Improving Managed Network Services Using Cooperative Synthetic Data Augmentation

Submitted in partial fulfillment of the requirements for

the degree of

Master of Science

in

Information Networking

Minhao Jin

B.S., Electrical and Computer Engineering, Shanghai Jiao Tong University

Carnegie Mellon University
Pittsburgh, PA

December, 2022

Acknowledgements

I thank Prof. Vyas Sekar, Prof. Giulia Fanti, Yucheng Yin and Zinan Lin for their insightful suggestions on this project. I thank Rahul Anand Sharma for his invaluable support with regard to fingerprinting in this paper. I would like to express my gratitude to Yucheng Yin and Zinan Lin for helping me become more familiar with the concepts related to machine learning, networking, security, and privacy. This work was supported in part by the National Science Foundation through Convergence Accelerator grant CA-2040675 and RINGS grant 2148359. I also acknowledge the generous support of the Sloan Foundation, Intel, Siemens, Bosch, J.P. Morgan Chase, and Cisco.

Abstract

Many managed service vendors in networking are adopting machine learning (ML) for many applications for their customers; e.g., anomaly detection, device fingerprinting, and resource management. Today, the data for training is siloed across customers leading to sub-optimal performance. While there are emerging proposals (e.g., federated learning, multi-party computation) to enable cooperative learning, these are at odds with analysts need for data for model exploration and testing. In this thesis, we envision a novel use of synthetic data generated using Generative Adversarial Networks (GANs) to augment the performance of existing ML workflows. We formulate the cooperative data augmentation problem, identify the design space of options, and identify key research challenges. We demonstrate the preliminary promise under two settings: (1) traffic classification and (2) novelty detection showing that our improved workflow can enhance the performance of ML models up to 58% in AUC score. We also identify limitations and discuss for future work.

Table of Contents

List of Figures

1

Introduction

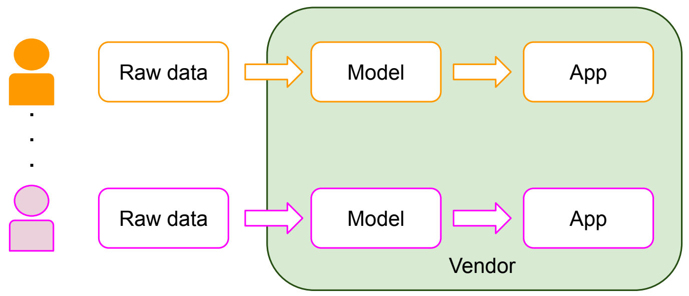
Many vendors offering network management or network monitoring as a service are beginning to offer ML-driven capabilities to improve the performance, reliability, and security of their customers [60, 10]. For instance, recent efforts have demonstrated the promise of ML-driven approaches for user experience prediction [32, 31], novelty detection [73], traffic fingerprinting [52, 57, 13], among others. There are even grander aspirations for "self driving networks" that use AI/ML driven network management [19]

The typical workflow is that a service vendor S receives data feeds from individual customers $C_1, ...C_N$ for various management tasks of interest and develops custom ML-driven workflows for *each customer in isolation.* The data-driven workflows for different customers are *siloed* due to internal (to S) and contractual (from the customers) policy concerns. Unfortunately, the value of the ML-driven workflows is only as good as the data that feeds into it. As such, silos result in suboptimal performance and significant blind spots with respect to emerging patterns. For example, LastLine estimates that on average, 50% of alerts in endpoint protection systems are false positives [35]. Indeed, recent efforts have explicitly called into question the robustness of ML-derived algorithms when data is limited [72].

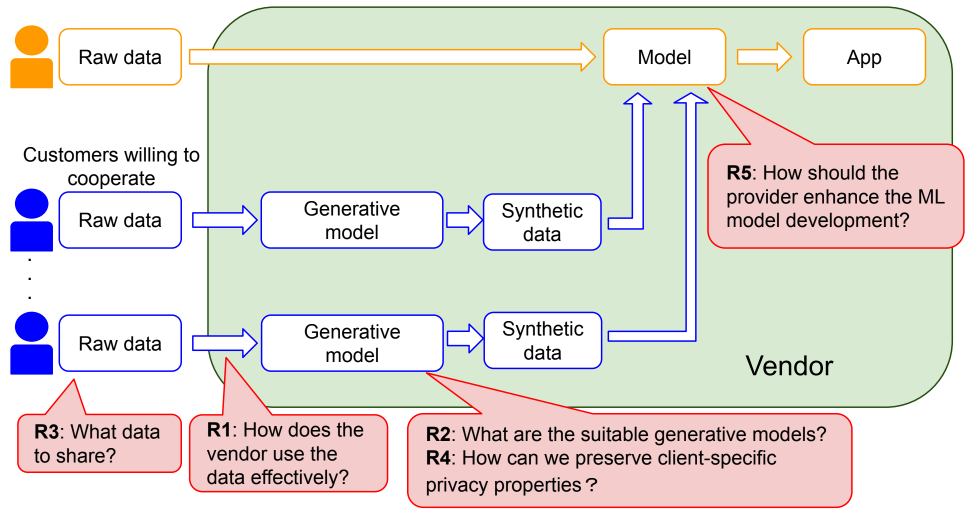
To tackle the pitfalls of data silos, concurrent efforts in the ML, systems, and privacy literature have explored solutions based on techniques like federated learning [34, 11, 33] and machine learning over encrypted data [70]. While these are valuable, they are not always appropriate for data scientists developing ML-for-networking applications. They typically assume that the ML algorithm, features, and model requirements are known *a priori*. As such, their focus is on privacy-preserving computation and communication frameworks for model learning. Unfortunately, the reality of ML-driven workflows is much messier, and data scientists and analysts spend significant effort on *model and data exploration* to clean datasets, derive the features of interest, and explore algorithmic development [42]—tasks that would be difficult or impossible using these alternatives [8]. Furthermore, such "blackbox" approaches complicate efforts to provide model interpretability and explainability.

In this thesis, we explore a pragmatic alternative for tackling the data silo problem using *synthetic data from deep generative models*—that is, data generated with the same statistical properties as real data that can be safely released to enable cross-stakeholder cooperation. While synthetic data is not a new idea in networking [64, 63], prior work has relied on custom simulators (e.g., [64, 63, 30, 50, 16, 43, 62]) or expert-driven models (e.g., [15, 66]). Unfortunately, these do not generalize across datasets and customer-specific scenarios. Fortunately, recent efforts have shown the feasibility of using deep generative models, of which Generative Adversarial Networks (GANs) are a well-known example, to generate synthetic network datasets [71, 40, 9]. The primary benefit deep generative models offer is the ability to learn high fidelity representations of high-dimensional relationships. A secondary benefit is *flexibility* to tune generation (e.g., augment anomalous or sparse events), which would not be possible with raw or anonymized datasets [49, 41].

The contribution in this thesis is in *formulating* the problem of generative-model-



(a) Status quo: Each customer's model can only use the local dataset collected from that customer


Raw data
Model
App
Customers willing to cooperate
Raw data
Generative model
Synthetic data
Raw data
Generative model
Synthetic data
Vendor
R5: How should the provider enhance the ML model development?
R3: What data to share?
R1: How does the vendor use the data effectively?
R2: What are the suitable generative models?
R4: How can we preserve client-specific privacy properties?

