be overfitting to some degree and generating patterns that show up more frequently in the data.



a)

True Value (Original Data)

Model O Prediction:

	0	1
0	78%	22%
1	17%	83%

Model G Prediction:

	0	1
0	62%	38%
1	7%	93%

b)

True Value (Generated Data)

Model O Prediction:

	0	1
0	97%	3%
1	12%	88%

Model G Prediction:

	0	1
0	97%	3%
1	1%	99%

FIGURE 5: Confusion matrices for the predictive models of Opiate Usage. Model O is trained on the original data and Model G is trained on generated data. The performance is tested on the original data (a) and on generated data (b).

NUMERICAL TABLES: SENSOR DATA FROM MINE HAUL TRUCKS

Generative adversarial modeling of time series data is a nascent field of research. To date, only two examples are published: RGAN and GAN-AD (C. Esteban; D. Li). The former was devised to generate real-valued univariate medical time series data, while the latter effectively generated multivariate, albeit PCA-reduced, signals in the scope of an anomaly detection framework. Both architectures employ recurrent neural networks (RNNs) as feature extractors in the generator and discriminator. Our approach instead uses 1-D convolution, which, like RNNs, has seen wide application in time series-based problems. Here, we rely on successive convolution operations to capture both correlation and autocorrelation in the sensor signals.

The structure of our GAN is like that of the popular Deep Convolutional GAN (DCGAN), with the primary exception being that successive convolutions are one-dimensional operations (Figure 6). The input to the discriminator is a depth-wise stack of vectors of length *t*, where each vector is a unique sensor variable that is temporally aligned with its neighboring variables. A key distinction between our network and conventional convolutional GANs is the absence of pooling in the discriminator. We found that the quality of generated samples was degraded by down-sampling, and thus carried the input dimension t through the final fully connected layer.

Training was conducted with a set of 1,533 non-overlapping 1000-second sequences of six variables using stochastic gradient descent with momentum (0.9) and a fixed learning rate of 5x10^{-4} for 200 epochs. LeakyReLU activation with a scale factor of 0.2 was used at all layers except for the final layers of the generator and discriminator, where *tanh* and *sigmoid* functions were used, respectively.

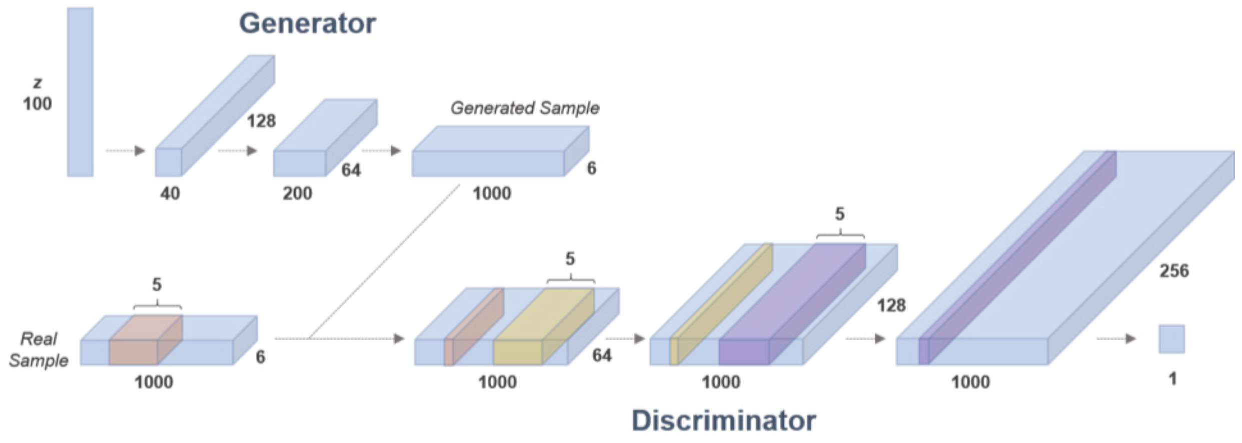


FIGURE 6: Architecture of 1-D convolutional GAN for multivariate time series modeling.

