Proceedings of the 2021 Winter Simulation Conference
S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo, and M. Loper, eds.

CHALLENGES AND OPPORTUNITIES FOR GENERATIVE METHODS IN THE CYBER DOMAIN

Marc Chalé

Department of Operational Sciences
Air Force Institute of Technology
2950 Hobson Way
Wright-Patterson AFB, OH, 45433, USA

Nathaniel D. Bastian

Army Cyber Institute
United States Military Academy
2101 New South Post Road
West Point, NY, 10996, USA

ABSTRACT

Large, high quality data sets are essential for training machine learning models to perform their tasks accurately. The lack of such training data has constrained machine learning research in the cyber domain. This work explores how Markov Chain Monte Carlo (MCMC) methods can be used for realistic synthetic data generation and compares it to several existing generative machine learning techniques. The performance of MCMC is compared to generative adversarial network (GAN) and variational autoencoder (VAE) methods to estimate the joint probability distribution of network intrusion detection system data. A statistical analysis of the synthetically generated cyber data determines the goodness of fit, aiming to improve cyber threat detection. The experimental results suggest that the data generated from MCMC fits the true distribution approximately as well as the data generated from GAN and VAE; however, the MCMC requires a significantly longer training period and is unproven for higher dimensional cyber data.

1 INTRODUCTION

The year 2021 was marred as the year when concern of cybersecurity transitioned from a niche academic and professional concern to an everyday priority. In late 2020, a massive attack on the SolarWinds information technology management platform resulted in 18,000 customers installing malicious code into their networks. Among the victims were defense contractors, government agencies, and hospital systems (Jibilian and Canales 2021). The intricate attack is believed to have gone undetected for months and may be linked to previously undetected exploitations (Volz and McMillan 2020). The reported frequency and damage associated with cyber attacks has increased steadily between 2005 and 2018 (Evans and Smith 2019). We contend that this number may be biased by organizations unable to detect or unwilling to divulge damaging attacks (Evans and Smith 2019). The cyber threat against friendly assets extends beyond information systems to the Internet of Battlefield Things (IoBT).

It is anticipated that future wars will be fought largely in the cyber domain but there is little precedent to understand what such a war would look like. The 2014 Russian occupation of Crimea showcased the synchronization of cyber exploitation with kinetic attacks (Collins 2018). A 2018 report outlines that foreign actors used malware to access sensitive information from the Naval Undersea Warfare Center (Evans and Smith 2019). Many legacy cyber-physical systems are built with no foresight to cyber vulnerabilities (Applegate 2013). As the IoBT grows to include more of these systems, the potential cyber threat exposure also grows. It is critical that we address our cybersecurity shortcomings with novel techniques before adversaries exploit them. Intrusion detection systems (IDS), for example, are an approach to deter and defend from cyber attacks (Applegate 2013). The IDS scans log files of cyber data traffic through network or on host devices for evidence of malicious behavior. Machine learning (ML) based IDS require large

data sets for training (Chalé et al. 2020) and organic cyber attack data, with labelled entries, is notoriously scarce.

Generative modeling techniques, stemming from the fields of statistical computing and artificial intelligence, have the potential to address the data deficiency. Methods such as Markov Chain Monte Carlo, variational autoencoders (Gelman et al. 2013), and generative adversarial networks (Bengio et al. 2017) are used to estimate the probability distribution functions of existing multivariate data sets. It is then possible to draw new, synthetic data from these generative models. This research seeks to understand which existing generative approaches are best suited for generating realistic synthetic cyber data. Specifically, this work reports the performance of three generative methods and suggests future work to address their shortcomings. We expect these findings will enhance the design and training of more effective IDS.

This paper is structured as follows. Section 2 outlines the evolution of IDS, cyber data sets and how generative modeling techniques play a role. Section 3 describes the generative methods under investigation, to include presenting an application of generative methods to model the NSL-KDD cyber data set. Section 4 describes the computational experimentation, to include the results of using the three generative methods for generating synthetic cyber data and an approach for assessing the quality of the synthetic data. Conclusions, recommendations and future work are described in Section 5.

2 Literature Review

Cyber borne threats have plagued computer networks since the early days of ARPANET (Chen and Robert 2004). By 1971, researchers at Bolt Beranek and Newman Inc. implemented the concept of "self-replicating automata" (Chen and Robert 2004) which was previously postulated by Neumann, Burks, et al. (1966). This experimental program was itself benign, but it paved the way for malignant worms and viruses (Chen and Robert 2004). Cybersecurity, the scientific approach of protecting data from illegal access and alteration, has become an important and constantly evolving field.

In 1972, a panel of experts presented a series of cybersecurity vulnerabilities to the U.S. Air Force Systems Command. The report outlined redesign requirements for secure information systems. The cost of *inaction* against threats was determined to be greater than the cost of securing the vulnerable systems (Anderson 1972). A subsequent report advocated for increased logging of network traffic; illegal users exhibit abnormal behavior which could be detected during network audits.

2.1 Intrusion Detection Systems

Intrusion detection systems are a distinct layer of cybersecurity that detect and react to malicious activities within networks or at a host device. Denning (1987) describes six components of an IDS, as well as best practices for logging activity, auditing records, and deploying responses; Axelsson (2000) provides a more recent, though now dated chronicle of IDS. IDS detectors may follow a signature-based strategy where information from prior records are used to flag specific attack behavior. Alternatively, anomaly-based IDS detect statistical abnormalities in the data logs, which are also indicative of an attack (Stallings et al. 2012). A comprehensive review of anomaly detection techniques is given by (Chandola et al. 2009). Some recent IDS employ a hybrid of both methods (Patcha and Park 2007).