(b) Our vision for augmenting ML workflows with synthetic data and the technical questions this vision raises.

Figure 1.1: Contrasting our vision with the status quo of ML workflows for managed services

augmented ML workflows for managed network services use cases and identifying *research challenges* that arise therein. We identify the design space of data augmentation strategies. For example,

1. Do we use normal or anomalous data from customers?
2. Do we generate and utilize synthetic normal or anomalous data for augmentation?

We also explore the choice of generative models to generate the synthetic data across these design space alternatives; e.g., generating anomalous data is fundamentally harder than generating the common case patterns. Figure 1.1b shows the general structure of the workflow and would be explained in details in Chapter 3

- We present a preliminary proof of concept framework and empirically demonstrate the value of our framework using real-world datasets.
- We simulate a data-hungry and type-imbalanced learning pipeline, and model both homogeneous and heterogeneous customer data distributions to present the benefits of this framework.
- We observe that a given customer can obtain up to 36% improvement in model performance (AUC score) by receiving augmented synthetic data from other customers in the homogeneous setting, and up to 58% improvement in the heterogeneous setting.
- We also identify a range of limitations and open questions for future work.

2

Motivation

We start by illustrating some exemplar use cases of ML-for-networking, the value of larger datasets in training these models, and argue why current efforts are qualitatively at odds with operational workflows.

2.1 Motivating use cases

A growing body of work shows that future network systems will require novel *data-driven* techniques to meet their security, adaptability, robustness, and autonomy goals (e.g., [19, 32, 28]). In parallel, small- to medium-sized enterprises are increasingly outsourcing their requirements on those novel data-driven applications to third-party network service vendors (e.g., Cisco, Palo Alto Network)[60, 10]. Those vendors have the expertise and help its customers to develop the applications based on their own needs.

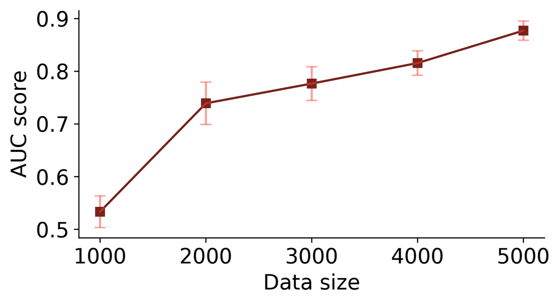
Figure 1.1a shows the structure of the third-party service workflow between the vendors and customers. The vendor makes use of the required customer's data to help build various kinds of data-driven applications based on its customized requirements. We highlight three example applications: traffic classification, novelty detection and fingerprinting.

U1: Traffic classification. Traffic classification plays a significant role in network security and management. The network service vendor analyzes customer's network data to classify the anomalous or attack traffic and to investigate events (e.g., [21, 6, 77, 7]). For example, traffic type prediction is one of the vital tasks for the vendor to maintain network security such as helping to implement intrusion detection system for the customers.

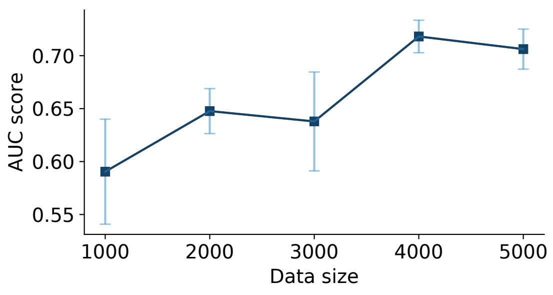
U2: Novelty detection. Novelty detection is to determine if the incoming unseen event is unusual or anomalous. It has been widely implemented in many different use cases. For example, with the rapid development of Internet of Things (IoT), many interconnected IoT devices are deployed to monitor and detect new, unusual, and anomalous data [73]. Determining if the new unseen data is unusual and anomalous can be indicative on security incidents or device malfunctions. Vendors could train a novelty detection model that determines if the data representation from an incoming traffic is anomalous relative to the majority of the traffic if required from *all* of its customers.

U3: Fingerprinting. Fingerprinting can be used to identify compromised devices [52, 57, 13]. It consists of detecting patterns and observing differences in the network packets, physical behavior etc. generated by a compromised versus a regular device. This allows a vendor to detect and limit the harm caused by a compromised device. For e.g., it can be used to identify and quarantine an IoT device that is running a port scan request after it has been compromised.

In most cases, service vendors are only allowed to build separate application models based on each customer's data individually due to policy and privacy concerns. Because of the limited data size, most customers can only achieve suboptimal performance. Intuitively, data- and ML-driven applications gain benefits when having more data access. As an illustrative example, Figure 2.1 shows the performance of traffic



(a) Traffic classification.



(b) Novelty detection.

Figure 2.1: Data-driven applications such as traffic classification and novelty detection model can benefit with more data. Detailed explanation of dataset, models, performance metrics is in Chapter 4.2.

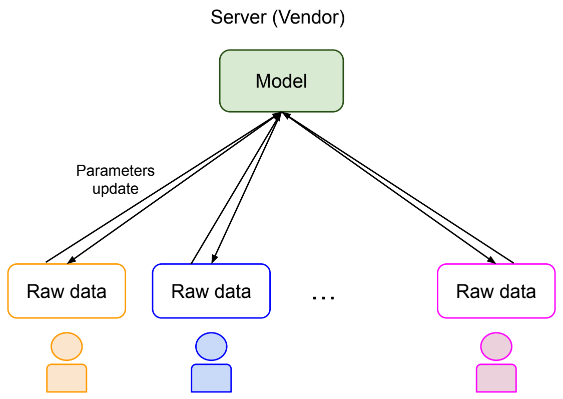


Figure 2.2: Structure for the workflow of Federated Learning.

classification and novelty detection tasks [12] using public datasets [51] and models as a function of the data size used for training.[1] As we can see, the performance improves significantly with more data.

2.2 Alternative approaches

Today, there are two leading research proposals for tackling data silos. Federated Learning (FL) learns a ML model at the vendor from private customer data [34]. Figure 2.2 shows the general structure of the workflow. The server (vendor) would use

[1] Details of the dataset are in §4.2.

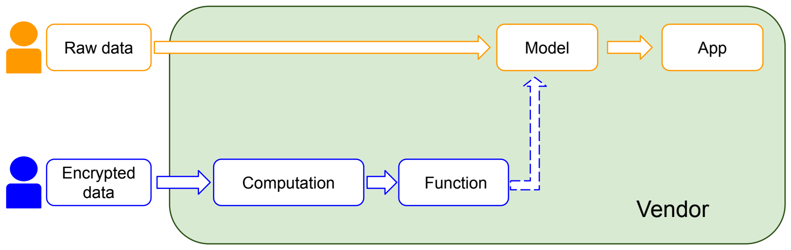


Figure 2.3: The vendor is using the encrypted data owned by the blue customer to help the yellow one.