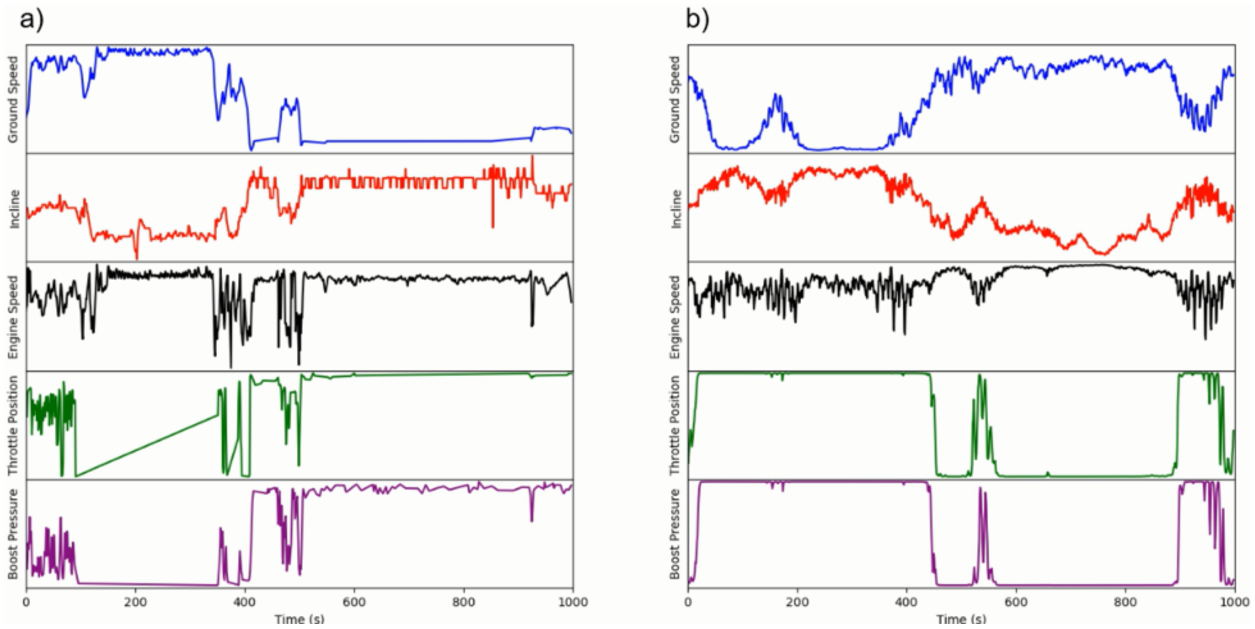


FIGURE 7: Sample time series from a) real and b) generated sensor data.

Representative sequences with a length of 1,000-seconds of real and generated sensor output are shown in Figure 7. We expect the signals for ground speed and incline to be relatively smooth compared to other sensor outputs, given that the source vehicles tend to travel at constant speed on road surfaces of consistent grade for minutes at a time. The real example in Figure 7a demonstrates this behavior, as the vehicle maintains high speed in a region of zero to negative incline, then transitions into positive incline, after which a sustained low speed is observed. This mode of operating is also apparent in the generated example (Figure 7b), and local variations in speed and incline are within reason. It is worth noting that the discrete nature of the real incline signal, caused by backend rounding of the raw sensor information, is reasonably approximated by continuous values in the generated sample but is not effectively reproduced.

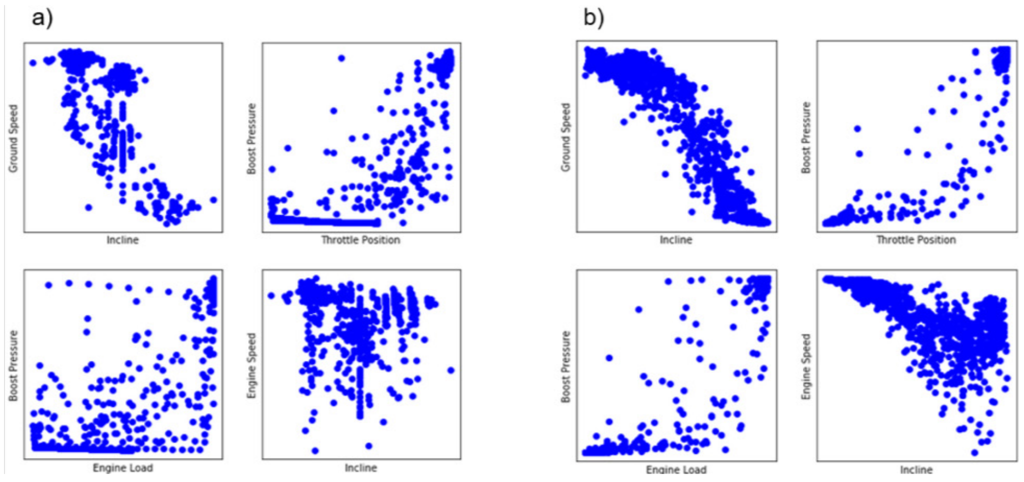


FIGURE 8: Bivariate plots of 1,000-second sequences of a) real and b) generated sensor data.