Japkowicz et al. (1995) performs classification by training an auto-encoder on a positive data class only. Once trained, data examples that yield high error in the auto-encoder are adjudicated as anomalies, and therefore members of the negative class. This approach is desirable when data on the negative class is scarce. Butt (2018) applies this method to intrusion detection, however the project warrants a revisit. We emphasize guidance by Patcha and Park (2007) that the mean square error threshold for classification should be tuned during a validation testing to optimize a performance metric such as recall, and we direct the interested reader to best practices for model selection (James et al. 2013).

A compelling anomaly detection technique based on Markov Chain Monte Carlo modeling has been successfully applied to IDS (Scott 2001). The model uses an indicator variable to fit data to either baseline

traffic or contaminated traffic. Posterior analysis informs the probability of contamination. Patcha and Park (2007) cautions that anomalous network records are not always malicious and that signature-based detectors often exhibit superior false negatives on known attack types. Therefore, we must also consider the strengths and weakness of signature-based IDS.

Signature-based IDS excel when there is a bounty of training examples for normal and illegal activity though the model must be updated frequently as new attack signatures are discovered (Patcha and Park 2007). These discriminators may struggle to identify contextual evidence of attacks that span multiple records. Patcha and Park (2007) and Hindy et al. (2020) advocate that signature-based IDS must be updated frequently to reflect newly discovered attack types, but they do have excellent detection rates on known attack types. Signature-based systems are unable to detect a zero day attack; that is a novel attack type exploiting unknown vulnerabilities. Unfortunately, they also struggle to model contextual evidence of attacks that spans multiple records (Patcha and Park 2007; Hindy et al. 2020).

Among 85 recently published IDS manuscripts reviewed by Hindy et al. (2020), the vast majority of IDS detectors employ machine learning techniques. Chalé et al. (2020) leverage a signature-based approach that uses a meta-learner to predict the best candidate classification algorithm on unseen cyber data; they experimentally concluded that publicly available cyber data is insufficient to train an effective IDS, a sentiment shared by Hindy et al. (2020). In general, the extraction of features from network traffic for use in training ML-based classifiers for IDS is a necessary pre-processing step. Maxwell et al. (2019) experimentally analyze various methods of feature engineering for ML-based IDS. However, subject-matter-expertise is often required to extract optimum features, so De Lucia et al. (2021) avoid the feature engineering problem by developing novel algorithms using deep learning for IDS using only raw network traffic.

2.2 Cyber Data Sets

According to Hindy et al. (2020), cyber databases become obsolete almost immediately as attackers constantly evolve their strategies. Databases almost never reflect real-world cyber traffic with realistic proportions of varying attack types and background traffic. The databases don't reflect temporal changes that occur in modern networks. Viegas et al. (2017) provides specific requirements for IDS training data sets; notably, they should be easily amendable. Sharafaldin et al. (2018) advocates for effective documentation so greater audiences can utilize and update public databases. Hindy et al. (2020) identifies a literature gap in metrics to assess realness of IDS databases. To this point, we direct the curious reader to *machine learning efficacy* (Xu et al. 2019), which assesses the databases usefulness as a training set for an arbitrary ML task. This metric may be a starting point for future work assessing realness. We offer our support to the sentiment that there should be better community coordination in maintaining the taxonomy of cyber threats (Hindy et al. 2020) and evolving databases. For instance, Tavallaee et al. (2009) critiqued the statistical properties of the well cited KDD CUP 99 data set and provided an improved version, NSL-KDD, to the community.

2.3 Generative Methods

Generative methods are a excellent tool set when the measurable relationships in a system are complex, possibly beyond the useful scope of deterministic models (Jebara 2012). The premise of generative modeling is to estimate a joint probability distribution of the variables in a system and many techniques heavily leverage Bayes' theorem. Evidence from data refines prior beliefs about the form of a model, and in many cases estimates parameters in a closed form model. Whether or not a closed form model is produced, generative models are often used to generate new data from the same statistical distribution as the training data (Jebara 2012).

Generative machine learning (GML) stands in contrast to discriminative machine learning (DML) which does not directly model the distribution of data, but nonetheless models the conditional relationship of the

class label given predictor information. DML methods typically generate decision boundaries for class labels (Jebara 2012). DML includes methods such as support vector machine and logistic regression in which the joint distribution of data is not modelled. The joint distribution created by GML is in many ways more informative than the models from DML. Further, it is possible to test the correctness of the generated distribution against real data (Goodfellow 2017).

2.3.1 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) sampling stems from the field of Bayesian analysis. Many generative methods are unable to accommodate *high dimensional* data, where in some cases the dimensionality is greater than the number of records. The complexity of these models lend to exponentially increasing computational time. Another situation is intractable inference, where a conditional distribution is estimated via an intensive summation over other variables marginal probability density functions. Lastly, intractable normalization constant is problematic for certain function classes including logistic, where gradients are difficult to estimate. MCMC methods are appealing in these cases (Bengio et al. 2017).

As a method within the broader field of Bayesian data analysis, we follow three high level steps of analysis. Firstly, the full probability model is defined in the form consistent with prior knowledge, including all observable and latent variables in the system. Second, we find the posterior distribution, that is the probability distribution of the latent variables conditioned on observed values. Estimates on the posterior distribution provide our approximate joint distribution. Finally, we evaluate the model according to fit, utility, and correctness of assumptions then assess any necessary revisions. In some applications, the Bayesian model, which conditions on observed data and incorporates prior beliefs, is often more informative than the frequentist approach, which calculates statistics from data alone (Gelman et al. 2013).

One of the primary appeals of the Bayesian modeling framework is the compatibility with multi-level, or hierarchical modelling. These models incorporate parameters for phenomena at different levels in the system, each which plays a role in modelling the joint distribution. These models provide useful predictions in complex systems (Gelman et al. 2013).