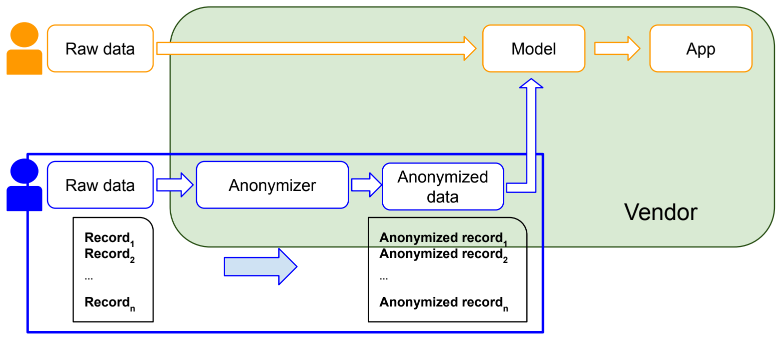


Figure 2.4: The blue customer is willing to share its anonymized data to the yellow customer to improve the yellow one's model performance.

all the customers' data to train a global model without touching them. Customer privacy is preserved through differentially-private model updates and/or cryptographic aggregation techniques [33]. As shown in Figure 2.3, the second approach relies on computation over encrypted data, such as homomorphic encryption or secure multi-party computation [18, 70, 68, 58]. These approaches allow a vendor to compute a function of the data without plaintext access.

Both approaches are powerful and seeing real-world adoption [33, 1, 3]. By design, they hide raw data from the learning party, i.e., the vendor. In reality, however, up to 80% of data analysts' time can be spent transforming data into a usable form [42]. Cleaning data and extracting features is typically the bulk of work for data scientists. As such, technologies that hide raw data from the data scientists make key data science tasks difficult, if not impossible [8, 54].

An alternative common practice is to share anonymized data, as demonstrated in Figure 2.4 [59, 56]. Anonymized traces preserve a one-to-one mapping between each packet or flow in the original and shared datasets. While anonymization is appealing at first glance, prior efforts have shown “linkage” attacks against anonymization (e.g., [53]). Second, the one-to-one mapping restricts the flexibility to produce new or more samples to meet the data analyst’s needs, e.g., adding more attack samples or balancing datasets across classes.

Synthetic data can potentially help with all of these challenges; it provides data in the same schema (and ideally, from the same distribution) as the original data. Indeed, recent efforts have designed deep-learning-based synthetic data models for network data [71, 40, 9]. However, little work has gone towards exploring the efficacy of synthetic data for cooperative learning pipelines in the networking domain. This is the focus of our work.

3

Overview

In this chapter, we provide a high-level view of our vision and contrast it to the status quo. We also set up a broad set of algorithmic and system design challenges that we need to address as part of this vision.

Figure 1.1a shows a simplified view of the current workflow that the service vendor uses for their customers. The vendor uses the customer's raw data to build the model for the data-driven application. As discussed, it is hard to build a robust ML model because the data from that customer may be sparse. Such sparsity can both result in two-sided types of errors in typical applications; e.g., for anomaly detection poor coverage over normal patterns can result in false positives (i.e., normal patterns not seen before get flagged erroneously) and poor coverage over attack/anomaly scenarios can result in false negatives (i.e., anomalies are missed by the inference model).

To address this limitation, we envision synthetic data augmentation as a pragmatic alternative to improve the models while still addressing the privacy concerns of the stakeholders. As such, we envision a new cooperative framework to enhance applications as shown in Figure 1.1b . We envision one or more customers of the service vendors *opting in* to allow the the service vendor to use synthetic data for cooperatively boosting the performance of the collective as well as their own performance.

For each customer that has opted in, we envision generating high-fidelity, privacy preserving *synthetic* data to be shared with the vendor. Depending on the deployment and policy constraints, the generation may either run "on premise" at the customer side or may be run within the vendor's infrastructure. The service vendor can use the augmented dataset to drive the target ML applications of interest.

Given this workflow, several natural questions arise:

- **R1: How does the vendor use the data effectively?** There are 2 subquestions: (1) what data do we want the generative model to generate for augmentation, and (2) what data do we want to train the generative model with. Regarding (1): in some cases, generating only a subset of the traffic types may be sufficient. For example, when a target customer requires the vendor to build a traffic type classification application, augmenting the anomalous traffic can dramatically help the classifier reduce the false positive and false negative rate. It may not be necessary for the vendor to generate synthetic data from all the data provided by the customers. Regarding (2): even if we only want to augment one subset of the traffic types (e.g., anomalous traffic), training the generative model using all available traffic (e.g., both normal and anomalous traffic) may be beneficial depending on the type of generative models.
- **R2: What are suitable generative models?** The quality and fidelity of synthetic data generated from each customer is vital to the performance of the final augmented model. However, networking data is usually high-dimensional timeseries with complex patterns. What's more, rare events (e.g., anomalies) makes synthetic data generation more challenging as it will be hard to generate new, representative examples given limited training samples for rare classes.

- **R3: What data to share?** We envision for many typical applications that customers will have both *normal* and *anomalous* traffic patterns of interest. Each customer can opt in to provide (with suitable privacy policies discussed below) data on either normal or anomalous patterns or both, depending on policy considerations.
- **R4: How can we preserve customer privacy?** In addition to contractual arrangements, privacy guarantees are also critical to allow the customers to share their networking data. The vendor must be able to generate the privacy-preserving synthetic data for augmentation. Currently, there is still no unified privacy metrics especially for networking data. Differential privacy (DP) [17] is one of the leading metrics but prior works [40] have shown that naive DP could destroy the fidelity of synthetic data even under very weak privacy guarantees.
- **R5: How should the vendor enhance the ML model development?** As a starting point, we can envision the service vendor simply using the existing ML models with augmented training datasets. However, it may also be worthwhile for them to revisit the model in light of the larger dataset. For instance, we may need to consider changing the structure or the hyperparameters of the model in order to reach the optimal performance given a larger training set. Looking even further, the vendor may even be able to consider more expressive complex models (e.g., foundation models or transformers) given the new larger dataset at its disposal that may have been previously infeasible.

Our goal in this thesis is to articulate these new design questions and challenges that arise in this novel workflow rather than seek conclusive answers to these research questions. As such, our preliminary exploration in the next Chapter presents an initial study to R1 and R2 to demonstrate the potential utility of the framework and lays out the design space for R3, R5. We defer a discussion of R4 and questions related to generality to Chapter 6.

4

Formulation and Initial Study

In this chapter, we formulate and scope our problem in a data hungry and traffic type unbalanced assumption. We describe our initial ideas on how to determine some of the design choices described in §3.

4.1 Problem Formulation

A vendor has N customers, $\{C_1, \ldots, C_N\}$. Each customer has a local traffic dataset D_i. In this thesis, we will study downstream use cases that involve detecting anomalous or malicious traffic, so we further specify that $D_i = (N_i, A_i)$, where N_i represents the set of normal traffic records and A_i the set of anomalous traffic records. Note that the customers may or may not know the labels (i.e., normal v.s. anomalous) of the traffic depending on setting.

The vendor is trying to learn a ML model f_i for each of its customers to accomplish a downstream task. The downstream task is assumed to be the same for all customers, but the models f_i may have different parameters, hyperparameters, or architectures, due to different local training data distributions. When data is siloed, each f_i is learned using only the corresponding local dataset D_i (Figure Figure 1.1a).