Sensor signals like engine speed, throttle position and boost pressure tend to be more erratic, as their magnitude can change substantially within a one-second time resolution during normal operation. However, these signals are often stable within time windows where the vehicle was traveling at steady speed. For example, both real and generated samples show extended periods of time where engine speed is nearly constant and boost pressure is at its lowest value, which is consistent with a vehicle coasting downhill without any throttle input. This mode of operating is confirmed by the incline and throttle position signals in Figure 7 (Note that the perfectly linear increase in throttle position in Figure 7a is due to a difference-based data storage paradigm and subsequent interpolation operating on the backend. Therefore, the real value for throttle position is assumed to be the first value in the linear sequence).

Key bivariate relationships for the same real and generated examples are plotted in Figure 8. As noted previously, a strong negative correlation between ground speed and incline is preserved in the generated samples. Furthermore, positive correlations between boost pressure and throttle position as well as boost pressure and engine load are evident in both real and generated data. A characteristic nonlinear relationship exists between boost pressure and throttle position, whereby low boost—high throttle states are observed but high boost—low throttle states are not. This is attributed to the mechanical operation of turbocharged engines, where large positive changes in throttle input coincide with a lagged increase in boost pressure, i.e., "boost lag." Interestingly, the generator appears to have learned this relationship.

A PREDICTIVE MODEL FOR TRUCK THROTTLE POSITION

Using the generated data with 50,000 records, we trained a simple random forest model to predict the *Throttle Position* for each second using the values of the other five sensors. The speed of the truck on a given incline, as well as the engine load and engine speed, are correlated with the throttle position, so it makes sense that it would be possible to predict this variable.

The model built with generated data (Model G) performs reasonably well. The position has been scaled between -1 and 1. In Figure 9, the predicted position (green) is generally at a similar level as the true value (blue) in the sample. We also trained a random forest model using a subset of the original data set (Model O). The output of this model is also shown in Figure 9 (red). The output of the two models are qualitatively alike. The mean absolute error in predictions of Model G is 0.31 and the mean absolute error in Model O is 0.18. There is some loss in model performance with the generated data, which should be expected.

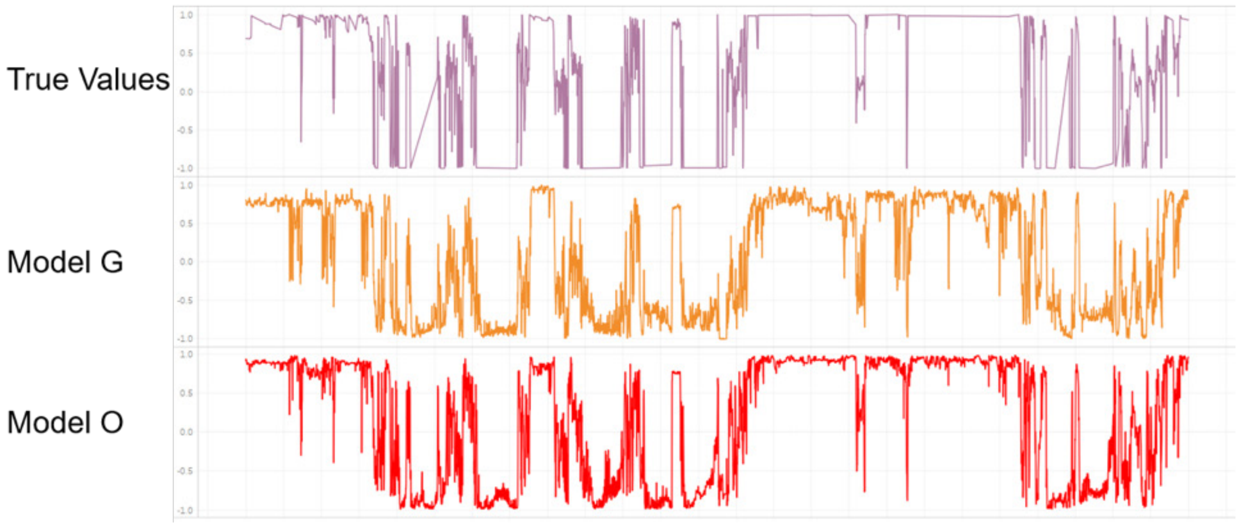


FIGURE 9: A sample of the true values of Throttle Position (top) and the predicted values of each model based on the other five sensor variables.

The comparative model performance provides another way to track the progress of the GAN training. As the GAN improves through training, a predictive model built with generated data should have an accuracy that is closer to a model built using the original data. While it may not be possible in every case to train a predictive model, when it is feasible, we can assess the performance of the GAN in a quantitative way using a predictive model, as well as using the marginal distributions and bivariate correlations in the generated data.