MCMC generative methods find model parameters θ that generate a posterior distribution $p(\theta|y)$ that matches a target distribution. It works by drawing parameters θ from a proposal distribution, assessing the correctness of the draw, and progressing towards a better vector θ. Markov chain convergence occurs when the samples of θ exhibit a stationary distribution. The Markovian property implies that the sampling sequence leads to the same region of the parameter space regardless of initial point θ^0. Therefore, multiple chains achieving the same stationary distributions provide evidence that proper convergence has occurred.

Several sampling methods of MCMC modeling have unique advantages. Gibbs sampling is the simplest form of MCMC model learning. The parameter space of the model is divided into d subsets. Within an iteration of the Gibbs sampler, the algorithm iterates through the d subset of parameters. Within each of d steps of the iteration, a subset of parameters are drawn from a posterior distribution which has been conditioned on the current values of all other parameters outside the subset. Under the reasonable assumption that the chain follows an irreducible, aperiodic, non-transient Markov process, the Gibbs sampler converges to a stationary output in a finite number of iterations (Gelman et al. 2013).

Metropolis-Hasting sampling is differentiated from the Gibbs sampler by relying on independent parameter sampling and by incorporating probabilistic acceptance of each step (Hastings 1970). The acceptance function promotes steps in high density regions of the joint distribution, without getting stuck at any particular region. The algorithm satisfies the condition of *detailed balance* (Kass et al. 1998), which guarantees the posterior distribution generated by the algorithm converges to the target distribution in the data. In practice, the Metropolis-Hastings algorithm tends to spend fewer iterations stuck near local minima before converging to the joint distribution (Yildirim 2012).

2.3.2 Generative Adversarial Networks and Autoencoders

Generative adversarial networks (GAN) is a ML technique for building generative models introduced by Goodfellow et al. (2014) and updated in Goodfellow et al. (2020) with recent insight. GANs are built using a game-theoretic, adversarial process. On one side, a multi-layer perceptron (MLP) generates data (the generator), initially with random weights, while on the other side, a discriminative MLP (the discriminator) detects whether or not the generated data matches the distribution of a "target" set. Initially, the discriminator set can easily distinguish generated data examples from true example, but the generator improves its model over time. Simultaneously, the discriminator learns on its mistakes and improves its model. Eventually, the generator produces examples that match the original distribution very well, and the discriminator cannot detect a difference in the generated examples (Goodfellow et al. 2014).

There are instances of GANs used for cyber data. Yao et al. (2018) provides a semi-supervised generative approach for fabricating realistic cyber data that only requires labels on 10% of training examples. Alhajjar, Maxwell, and Bastian (2020) use GANs to generate adversarial examples to fool ML-based network IDS classifiers. CTGAN is collection of deep learning based synthetic data generators for single table data (with both categorical and continuous), which are able to learn from real data and generate synthetic clones with high fidelity. The GAN is trained by conditioning on the discrete variables and perceived mode within the continuous variables. Results indicate that CTGAN generates synthetic tabular data, which is both a statistical fit to baseline data and performs well in data science applications (Xu et al. 2019).

Autoencoders are unsupervised ML algorithms that learn to encode an input data set to a smaller summary code, then process the encoded data such that the original set is recovered with minimal error. The encoder function, $\mathbf{h}$, is typically a single hidden layer artificial neural network, where $\mathbf{h} = f(\mathbf{x})$. The decoder, $\mathbf{r} = g(h)$ is the complimentary network responsible for transforming h back to its original form; it typically has a single hidden layer as well. The models are generally trained with minibatch gradient descent and back-propogation. A properly trained autoencoder learns $g(f(\mathbf{x})) = \mathbf{x}$ for all $\mathbf{x}$ in the entire domain (Bengio et al. 2017). The primary advantages of an autoencoder are to learn which data features are most important to describe the system, and to compress data with minimal reconstructive error.

3 Methodology

The methodology seeks to employ and compare three different generative methods in their capability to generate synthetic, realistic network IDS data. Since cyber data may contain a vast variety of variables, the methods must model the joint probability distribution of discrete and continuous features.

3.1 NSL-KDD Cyber Data

The NSL-KDD cyber data set is selected as the baseline data set because it is well studied in the literature, and it has improved statistical properties over the predecessor KDD CUP 99 data. Three variables of the NSL-KDD cyber data set were selected for statistical modeling as part of this research. "Attack Type" is binarized as *attack* or *normal*. Additionally, the two continuous variables selected are `count` and `Srv_count`. Mini-max normalization is performed on each continuous variable. These variables were chosen because they are important features for IDS applications. They are few enough that we can experiment with computationally demanding models and plot results in $\mathbb{R}^2$ for intuitive analysis. Using both continuous and categorical variables provides a proof of concept that can be expanded into higher dimensions in future studies.

The real NSL-KDD data is plotted in Figure 1, a pair plot of the `count` and `Srv_count` continuous variables. The plots demonstrate several linear trends. The marginal distributions, which are conditioned on the categorical variable `Attack_Bin`, show that there are multiple modes, though it is difficult to discern the exact quantity from the plot.