As discussed in Chapter 3, we explore a setting where the vendor can *augment* the training data for each model. We use $\tilde{D}_j = (\tilde{N}_j, \tilde{A}_j)$ to denote a synthetic version of customer j's data, with synthetic normal data $\tilde{N}_j$ and synthetic anomalous data $\tilde{A}_j$. Hence, the vendor can augment customer i's real data D_i with (any subset of) synthetic data $\tilde{D}_j$ for $j \neq i$.

We focus on two research questions as a starting point:

1. (R1) How does the vendor use the data effectively?
2. (R2) What are the suitable generative models?

As a first step, we assume that customers willing to cooperate share all the data with the vendor (R3) and we leave the choices of privacy metrics and model re-weighting/update as an open question in the future directions (R4, R5).

R1: How does the vendor use the data effectively? In this work, we explore this design based on the downstream applications. For unsupervised ML applications like novelty detection, the vendor does not know the ground truth labels for the shared data. Therefore, we use all the data to build a generative model, and use the generative model to generate all types of traffic. For supervised ML applications like traffic classification, vendor knows the ground truth labels. Regarding the output of the generative model, we only generate anomalous traffic from the generative model due to its significant benefit. Regarding the input for training the generative model, we explore two choices: utilizing only anomalous traffic, and utilizing both normal and anomalous traffic. We leave the more comprehensive exploration to future work.

R2: What are the suitable generative models? Classical synthetic data models have been based on strong assumptions about the underlying data, including (cyclo)stationarity [46, 47, 45, 48, 65, 55, 5, 38], autoregressive generation [4, 76, 75, 67], or Markovian generation [27, 22, 39]. These assumptions may not hold for many datasets [65, 55, 5, 38, 4, 76, 75, 67]. As such, these approaches are outperformed by modern neural network models [29, 36], such as GANs [23].

GANs generally consist of two neural networks, one is called generator and the other is called discriminator. The job of the generator is to generate the fake samples while discriminator should be able to determine if the given sample is fake (generated from the generator) or is real (sampled from the training set). The two networks contests each other to get the final state. That is the generator can generate fake samples that looks very real and the discriminator can only achieves the 50% accuracy. In short, GANs take as input a set of training data samples, and output a model that can produce new samples from the same distribution as the original data. GANs are popular in part because they require few assumptions on the class of distributions from which the training data is drawn. They have emerged as a popular technique for generating or augmenting datasets, e.g., medical images or patient records [24, 14, 20, 61, 25], and more recently, for generating synthetic models of networking datasets [71, 40, 9].

Network traffic can be divided into classes (e.g., normal vs. anomalous). Naively, one could train a separate generative model for each data class. In practice, this approach has poor fidelity, particularly when some of the classes have little training data (e.g., anomalous traffic) [41].

A more robust approach in the generative modeling literature is to *cogenerate* traffic from different classes by training a single model capable of conditional generation. Common GAN-based methods for cogenerating samples from different classes include conditional GANs (CGAN) [49]. CGAN would accept an additional input specifying the label of the fake samples from the generator. In these models, the GAN can be made to output samples from a specific class by inputting the class label to the generator; this can be used to tune the ratio of synthetic normal vs. anomalous traffic, for instance. Although these models still struggle to learn from imbalanced training data, they can use data from other classes to learn general patterns that apply to rare classes. For example, a conditional GAN can learn what kinds of correlations appear between fields in normal NetFlow records, which may be partially relevant for generating realistic anomalous records.

As preliminary work, we explore different designs based on the downstream application. For example, for traffic classification — which requires training labels — we explore both design options mentioned above: using a GAN and cogeneration using a CGAN. For novelty detection—which is an unsupervised task—there are no labels in the training data, and we use a vanilla GAN to generate unlabeled synthetic data. We leave a comprehensive exploration for future work.

4.2 Preliminary Evaluation

We evaluate two tasks, traffic type classification and novelty detection under different setups.

4.2.1 Experimental Setup

We run experiments on one of the most popular data types in the networking domain, NetFlow [2]. We use the TON_IoT dataset [51], which is a telemetry IoT sensor dataset for evaluating the fidelity and efficiency of different cybersecurity applications. The original dataset contains 300k normal samples and nine different attacks with each attack about 20k anomalous samples. We set the total number of normal samples $N = \sum_1^n N_i = 20,000$ and the total number of anomalous samples as $A = \sum_1^n A_i = 2,000$. We set the total number of customers to $n = 10$ and evaluate both a homogeneous setup, where customers have data from the same distribution, and a heterogeneous setup, which models different customers having data from different distributions.

Homogeneous Setup. For the homogeneous setup, we randomly partition the entire dataset across the n customers, subject to $N_i/A_i = 10$ for $i = 1, 2, \cdots, n$.

To evaluate our pipeline, we used an 80%:20% i.i.d. train/test split for each customer's data. The held-out test set is used to evaluate the model before and after augmentation.

Heterogeneous Setup. To simulate different customers, we partitioned the entire sample into 11 evenly-sized chunks by time. Each of the 10 customers owns one chunk, and the last chunk is used as the test set. Since the data distribution changes over the course of the dataset, this partitioning causes different customers to have different data distributions, which models customer heterogeneity. It is possible that some customers may have only normal or only anomalous traffic. Moreover, by testing on a held-out time chunk, we model distribution drift—i.e., the model is trained over data from a different distribution than the final test data. In our experiments, we can only measure model performance for customers with both normal and anomalous traffic in their local test dataset.

4.2.2 Use cases

U1: Traffic classification. In our first use case, we predict the type of a given NetFlow record (normal/anomalous) given fields {port number, protocol, bytes/flow, packets/flow, flow duration}. We don't include the IP as a field for the task since the synthetic data generation model will generate new and unseen IP from the raw dataset to help with generalization. However, these newly generated IP cannot be considerred as a clue in some specific use cases such as traffic classification task.

We trained a MLP model as a traffic classifier which will classify the given netflow record as normal or anomalous. MLP is one of the most widely used fully connected neural networks which plays an important role in various use cases. Its performance is evaluated using AUC score. AUC score makes use of True Positive rate and False Positive rate under different cut-off thresholds to represent the degree of separability. Compared to other metrics such as accuracy, AUC can be a better indicator and performs well in the imbalanced dataset, which is appropriate given the imbalanced nature of our data (normal vs. anomalous). Since the different parameters of the MLP classifier has influence on the final performance, we further homogeneously split the training set into training data and validation data, the latter of which is used to select the best set of hyperparameters. The final classifier is selected as the one with the highest AUC score over the validation set.

As discussed in Chapter 4.1, we augment only anomalous traffic as a first step. That is, in Figure 1.1b, to train model f_i for the orange customer C_i, the blue customers C_j, $j \neq i$ contribute only synthetic anomalous data $\tilde{A}_j$. To generate this synthetic data, we evaluate two deep generative models, GAN and CGAN. Since we only augment anomalous traffic in this thesis, we sampled the anomalous traffic from CGAN, and only trained GAN to generate anomalous traffic. Even though we only generate anomalous traffic, CGAN can benefit from normal data to learn a better model.