Conclusions

Generative adversarial networks have the potential to create representative data that can obscure sensitive information while maintaining key properties of the original data set. Many industries could use this process to anonymize their data sets so they can be released in a legal and ethical way to third parties for further analysis. Current methods of obscuring data generally will still contain portions of the original data, and sometimes only a small amount of data is needed to be personally identifiable. Certain types of records (e.g. healthcare, finances) have strict regulations in how they can be released. GAN-generated data offers a way to produce new records that have no identifying information.

Most of the GAN work to date has been on image data. In this paper we expanded the scope beyond image data and illustrated two examples of generating tabular data with a GAN. In the first example we trained a network to generate binary health records. In the second example we used a novel approach with 1-D convolutions to generate truck sensor data in time series.

For both examples, a simple model was trained to predict the value of one variable based on the rest of the variables in the data. Models were trained using the original data and the generated data. In these examples we see a small decrease in the predictive accuracy of the model built with generated data. However, the generated models prove that useful predictions would be possible even with the generated data. Researchers analyzing generated data could take their results back to the owners of the original data, who would then be able to perform the same analysis and validate the results.

Overall, this work demonstrated several novel ideas:

1. **GANs can successfully be used to generate representative data on tabular binary and time-series data.**
2. **Generated tabular data from a GAN can be leveraged to train a model that can then be used to predict on real values with good accuracy.**
3. **An architecture for GANs on time-series data that is similar to DCGAN, with the primary exception being that successive convolutions are one-dimensional operations.**

In future work, this approach should be taken for other forms of data: geospatial, text, images, etc. For GAN-generated images, models should be trained to classify certain objects or features in the images. Artificial intelligence and neural networks are being used to classify many kinds of healthcare-related images, and several image data sets have been made public. Using our approach, a GAN could be trained to generate representative samples of these images and neural networks could be trained to classify them. The performance of these models could then be compared to existing classification models. This would further prove the usefulness of GANs in obscuring healthcare data for research purposes.

References

C. Esteban, S. L. Hyland, G. Rätsch. "Real-valued (medical) time series generation with recurrent conditional GANs." *arXiv:1706.02633* (2018).

D. Li, D. Chen, J. Goh, S. Ng. "Anomaly detection with generative adversarial networks for multivariate time series." *arXiv:1809.04758* (2018).

E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, J. Sun. "Generating multi-label discrete patient records using generative adversarial networks." *Machine Learning for Healthcare* (2017).

Hjelm, D. and A. Jacob. "Boundary-seeking GANs: A new method for adversarial generation of discrete data." April 2018. *Microsoft Research Blog*.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio. "Generative adversarial nets." *Advances in neural information processing systems*. 2014.

Sweeney, L. "Simple demographics often identify people uniquely." Pittsburgh: Carnegie Mellon University, Data Privacy Working Paper 3, 2000.

T. Karras, T. Aila, S. Laine, and J. Lehtinen. "Progressive growing of GANs for improved quality, stability, and variation." *International Conference on Learning Representations* (2018).

World Wide Technology

ARTIFICIAL INTELLIGENCE
RESEARCH & DEVELOPMENT

Learn more about WWT's Artificial Intelligence Research & Development at:
https://www.wwt.com/artificial-intelligence-research-and-development

ABOUT WWT ARTIFICIAL INTELLIGENCE RESEARCH & DEVELOPMENT

The Artificial Intelligence Research & Development program at WWT is an applied research initiative focused on investigating the one to three-year horizon of the artificial intelligence space. The AI R&D program conducts internal projects grounded in WWT's deep understanding of industry use cases and produces reusable components and white papers to share with customers and the AI community.

Through strategic partnerships, cutting-edge data science and ML Ops skills, and ability to test and learn in the Advanced Technology Center and public cloud, the AI R&D team advances WWT's knowledge of the AI and ML space, thus allowing WWT to be on the bleeding edge and remain strategic advisors to businesses in their AI needs.

ABOUT WWT

Founded in 1990, WWT has grown to become a global technology solution provider with nearly $12 billion in annual revenue. With thousands of IT engineers, hundreds of application developers and unmatched labs for testing and deploying technology at scale, WWT helps customers bridge the gap between IT and business. By bridging leading technology companies together in a physical yet virtualized environment through its Advanced Technology Center, WWT integrates individually impressive technologies to produce game-changing solutions.

Based in St. Louis, WWT employs more than 6,000 employees and operates over 4 million square feet of warehousing, distribution and integration space in more than 20 facilities throughout the world.