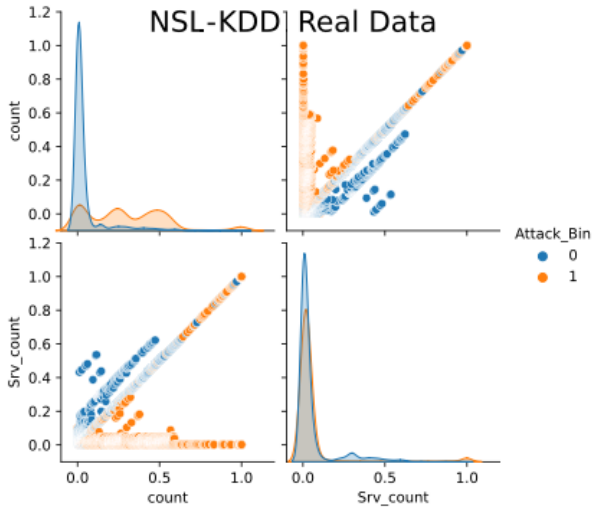


Figure 1: Real NSL-KDD Cyber Data

3.2 Conditional GAN and Variational Autoencoder

The Python implementation of CTGAN (Xu et al. 2019) is used to model cyber data and generate synthetic data, with an emphasis on data sets with both continuous and discrete variables as well as multiple modes and data imbalance. By conditioning on discrete categories, the GAN mitigates mode collapse, an effect where minor categories are lost in the training. It also bypasses the problem of non-sparse one-hot vectors generated by the GAN (Xu et al. 2019). Below is a summary of the CTGAN process.

Mode-specific normalization is used to improve the training of the conditional GAN. First, a univariate Gaussian mixture model is fit to each continuous variable. Then, mode membership is assigned for all values in that column from the training set. Finally, the values are normalized within their Gaussian cluster. Both mode membership and normalized value are input to the GAN. Both the generator and discriminator have a fully-connected architecture, as depicted in Figure 2a, to motivate learned relationships between data columns. The input of CTGAN's generator are a condition vector and Gaussian noise. The input of the discriminator are real rows, conditioned on a discrete factor, as well as generated rows. α_{ij} represents a mode specific normalized continuous value for continuous variable i, whereas β_{ij} represents a realization of one hot vector for discrete variable i; the condition vectors are denoted by d_{ij}. The two hidden layers employ relu with batch activation, while tanh and gumblesoftmax are used to accommodate scalar and discrete output features, respectively. Complete details on the conditional GAN model are presented in (Xu et al. 2019). The hallmark of a conditional GAN is that additional information is passed to the generator to condition training on discrete treatments. CTGAN provides a *mask vector* informing that one level is set for one particular discrete column. The generator learns to relay this vector, without modification, and simultaneously optimize weights to generate plausible rows of synthetic data.

A variational autoencoder based deep learning data synthesizer (TVAE) is also specified in Xu et al. (2019) with an implementation provided in the synthetic data vault Python library. The model estimates the joint distribution of the data via a compression and a decoding network, as depicted in Figure 2b. The encoder contains two hidden layers with relu activation functions and an output layer with softmax activation for discrete variables and tanh activation for continuous variables. Gaussian noise is then applied to continuous variables. The decoder network contains two hidden layers using relu activations and applies Gaussian noise to the output.

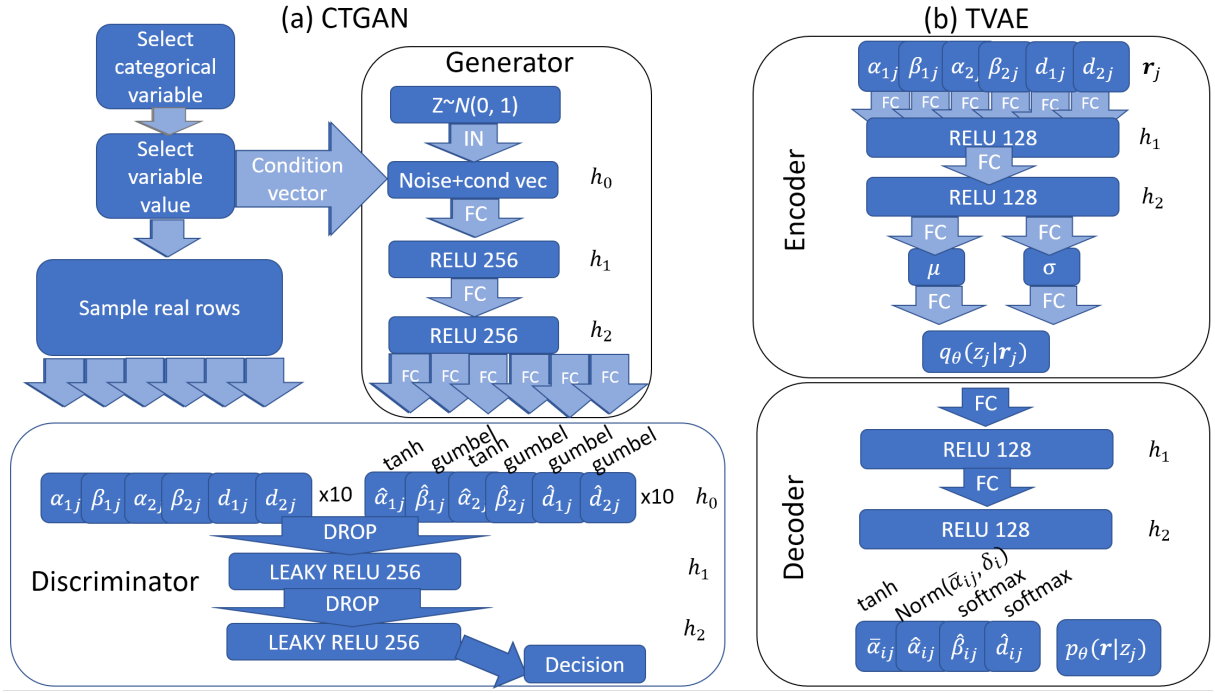


Figure 2: Network Architectures for CTGAN and TVAE Generative Methods

3.3 Conditional Hierarchical MCMC

The Bayesian approach to generative modeling is premised on updating a hypothesis as known information becomes available. Hierarchical probabilistic models use a relational structure such that higher level parameters inform lower level distributions that in turn inform the most probable parameters in the target joint distribution. Following the precedent of Teh (2010) and Liew et al. (2019), we develop a Gaussian mixture model that can fit up to three modes in the $\mathbb{R}^3$ space containing the variables. Notably, if fewer than three modes are present, the model may fit a weight of zero to any particular component. Since the continuous variables in our cyber data set are all positive, concentrated near zero, we employ the *reflection trick*. A copy of each data point is added to the set, with all continuous variables negated. This motivates the primary Gaussian cluster to center over zero, but allows the other two clusters to converge elsewhere in the space.