U2: Novelty detection. Our second use case is unsupervised novelty detection. We explore a model called Local Outlier Factor (LOF) [12]. LOF is a unsupervised anomaly detection or novelty detection method. Local density deviation will be computed for a given data point with respect to its neighbors and those with very low density deviation will be regarded as outliers. We use the same fields as in U1, and ignore labels in the training data. All the data from each customer is used to train a GAN without regard to whether the data is normal or anomalous. That is, to train novelty detection model f_i for customer C_i, the other customers C_j contribute unlabeled synthetic data $\tilde{D}_j$. Finally, the test set is passed through the learned model f_i. Given a netflow trace, f_i would determine if this incoming trace is of majority or is an outlier. We regard the majority as normal and an outlier as anomalous traffic. We evaluate the learned model by computing the AUC score over the ground truth labels in the test set.

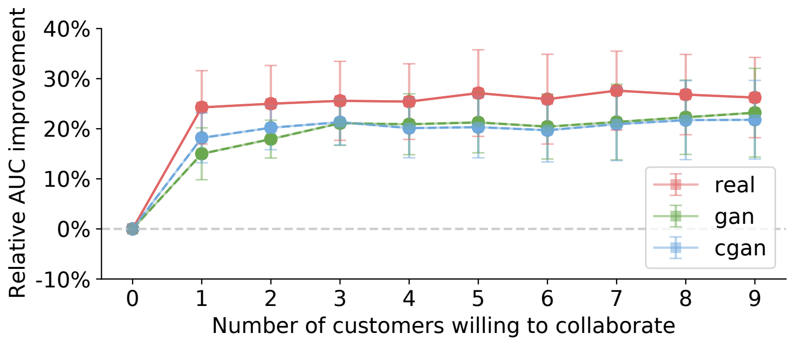


Figure 4.1: Traffic classification with different customers opt-in under homogeneous setup.

4.2.3 Results

We first show results for traffic type classification task under the homogeneous setup.

Finding 1: ***The vendor can build a robust traffic classification model when there are more cooperative customers.***

We consider a setting where the vendor augments a single customer's real data (real and anomalous) with synthetic anomalous traffic from different customers; at most, we can augment with up to 9 customers' anomalous data. If a customer decides to participate, it will contribute 160 synthetic samples. Figure 4.1 shows the change in AUC as we increase the number of customers who contribute synthetic anomalous data. We first observe that on this dataset, synthetic data augmentation achieves close AUC benefits as augmenting with real data, which is an upper bound on the performance improvements we can hope for. Second, having more customers participated in data sharing is helpful in getting better model performance.

Finding 2: ***The traffic classification model is relatively robust to the*** **amount** ***of augmented anomalous synthetic data.***

We next assume that for any target customer, the remaining 9 customers are all willing to share data. The vendor augments each customer's model with the same amount of synthetic anomalous data from all of the cooperative customers (up to 160 samples per customer). We define the augmentation ratio R_i for customer i as the ratio between the amount of augmented anomalous synthetic data and the original size of the dataset $|D_i| = |N_i| + |A_i|$, i.e $R_i = \sum_{j \neq i} |\tilde{A}_j|/|D_i|$. Because we are only augmenting with (up to) 1440 samples of anomalous data ($R_i \approx 81\%$), this ratio will be less than 1 in our experiments. Figure 4.2 shows the change in AUC as the augmentation ratio grows. We find that AUC is relatively robust to the ratio when it gets 30%, which suggests that the improvement may reach a state of "saturation" when amount of data augmented is beyond a "threshold".

Finding 3: ***Novelty detection also benefits from (sufficient amounts of) augmented data.***

This experiment evaluates the experimental setting from Figure 4.2, except on the novelty detection task. Here, we assume that all customers share the data and the vendor augments each customer's model with the same amount of synthetic data from

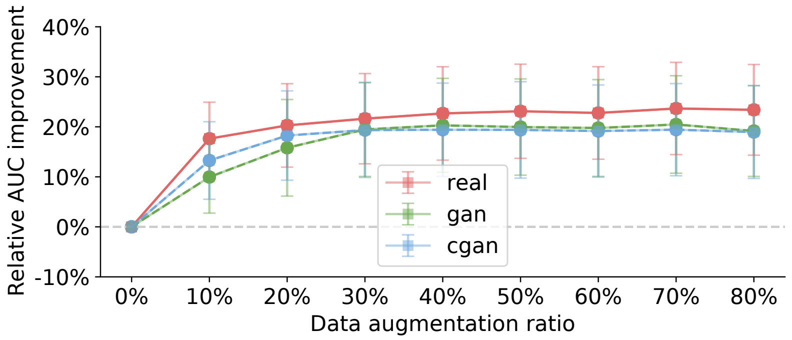


Figure 4.2: Traffic classification with different augmentation ratio under homogeneous setup.

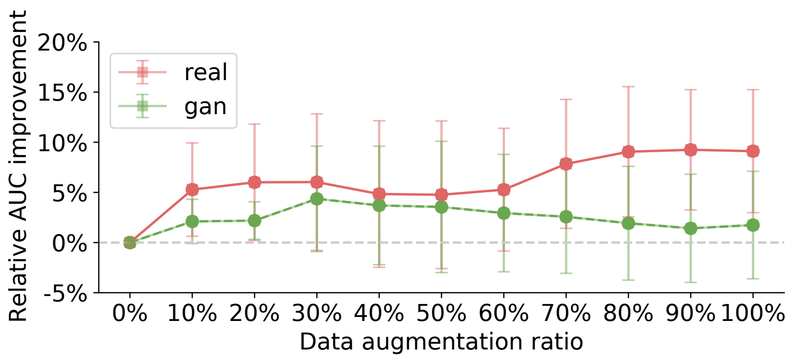


Figure 4.3: Novelty detection with different augmentation ratio under homogeneous setup.

all customers, but we vary the *number* of shared synthetic samples per customer. We capture the total amount of shared data by the augmentation ratio R_i for customer i, defined as the ratio between the amount of added data and the original data quantity $|D_i| = |N_i| + |A_i|$, i.e $R_i = \sum_{j \neq i} |\tilde{D}_j| / |D_i|$. Figure 4.3 shows the improvement in AUC as the augmentation ratio grows. With sufficient augmented data, we still see the relative improvement for both synthetic and real data augmentation. We also observe that sometimes synthetic data augmentation hurts the model. One possible reason is that the GAN cannot mimic the distribution of both normal and anomalous traffic well.

Finding 4: ***In heterogeneous data, different customers benefit differently from data augmentation.***

Figure 4.4 and Figure 4.5 show the AUC of traffic classification and novelty detection models before and after data augmentation in the heterogeneous setting. Each column shows the changes for a single customer, when their data is augmented with data from all other customers. For this experiment, all customers benefit from data augmentation. More importantly, different customers benefit by different amounts. We expect that in general, data augmentation can help or hurt heterogeneous cus-

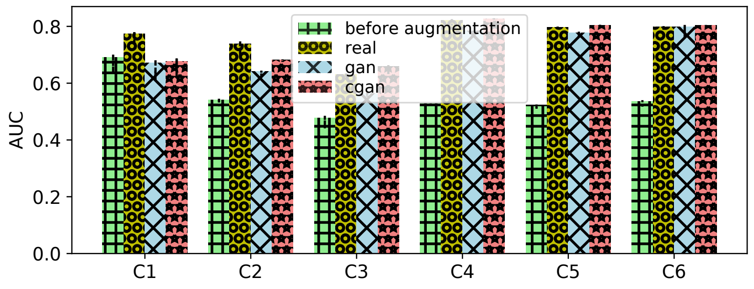


Figure 4.4: Traffic classification in heterogeneous setup.