MCMC sampling with the Metropolis-Hastings algorithm was initially performed on the previously described 3-dimensional data set. Results demonstrated that the algorithm converged to reasonable estimated of cluster means. However, the covariance matrices often reflected unreasonably high spread in the categorical variables. That is, the Gaussian distribution centered over binary level of -1, was wide enough to largely cover the +1 level, vice versa. This produced an unacceptable rate of mislabeled points when new data was generated from the learned distribution. Tuning the model did not resolve the issue. This approach also forced us to use a model $\in \mathbb{R}^3$ despite the observation that our three variate data is more naturally modelled $\in \mathbb{R}^2 \cup \{0, 1, ..., j\}^2$ where j is the number of levels for the particular categorical variable.

We took inspiration from Mirza and Osindero (2014), where statistical models are conditioned on given information. In our case, we conditioned on each level of the discrete variable, `Attack_Bin`. MCMC sampling was performed to create a model on each of the two subsets. Since we observe the proportion of rows that are normal and attacks, we can combine generated data that are attacks and generated data that are normal, in correct proportions, to estimate the joint distribution. Equations (1-5) define a hierarchical mixture model of multi-variable Gaussian distributions.

In Equation (1), we see that weight vector p is sampled from a Dirichlet distribution which is the natural conjugate to the multinomial distribution. Equation (2) shows that LKJ Cholesky is the distribution over the positive-definite, symmetric covariance matrix subject to shape parameter η. Equation (3) provides the distribution over the standard deviation for the LKJ Cholesky covariance, which is notably asymmetrical. Equation (4) presents a normal prior over the mean of each Gaussian cluster. Finally, Equation (5) is the mixed Gaussian model of the joint distribution, as a function of modeled parameters.

$$p \sim \mathsf{Dir}(\alpha = [100, 10, 1]), \in \mathbb{R}^K \tag{1}$$

$$\tau \sim \mathsf{LKJCholeskyCov}(\eta = 2, SD) \tag{2}$$

$$SD \sim \mathsf{halfCauchy}(\gamma = 1) \tag{3}$$

$$\mu_k \sim \mathsf{N}(\phi_k, \tau = 0.05 + 0.025 * k), \mu_k \in \mathbb{R}^D \ \forall \ k \in [K] \tag{4}$$

$$Y|p, \Sigma, \mu \sim \sum_{k=1}^{K} \ p_k \mathsf{N}(\mu_k, \Sigma_k) \tag{5}$$

The appeal of the hierarchical mixture model, Equations (1-5), stems from two possible advantages over GAN and VAE models. Firstly, parameters that define the joint distribution are reported in an interpretable, empirical mixture form for each discrete bin. Secondly, correlation between continuous variables is captured directly within each cluster. The GAN and VAE methods attempt to implicitly capture covariance within the parameters of their artificial neural networks. This strategy has demonstrated empirical success but risks improper network specification that can lead to high model bias or variance. Gaussian copulas are yet another method, which estimate the distribution of continuous columns and capture the covariance between columns. Common copula functions require at least one tunable parameter to properly estimate covariance in a joint distribution and it is difficult to automate model specification (Nelsen 2007).

4 Computational Experimentation

All computational experimentation was performed via the Massachusetts Institute of Technology (MIT) Supercloud (Reuther et al. 2018) with Intel Xeon Gold 6248 2 core processor, 384 GB RAM, running Python 3.7.3 in Jupyter Notebook 6.0.0. The variable `attack_type` is binarized as *normal* or `attack` and given the variable name `Attack_Bin`. Mini-max normalization is performed on the continuous variables `count` and `Srv_count`. A random subset of 10,000 rows was used for generative modeling, where 5,334 were normal and 4,666 were attacks (see Figure 1). The reflection operation was performed to obtain a negative clone of records in the data set, resulting in 20,000 data records for MCMC trials. By incorporating the negative clone for the continuous variables `count` and `Srv_count`, we provide symmetry about each axis, allowing Gaussian clusters to center over the origin for the MCMC model.

The conditional hierarchical MCMC model specified in Equations (1-5) was sampled with a warm up period of 25,000 iterations and 5,000 recorded iterations for each of the two subsets; two chains with unique seeds were performed on each subset. Stationary posterior distributions were observed for all chains after 25,000, sufficing for sampling termination. A trace plot for the weight variable, `p`, is presented in Figure 3, where chain one is plotted in blue and chain two is plotted in orange. Although both chains converge, they suffer from the phenomena of label switching due to model non-identifiability (Jasra et al. 2005). Ideally, the blue and orange posterior plots would be nearly identical for each parameter in each Gaussian cluster, but our results show nearly identical posterior plots attributed to non-matching Gaussian clusters. The informative prior distributions were intended to avoid non-identifiability, but were unsuccessful.

Parameters for the multivariate Gaussian mixture model were obtained from the MCMC trace. Note that 400 new data points were generated by sampling from each mixture model in proportion to the observed bin size. The absolute value of each data point was retained to compensate for the reflection transformation on continuous variables. The generated data from the MCMC model is plotted in Figure 4, which yielded

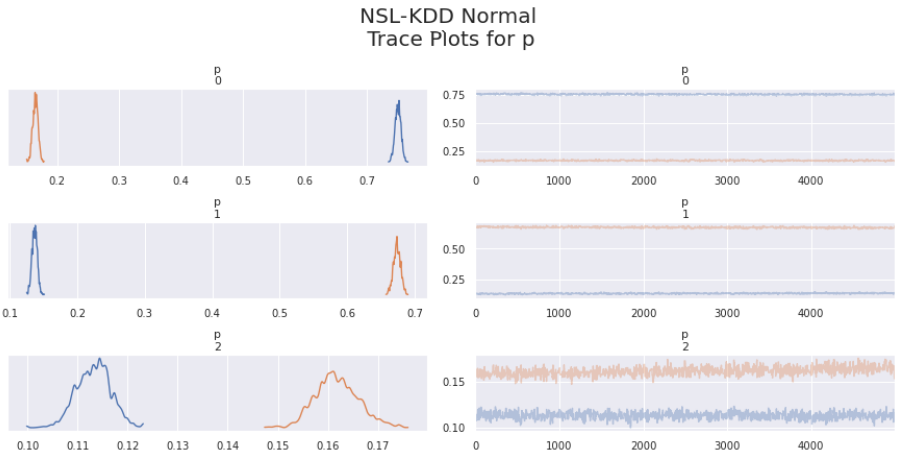


Figure 3: Trace Plot for Weight Vector p of the Gaussian Mixture

a score of 0.875 according to the inverted Kolmogorov–Smirnov (KS) D statistic (Patki et al. 2016). Note that a score of 1.0 indicates excellent fit and 0 indicates poor fit.

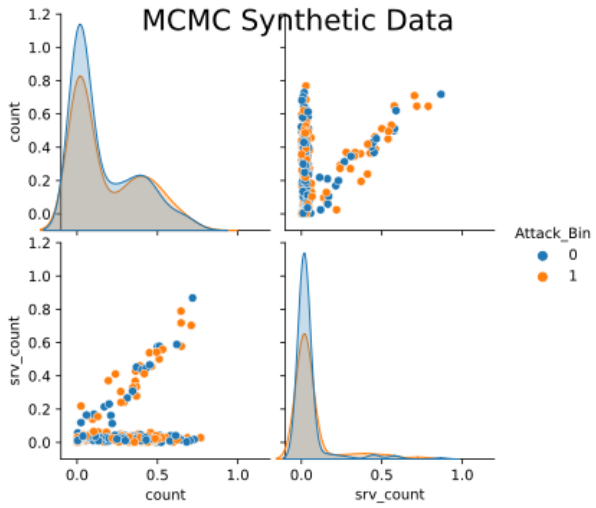
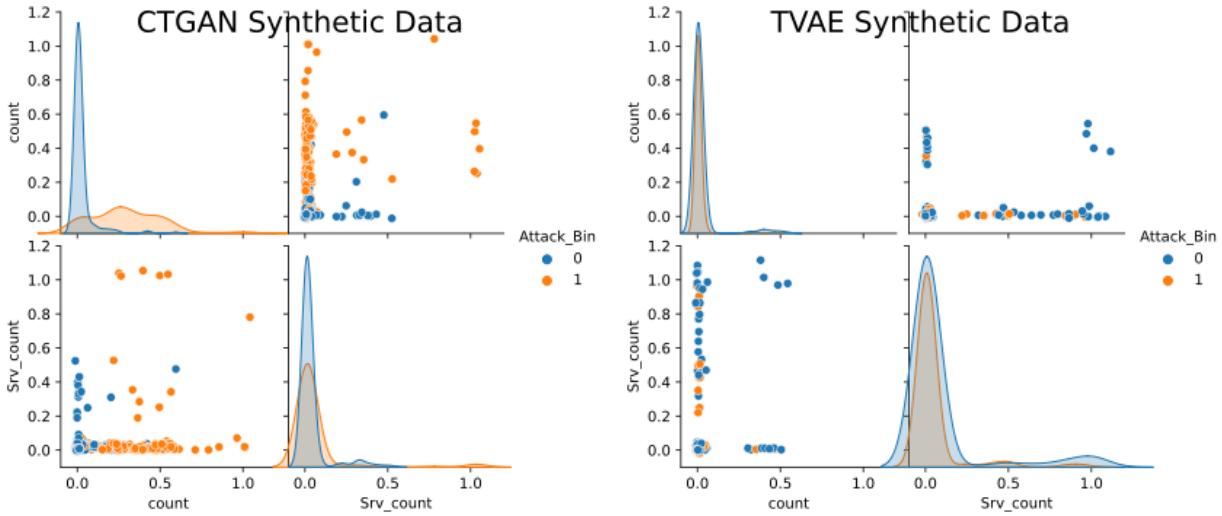


Figure 4: MCMC-based Synthetically Generated Cyber Data

The GAN model was implemented using the synthetic data vault Python package, CTGAN, with all default settings selected. The GAN model was fit with the same 10,000 data points as the MCMC model and 400 new data points were generated. The synthetically generated cyber data from the GAN model is plotted in Figure 5a, which yielded a score of 0.890 according to the inverted KS D statistic. Likewise, the VAE model was implemented using the synthetic data vault Python package, TVAE, with all default settings selected. Again, the VAE model was fit with the same 10,000 data points as the MCMC and GAN models, and 400 new data points were generated. The synthetically generated cyber data from the VAE model is shown in Figure 5b, which yielded a score of 0.723 according to the inverted KS D statistic.

A summary of performance results of these three generative methods is displayed in Table 1, where the inverted KS D statistic indicates goodness of fit between generated data and real data and the runtime indicates the computational demand to train each model. These results suggest that the cyber data generated from MCMC fits the true distribution approximately as well as the data generated from GAN and VAE methods. The GAN model performed best with a KS D statistic of 0.890, with MCMC close behind at



(a) CTGAN Generative Model

(b) TVAE Generative Model

Figure 5: GAN-based and VAE-Based Synthetically Generated Cyber Data

0.875 and the VAE poorest performing at 0.723. It should be noted that the MCMC model required a significantly longer training period compared to the GAN and VAE methods, and it is unproven for higher dimensional cyber data.