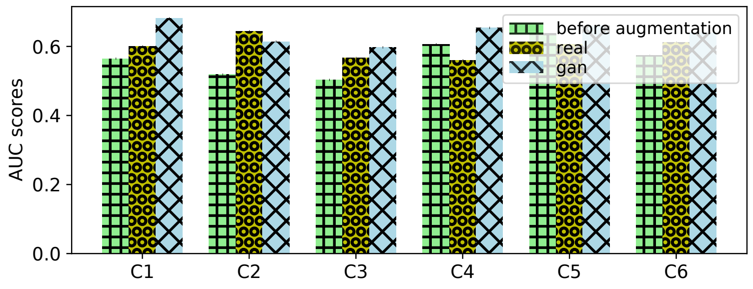


Figure 4.5: Novelty detection in heterogeneous setup.

tomers depending on their local data distribution. Understanding and predicting such trends is an interesting direction for future work.

5

Related Works

Data silos resulting in suboptimal performance is a long-term pain points which has a long history for exploration. We would list several related works in this Chapter.

Federated learning. In contrast to traditional machine learning algorithms, federated learning allows for the training of a ML model across decentralized clients. Instead of requiring each client to upload their data, they only need to compute an update using their local data to the global model maintained by the server [44]. Federated learning has definitely the potential to train a robust model with privacy guarantees by using all the clients' data without touching them. However, networking data is usually high-dimensional time series data which requires complicated preprocessing logic. For example, the state-of-the-art synthetic IP header trace generator, NetShare, uses a method called ip2vec to encode the field {port, protocol}, bit-encoding to preprocess the field {IP}, zero-one normalization on other continuous fields and one-hot encoding to other categorical fields [74]. This can be time-consuming for data scientists, and in federated learning, the server is unable to access the raw data, making the preprocessing task even more difficult in practice.

Data augmentation. So far, there are two main different ways of data augmentation: (1) synthetic data augmentation and (2) anonymized data augmentation. Regarding (1): a generative model will be utilized for synthetic trace generation. These synthetic traces would be used for data augmentation to improve the performance of the model. Strictly speaking, our cooperative data augmentation workflow also falls in this category. But, to the best of our knowledge, all the current works fail to consider suitable augmentation strategy and privacy guarantees as important factors during augmentation. [69] and [26] are two examples using GAN based generative model to augment networking traces in different use cases. However, they didn't include a systematic strategy on what data and how to augment. Meanwhile, they didn't formulate the problems in a vendor-customer scenario and of course didn't consider the appropriate privacy metrics as a necessary open question for exploration in their work. Regarding (2): anonymization policy plays an important role to ensure the security and privacy needs. However, the anonymized data processed based on anonymization policy may also have some shortcomings when used for augmentation. The first one is about the generalization. As briefly mentioned in Chapter 2.2, the anonymization would preserve a one-to-one mapping between the original and the anonymized dataset. Therefore, the lack of generalization and inflexibility for the amount of data may have a negative effect on the data augmentation and affect the final performance of the augmented data. The second one is the lost of some significant fields. For example, the anonymization policy in [56] would enforce the anonymized data to lose the exact timestamp and be replaced by a counter instead. The lost of timestamp would destroy the temporal pattern which is a vital property for the networking traces. What's more, this would also cause the data schema of the raw data and data for augmentation different, which would increase the difficulty on augmentation.

6

Limitations and Discussions

In this chapter, we revisit several limitations in this work and discuss the related open questions for future work:

Performance comparison with the alternatives: In this study, we did not compare our workflow with other approaches such as federated learning or anonymized data sharing. While these alternatives have their own limitations, they have the potential to improve customer data-driven models. Therefore, in future work, it would be worth evaluating these methods and see how our synthetic data augmentation workflow is compared to them.

Privacy metrics: The focus in this thesis was on establishing the design space and potential utility of synthetic data augmentation. To show the initial promise, we haven't included the privacy metrics in this work. We posit that in networking settings privacy notions explored in other communities such as differential privacy may not suffice; e.g., business or competitive concerns may matter more. This raises several question: What are the right privacy metrics for customers to specify such business requirements? Are there formal foundations for defining these domain-specific privacy metrics? Can we reason about the strength of various approaches (e.g., anonymization vs. synthetic data) in meeting these goals?

More realistic experiments setup: The dataset used in this experiment is a telemetry IoT sensor dataset. Partitioning only one dataset to different customers to simulate data distribution for each one may not reflect a real scenario. A better solution would be to assume that the data owned by each customer comes from different sources. Therefore, in a future work, a potential direction is to use synthetic data trained on completely different raw datasets for augmentation and evaluate the performance improvement of the augmented model.

Data heterogeneity: In my thesis, I simulated a scenario of data heterogeneity by splitting the data by time and assigning it to different customers. However, this does not provide a quantitative evaluation of the heterogeneity between each customer's data. However, the extent of heterogeneity may impact the augmentation strategy, the final performance of the data-driven model and also the required amounted of synthetic data required for the model to reach it. For a labeled dataset, a potential more systematic way is to calculate diversity of the labels within each customer's data and the heterogeneity of the data with the same label between different customers. A larger label heterogeneity indicates that each customer has quite different labels of data, which can directly help model to reach generalization after augmentation. However, it is unclear if this augmentation may negatively affect the accuracy of the original labels. Similarly, if the heterogeneity of the data with same label is large, how to augment to help improve the generalization while keeping the accuracy needs to be studied carefully.

Use case generalization and customization: We showed the initial promise for two use cases. In general, we posit that synthetic data augmentation can help cooperative workflows that rely on some "structural" understanding of the traffic rather than pinpointing particular actors/actions of interest. An open question then is if we can formally scope the spectrum of ML-driven tasks that can benefit from usage of synthetic data? It also raises an attendant question if generation can be tailored better with knowledge of these tasks.

Synergy with future ML algorithms: While we explore the use of GANs, recent advances in ML suggest transformers or foundation models can be even more successful. These raises two synergistic questions. First, can these transformer based approaches or other techniques be used for synthetic data generation [37]? Second, these models often require significantly more data to train to be useful and may not naturally be considered in a sparse data setting. Synthetic data augmentation could thus even be an enabler for exploring more powerful ML models that would have otherwise been stymied by data sparsity.

Bibliography

[1] "Opaque, inc." https://opaque.co/, 16 May 2022.

[2] "Cisco IOS NetFlow," https://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow/index.html, 20 May 2022.

[3] "Duality technologies," https://dualitytech.com/, 20 May 2022.

[4] A. Adas, "Traffic models in broadband networks," *IEEE Communications Magazine*, vol. 35, no. 7, pp. 82–89, 1997.

[5] W. Aiello, A. Gilbert, B. Rexroad, and V. Sekar, "Sparse approximations for high fidelity compression of network traffic data," in *Proceedings of the 5th ACM SIGCOMM conference on Internet measurement.* USENIX Association, 2005, pp. 22–22.

[6] B. Arzani, S. Ciraci, B. T. Loo, A. Schuster, and G. Outhred, "Taking the blame game out of data centers operations with netpoirot," in *Proceedings of the 2016 ACM SIGCOMM Conference*, 2016, pp. 440–453.