Table 1: Summary of the Generative Method Performance

	Inverted KS D	Runtime (mm:ss)
MCMC	0.875	38:53
GAN	0.890	4:07
VAE	0.723	2:54

5 Conclusions

As demonstrated in the experimentation, the MCMC, GAN, and VAE methods succeeded in producing synthetic network intrusion detection system data. The primary advantage of using the MCMC method to estimate the joint distribution of continuous variables in tabular data is to capture covariance *within* clusters. It also reports a closed form model with interpretable parameters. The GAN and VAE models are black box generative methods, which may provide inconveniences for certain users. Unfortunately, it is difficult for hierarchical MCMC models to converge in higher dimensions. Training times grow rapidly as parameters are added to the model, and models require manual tuning in order to converge in a reasonable time. As stated by Aitkin (2001), model formulation is further complicated if the number of modes is unknown prior to training. Future studies will determine mode quantity prior to MCMC modeling with methods such as variational Gaussian mixture models, K-nearest neighbor, or multi-dimensional mode-hunting. Alternatively, the Hierarchical Dirichlet Process infers the number of modes during training, but is a more complicated MCMC model. The goal of this work was to compare MCMC, GAN and VAE methods for generating realistic network intrusion data. Although the cyber data generated from the MCMC model fit the real data well, it took much longer to train and tune the model. Thus, MCMC should be pursued for applications such as cyber data generation where realism is more important than run time. Future research may also address whether meta-learning can reduce tuning efforts and training times.

DISCLAIMER

The views expressed in this paper are those of the authors and do not reflect the official policy or position of the Air Force Institute of Technology, the United States Military Academy, the United States Air Force, the United States Army, the Department of Defense, or the United States Government.

ACKNOWLEDGMENTS

This work was supported in part by the National Security Agency Laboratory for Advanced Cybersecurity Research under Interagency Agreement No. USMA21035, along with the GEN Omar N. Bradley Foundation Officer Research Fellowship in Mathematics. We offer a special thank you to Dr. Eric Brooks who enriched the research experience by generously sharing his knowledge of probabilistic programming. We would also like to thank Dr. Jeffery Weir for his guidance and mentorship throughout the research process.

REFERENCES

Aitkin, M. 2001. "Likelihood and Bayesian analysis of mixtures". *Statistical Modelling* 1(4):287–304.

Alhajjar, Elie and Maxwell, Paul and Bastian, Nathaniel D 2020. "Adversarial Machine Learning in Network Intrusion Detection Systems".

Anderson, James P 1972. "Computer security technology planning study—Vol 1".

Applegate, S. D. 2013. "The dawn of kinetic cyber". In *2013 5th international conference on cyber conflict (CYCON 2013)*, 1–15. IEEE.

Axelsson, S. 2000. "Intrusion detection systems: A survey and taxonomy". Technical report, Chalmers University of Technology.

Bengio, Y., I. Goodfellow, and A. Courville. 2017. *Deep learning*, Volume 1. MIT press Massachusetts, USA:.

Spencer A. Butt 2018. "Cyber Data Anomaly Detection Using Autencoder Neural Network".

Chalé, M., N. D. Bastian, and J. Weir. 2020. "Algorithm Selection Framework for Cyber Attack Detection". WiseML '20, 37–42. New York, NY, USA: Association for Computing Machinery.

Chandola, V., A. Banerjee, and V. Kumar. 2009, July. "Anomaly Detection: A Survey". *ACM Comput. Surv.* 41(3).

Chen, T. M., and J.-M. Robert. 2004. "The evolution of viruses and worms". *Statistical methods in computer security* 1:1–16.

Collins, Liam 2018. "Learning from Russia's Information Offensives".

De Lucia, M. J., P. E. Maxwell, N. D. Bastian, A. Swami, B. Jalaian, and N. Leslie. 2021. "Machine Learning Raw Network Traffic Detection". In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications III*, Volume 11746, 117460V–1. International Society for Optics and Photonics.

Denning, D. 1987. "An Intrusion-Detection Model.". *IEEE Transactions on Software Engineering, Software Engineering, IEEE Transactions on, IIEEE Trans. Software Eng* SE-13(2):222 – 232.

Evans, Corbin and Smith, Christopher 2019. "Beyond Obfuscation: The Defense Industry's Position within the Cybersecurity Policy".

Gelman, A., J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. 2013. *Bayesian data analysis*. CRC press.

Ian Goodfellow 2017. "NIPS 2016 Tutorial: Generative Adversarial Networks".

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. 2014. "Generative adversarial nets". In *Advances in neural information processing systems*, 2672–2680.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. 2020, October. "Generative Adversarial Networks". *Commun. ACM* 63(11):139–144.

Hastings, W. K. 1970. "Monte Carlo sampling methods using Markov chains and their applications".

Hindy, H., D. Brosset, E. Bayne, A. K. Seeam, C. Tachtatzis, R. Atkinson, and X. Bellekens. 2020. "A Taxonomy of Network Threats and the Effect of Current Datasets on Intrusion Detection Systems". *IEEE Access* 8:104650–104675.

James, G., D. Witten, T. Hastie, and R. Tibshirani. 2013. *An introduction to statistical learning*, Volume 112. Springer.

Japkowicz, N., C. Myers, M. Gluck et al. 1995. "A novelty detection approach to classification". In *IJCAI*, Volume 1, 518–523. Citeseer.