[7] B. Arzani, S. Ciraci, S. Saroiu, A. Wolman, J. Stokes, G. Outhred, and L. Diwu, "Privateeye: Scalable and privacy-preserving compromise detection in the cloud," in *17th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 20)*, 2020, pp. 797–815.

[8] S. Augenstein, H. B. McMahan, D. Ramage, S. Ramaswamy, P. Kairouz, M. Chen, R. Mathews, and B. A. y Arcas, "Generative models for effective ml on private, decentralized datasets," in *International Conference on Learning Representations*, 2019.

[9] S. Bergsma, T. Zeyl, A. Senderovich, and J. C. Beck, "Generating complex, realistic cloud workloads using recurrent neural networks," in *Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles*, 2021, pp. 376–391.

[10] D. Bisson, "Cloud vs. on-premises: Understanding the security differences," Tripwire, April 2018.

[11] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," *arXiv preprint arXiv:1902.01046*, 2019.

[12] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," *SIGMOD Rec.*, vol. 29, no. 2, p. 93–104, may 2000. [Online]. Available: https://doi.org/10.1145/335191.335388.

[13] M. Caselli, D. Hadžiosmanović, E. Zambon, and F. Kargl, "On the feasibility of device fingerprinting in industrial control systems," in *International Workshop on Critical Information Infrastructures Security.* Springer, 2013, pp. 155–166.

[14] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, "Generating multi-label discrete patient records using generative adversarial networks," *arXiv preprint arXiv:1703.06490*, 2017.

[15] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, "Benchmarking cloud serving systems with ycsb," in *Proceedings of the 1st ACM symposium on Cloud computing*, 2010, pp. 143–154.

[16] S. Di and F. Cappello, "Gloudsim: Google trace based cloud simulator with virtual machines," *Software: Practice and Experience*, vol. 45, no. 11, pp. 1571–1590, 2015.

[17] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[18] J. Fan, J. Xu, and M. H. Ammar, "Crypto-pan: Cryptography-based prefix-preserving anonymization," *Computer Networks*, vol. 46, no. 2, 2004.

[19] N. Feamster and J. Rexford, "Why (and how) networks should run themselves," in *Proceedings of the Applied Networking Research Workshop*, ser. ANRW '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 20. [Online]. Available: https://doi.org/10.1145/3232755.3234555.

[20] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Synthetic data augmentation using gan for improved liver lesion classification," in

2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018). IEEE, 2018, pp. 289–293.

[21] J. Gao, N. Yaseen, R. MacDavid, F. V. Frujeri, V. Liu, R. Bianchini, R. Aditya, X. Wang, H. Lee, D. Maltz, M. Yu, and B. Arzani, "Scouts: Improving the diagnosis process through domain-customized incident routing," in *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, 2020, pp. 253–269.

[22] A. M. González, A. S. Roque, and J. García-González, "Modeling and forecasting electricity prices with input/output hidden markov models," *IEEE Transactions on Power Systems*, vol. 20, no. 1, pp. 13–24, 2005.

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

[24] J. T. Guibas, T. S. Virdi, and P. S. Li, "Synthetic medical images from dual generative adversarial networks," *arXiv preprint arXiv:1709.01872*, 2017.

[25] C. Han, H. Hayashi, L. Rundo, R. Araki, W. Shimoda, S. Muramatsu, Y. Furukawa, G. Mauri, and H. Nakayama, "Gan-based synthetic brain mr image generation," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018).* IEEE, 2018, pp. 734–738.

[26] L. Han, Y. Sheng, and X. Zeng, "A packet-length-adjustable attention model based on bytes embedding using flow-wgan for smart cybersecurity," *IEEE Access*, vol. 7, pp. 82 913–82 926, 2019.

[27] M. R. Hassan and B. Nath, "Stock market forecasting using hidden markov model: a new approach," in *Intelligent Systems Design and Applications, 2005. ISDA'05. Proceedings. 5th International Conference on.* IEEE, 2005, pp. 192–196.

[28] J. Holland, P. Schmitt, N. Feamster, and P. Mittal, "New directions in automated traffic analysis," ser. CCS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 3366–3383. [Online]. Available: https://doi.org/10.1145/3460120.3484758.

[29] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[30] T. Issariyakul and E. Hossain, "Introduction to network simulator 2 (ns2)," in *Introduction to network simulator NS2*. Springer, 2009, pp. 1–18.

[31] J. Jiang, R. Das, G. Ananthanarayanan, P. A. Chou, V. Padmanabhan, V. Sekar, E. Dominique, M. Goliszewski, D. Kukoleca, R. Vafin *et al.*, "Via: Improving internet telephony call quality using predictive relay selection," in *Proceedings of the 2016 ACM SIGCOMM Conference*. ACM, 2016, pp. 286–299.

[32] J. Jiang, V. Sekar, H. Milner, D. Shepherd, I. Stoica, and H. Zhang, "Cfa: A practical prediction system for video qoe optimization." in *NSDI*, 2016, pp. 137–150.

[33] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. D'Oliveira, H. Eichner, S. El Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascon, B. Ghazi, P. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecny, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Ozgur, H. Pagh, Rasmus Qi, D. Ramage, R. Raskar, M. Raykova, D. Song, W. Song, S. Stich, Z. Sun, A. T. Suresh, F. Tramer, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, and H. Yu, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[34] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[35] C. Kruegel, "Survey highlights security gaps – from inadequate staffing to excessive false positives to risk elevated by cloud migration," https://www.lastline.com/blog/survey-highlights-security-gaps-from-inadequate-staffing-to-excessive-false-positives-to-risk-elevated-by-cloud-migration/, 2020, accessed on May 14, 2021.

[36] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," *arXiv preprint arXiv:1603.01360*, 2016.

[37] F. Le, M. Srivatsa, R. Ganti, and V. Sekar, "Rethinking data-driven networking with foundation models: Challenges and opportunities," in *Proceedings of the 21st ACM Workshop on Hot Topics in Networks*, ser. HotNets '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 188–197. [Online]. Available: https://doi.org/10.1145/3563766.3564109.

[38] Q. Li, H. Jianming, and Z. Yi, "A flow volumes data compression approach for traffic network based on principal component analysis," in *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE.* IEEE, 2007, pp. 125–130.

[39] Y. Li, Y.-n. Dong, H. Zhang, H.-t. Zhao, H.-x. Shi, and X.-x. Zhao, "Spectrum usage prediction based on high-order markov model for cognitive radio networks," in *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on.* IEEE, 2010, pp. 2784–2788.

[40] Z. Lin, A. Jain, C. Wang, G. Fanti, and V. Sekar, "Using gans for sharing networked time series data: Challenges, initial promise, and open questions," in *Proceedings of the ACM Internet Measurement Conference*, 2020, pp. 464–483.

[41] Z. Lin, H. Liang, G. Fanti, V. Sekar, R. A. Sharma, E. Soltanaghaei, A. Rowe, H. Namkung, Z. Liu, D. Kim *et al.*, "Raregan: Generating samples for rare classes," *arXiv preprint arXiv:2203.10674*, 2022.

[42] S. Lohr, "For big-data scientists,'janitor work'is key hurdle to insights," *New York Times*, vol. 17, p. B4, 2014.