Jasra, A., C. C. Holmes, and D. A. Stephens. 2005. "Markov Chain Monte Carlo Methods and the Label Switching Problem in Bayesian Mixture Modeling". *Statistical Science* 20(1):50–67.

Jebara, T. 2012. *Machine learning: discriminative and generative*, Volume 755. Springer Science & Business Media.

Jibilian, Isabella and Canales, Katie 2021. "Here's a simple explanation of how the massive SolarWinds hack happened and why it's such a big deal". https://www.businessinsider.com/solarwinds-hack-explained-government-agencies-cyber-security-2020-12, accessed March 4, 2021.

Kass, R. E., B. P. Carlin, A. Gelman, and R. M. Neal. 1998. "Markov chain Monte Carlo in practice: a roundtable discussion". *The American Statistician* 52(2):93–100.

Liew, S. X., M. Afrasiabi, and J. L. Austerweil. 2019. "An introduction to data analysis using the PyMC3 probabilistic programming framework: A case study with Gaussian Mixture Modeling".

Maxwell, P., E. Alhajjar, and N. D. Bastian. 2019. "Intelligent Feature Engineering for Cybersecurity". In *2019 IEEE International Conference on Big Data (Big Data)*, 5005–5011. IEEE.

Mirza, M., and S. Osindero. 2014. "Conditional Generative Adversarial Nets". *CoRR* abs/1411.1784.

Nelsen, R. B. 2007. *An introduction to copulas*. Springer Science & Business Media.

Neumann, J., A. W. Burks et al. 1966. *Theory of self-reproducing automata*, Volume 1102024. University of Illinois press Urbana.

Patcha, A., and J.-M. Park. 2007. "An overview of anomaly detection techniques: Existing solutions and latest technological trends". *Computer networks* 51(12):3448–3470.

Patki, N., R. Wedge, and K. Veeramachaneni. 2016. "The Synthetic Data Vault". In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 399–410.

Reuther, A., J. Kepner, C. Byun, S. Samsi, W. Arcand, D. Bestor, B. Bergeron, V. Gadepally, M. Houle, M. Hubbell et al. 2018. "Interactive supercomputing on 40,000 cores for machine learning and data analysis". In *2018 IEEE High Performance extreme Computing Conference (HPEC)*, 1–6. IEEE.

Scott, S. L. 2001. "Detecting network intrusion using a markov modulated nonhomogeneous poisson process". *Submitted to the Journal of the American Statistical Association* 31:80.

Sharafaldin, I., A. Gharib, A. H. Lashkari, and A. A. Ghorbani. 2018. "Towards a reliable intrusion detection benchmark dataset". *Software Networking* 2018(1):177–200.

Stallings, W., L. Brown, M. D. Bauer, and A. K. Bhattacharjee. 2012. *Computer security: principles and practice*. Pearson Education Upper Saddle River, NJ, USA.

Tavallaee, M., E. Bagheri, W. Lu, and A. A. Ghorbani. 2009. "A detailed analysis of the KDD CUP 99 data set". In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 1–6.

Teh, Yee Whye 2010. "Dirichlet Process.".

Viegas, E. K., A. O. Santin, and L. S. Oliveira. 2017. "Toward a reliable anomaly-based intrusion detection in real-world environments". *Computer Networks* 127:200–216.

Dustin Volz and Robert McMillan 2020. "Hack Suggests New Scope, Sophistication for Cyberattacks". https://www.wsj.com/articles/hack-suggests-new-scope-sophistication-for-cyberattacks-11608251360, accessed March 4, 2021.

Xu, L., M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni. 2019. "Modeling Tabular Data using Conditional GAN". *33rd Conference on Neual Information Processing Systems*.

Yao, S., Y. Zhao, H. Shao, C. Zhang, A. Zhang, S. Hu, D. Liu, S. Liu, L. Su, and T. Abdelzaher. 2018, September. "SenseGAN: Enabling Deep Learning for Internet of Things with a Semi-Supervised Framework". *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2(3).

Yildirim, I. 2012. "Bayesian inference: Metropolis-hastings sampling". *Dept. of Brain and Cognitive Sciences, Univ. of Rochester, Rochester, NY*.

AUTHOR BIOGRAPHIES

MARC CHALÉ is a graduate student in the Department of Operational Sciences at the Air Force Institute of Technology (AFIT) pursuing a Ph.D. degree in Operations Research with a doctoral minor in Statistics. Prior to his doctoral studies at AFIT, he also earned M.S. degrees in both Industrial Engineering and Operations Research. His recent research has involved the use of discriminative and generative machine learning and Bayesian methods to improve cybersecurity measures such as network intrusion detection systems, as well as the development and use of meta-learning techniques for the problem of machine learning algorithm selection. His email address is marc.chale@afit.edu. His website is http://www.marcchale.com/resume.html.

NATHANIEL BASTIAN is Chief Data Scientist and Senior Research Scientist at the Army Cyber Institute (ACI) at West Point, also serving as Assistant Professor of Operations Research and Data Science at the United States Military Academy (USMA). He leads the ACI's Data Science Research Team and oversees the ACI's Intelligent Cyber-Systems and Analytics Research Laboratory, researching, building, assessing and deploying computationally intelligent, assured (secure, resilient, robust, safe, trusted) and distributed decision-support models, tools and systems for autonomous cyber operations in highly-contested, complex battlefield environments. He holds a Ph.D. degree in Industrial Engineering and Operations Research from the Pennsylvania State University, a M.Eng. degree in Industrial Engineering from Penn State, a M.S. degree in Econometrics and Operations Research from Maastricht University, and a B.S. degree in Engineering Management (Electrical Engineering) with Honors from USMA. His email address is nathaniel.bastian@westpoint.edu. His website is https://cyber.army.mil/Research/Research-Team/Bastian/.