[43] D. Magalhães, R. N. Calheiros, R. Buyya, and D. G. Gomes, "Workload modeling for resource usage analysis and simulation in cloud computing," *Computers & Electrical Engineering*, vol. 47, pp. 69–81, 2015.

[44] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016. [Online]. Available: https://arxiv.org/abs/1602.05629.

[45] B. Melamed, "An overview of tes processes and modeling methodology," in *Performance Evaluation of Computer and Communication Systems.* Springer, 1993, pp. 359–393.

[46] B. Melamed and J. R. Hill, "Applications of the tes modeling methodology," in *Proceedings of 1993 Winter Simulation Conference-(WSC'93).* IEEE, 1993, pp. 1330–1338.

[47] B. Melamed, J. R. Hill, and D. Goldsman, "The tes methodology: Modeling empirical stationary time series," in *Proceedings of the 24th conference on Winter simulation.* ACM, 1992, pp. 135–144.

[48] B. Melamed and D. E. Pendarakis, "Modeling full-length vbr video using markov-renewal-modulated tes models," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 5, pp. 600–611, 1998.

[49] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” *arXiv preprint arXiv:1411.1784*, 2014.

[50] I. S. Moreno, P. Garraghan, P. Townend, and J. Xu, “Analysis, modeling and simulation of workload patterns in a large-scale utility cloud,” *IEEE Transactions on Cloud Computing*, vol. 2, no. 2, pp. 208–221, 2014.

[51] N. Moustafa, “A new distributed architecture for evaluating ai-based security systems at the edge: Network ton_iot datasets,” *Sustainable Cities and Society*, vol. 72, p. 102994, 2021.

[52] N. Msadek, R. Soua, and T. Engel, “Iot device fingerprinting: Machine learning based encrypted traffic analysis,” in *2019 IEEE wireless communications and networking conference (WCNC)*. IEEE, 2019, pp. 1–8.

[53] A. Narayanan and V. Shmatikov, “Robust de-anonymization of large sparse datasets,” in *Security and Privacy, 2008. SP 2008. IEEE Symposium on*. IEEE, 2008, pp. 111–125.

[54] P. Notaro, J. Cardoso, and M. Gerndt, “A survey of aiops methods for failure management,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 12, no. 6, pp. 1–45, 2021.

[55] A. Nucci, A. Sridharan, and N. Taft, “The problem of synthetically generating ip traffic matrices: initial recommendations,” *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 3, pp. 19–32, 2005.

[56] R. Pang, M. Allman, V. Paxson, and J. Lee, “The devil and packet trace anonymization,” *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 1, pp. 29–38, 2006.

[57] F. Parastar, M. Sepahi, and M. Moudi, “On-device ml for the current and the emerging networks: A survey on current approaches and challenges.”

[58] R. Poddar, S. Kalra, A. Yanai, R. Deng, R. A. Popa, and J. M. Hellerstein, “Senate: A maliciously-secure MPC platform for collaborative analytics,” in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 2129–2146.

[59] C. Reiss, J. Wilkes, and J. L. Hellerstein, “Obfuscatory obscanturism: making workload traces of commercially-sensitive systems safe to release,” in *2012 IEEE Network Operations and Management Symposium*. IEEE, 2012, pp. 1279–1286.

[60] C. Security, "How manual application vulnerability management delays innovation and increases business risk," https://www.contrastsecurity.com/hubfs/How-Manual-Application_eBook_042020_Final.pdf?hsLang=en, accessed on May 14, 2021.

[61] H.-C. Shin, N. A. Tenenholtz, J. K. Rogers, C. G. Schwarz, M. L. Senjem, J. L. Gunter, K. P. Andriole, and M. Michalski, "Medical image synthesis for data augmentation and anonymization using generative adversarial networks," in *International Workshop on Simulation and Synthesis in Medical Imaging*. Springer, 2018, pp. 1–11.

[62] L. Sliwko and V. Getov, "Agocs—accurate google cloud simulator framework," in *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. IEEE, 2016, pp. 550–558.

[63] J. Sommers, R. Bowden, B. Eriksson, P. Barford, M. Roughan, and N. Duffield, "Efficient network-wide flow record generation," in *2011 Proceedings IEEE INFOCOM*. IEEE, 2011, pp. 2363–2371.

[64] J. Sommers, V. Yegneswaran, and P. Barford, "A framework for malicious workload generation," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, 2004, pp. 82–87.

[65] A. Soule, A. Nucci, R. Cruz, E. Leonardi, and N. Taft, "How to identify and estimate the largest traffic matrix elements in a dynamic environment," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 32, no. 1. ACM, 2004, pp. 73–84.

[66] V. Tarasov, E. Zadok, and S. Shepler, "Filebench: A flexible framework for file system benchmarking," *USENIX; login*, vol. 41, no. 1, pp. 6–12, 2016.

[67] M. Valipour, M. E. Banihabib, and S. M. R. Behbahani, "Comparison of the arma, arima, and the autoregressive artificial neural network models in forecasting the monthly inflow of dez dam reservoir," *Journal of hydrology*, vol. 476, pp. 433–441, 2013.

[68] N. Volgushev, M. Schwarzkopf, B. Getchell, M. Varia, A. Lapets, and A. Bestavros, "Conclave: secure multi-party computation on big data," in *Proceedings of the Fourteenth EuroSys Conference 2019*, 2019, pp. 1–18.

[69] P. Wang, S. Li, F. Ye, Z. Wang, and M. Zhang, "Packetcgan: Exploratory study of class imbalance for encrypted traffic classification using cgan," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–7.

[70] R. Xu, J. B. Joshi, and C. Li, "Cryptonn: Training neural networks over encrypted data," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 1199–1209.

[71] S. Xu, M. Marwah, and N. Ramakrishnan, "Stan: Synthetic network traffic generation using autoregressive neural models," *arXiv preprint arXiv:2009.12740*, 2020.

[72] F. Y. Yan, H. Ayers, C. Zhu, S. Fouladi, J. Hong, K. Zhang, P. Levis, and K. Winstein, "Learning in situ: a randomized experiment in video streaming," in *Proc NSDI*, 2020.

[73] K. Yang, S. Kpotufe, and N. Feamster, "Feature extraction for novelty detection in network traffic," *arXiv preprint arXiv:2006.16993*, 2020.

[74] Y. Yin, Z. Lin, M. Jin, G. Fanti, and V. Sekar, "Practical gan-based synthetic ip header trace generation using netshare," in *Proceedings of the ACM SIGCOMM 2022 Conference*, ser. SIGCOMM '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 458–472. [Online]. Available: https://doi.org/10.1145/3544216.3544251.

[75] Z. R. Zaidi and B. L. Mark, "Mobility estimation for wireless networks based on an autoregressive model," in *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE*, vol. 6. IEEE, 2004, pp. 3405–3409.

[76] G. P. Zhang, "Time series forecasting using a hybrid arima and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[77] Q. Zhang, G. Yu, C. Guo, Y. Dang, N. Swanson, X. Yang, R. Yao, M. Chintalapati, A. Krishnamurthy, and T. Anderson, "Deepview: Virtual disk failure diagnosis and pattern detection for azure," in *15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18)*, 2018, pp. 519–532.