ADVANCING FAIRNESS AND DIFFERENTIAL PRIVACY IN MACHINE LEARNING FOR SOCIALLY RELEVANT APPLICATIONS

MAYANA PEREIRA

ORIENTADOR: RAFAEL TIMÓTEO DE SOUSA JÚNIOR, PROFESSOR, ENE/UNB

TESE DE DOUTORADO
EM ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Elétrica

Advancing Fairness and Differential Privacy in Machine Learning for Socially Relevant Applications

Mayana Pereira
Orientador: Rafael Timóteo de Sousa Júnior, Professor, ENE/UNB

TESE DE DOUTORADO SUBMETIDA AO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR.

APROVADA POR:

Fábio Mendonça, Professor Associado (Universidade de Brasília)
(Presidente da Banca)

Ricardo Custódio, Professor Titular (Universidade Federal de Santa Catarina)
(Examinador Externo)

Mario Laranjeira, Professor Associado (Tokyo Institute of Technology)
(Examinador Externo)

William Ferreira Giozza, Professor Associado (Universidade de Brasília)
(Examinador Interno)

Brasília/DF, Abril de 2024.

FICHA CATALOGRÁFICA

PEREIRA, MAYANA

Advancing Fairness and Differential Privacy in Machine Learning for Socially Relevant Applications. [Brasília/DF] 2024.

xiii, 101p., 210 x 297 mm (ENE/FT/UnB, Doutor, Tese de Doutorado, 2024).

Universidade de Brasília, Faculdade de Tecnologia, Departamento de Engenharia Elétrica.

Departamento de Engenharia Elétrica

1. Aprendizado de Máquina
2. Privacidade Diferencial
3. Dados Sintéticos
4. Medias com conteúdo de abuso
5. Equidade algorítmica
6. Inteligência artificial

I. ENE/FT/UnB
II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

PEREIRA, MAYANA (2024). Advancing Fairness and Differential Privacy in Machine Learning for Socially Relevant Applications. Tese de Doutorado, Publicação PPGEE.205/2024, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 101p.

CESSÃO DE DIREITOS

AUTOR: Mayana Pereira

TÍTULO: Advancing Fairness and Differential Privacy in Machine Learning for Socially Relevant Applications.

GRAU: Doutor ANO: 2024

Mayana Pereira

Universidade de Brasília (UnB)

Campus Darcy Ribeiro

Faculdade de Tecnologia - FT

Departamento de Engenharia Elétrica(ENE)

Brasília - DF CEP 70919-970

Aos meus amores Anderson, Antonio e João, e à minha fiel companheira Izzy.

ACKNOWLEDGEMENTS

First and foremost, I would like to acknowledge the impact that my advisor, Rafael Timoteo de Sousa, had on my research. His guidance and mentorship have been instrumental in shaping my research focus. I am particularly grateful for his emphasis on addressing problems with direct applications to society and the local community.

I want to thank Microsoft's AI for Good research lab, which has greatly influenced my research journey. The lab's culture of fostering a growth mindset, with constant discussions on how to leverage data and AI to make a positive impact in society has been transformative. I would like to extend my heartfelt thanks to Rahul Dodhia, Juan Lavista, Kevin White, Allen Kim, Darren Tanner, Meghana Kshirsagar, and Sumit Mukherjee for their collaboration, discussions, and contributions.

I would also like to acknowledge my collaborators from the University of Washington, Sikha Pentyala and Martine DeCock, for their partnership in advancing the state-of-the-art in the synthetic data field. Their expertise and dedication have contributed to the success of our joint efforts.

Furthermore, I am grateful to Project VIC and Richard Brown for placing their trust in me to develop a solution aimed at advancing technologies to eradicate child sexual abuse media online. I am honored to have been given the opportunity to contribute to such an important cause.

Finally, I would like to express my appreciation to all the individuals, too numerous to mention individually, who have supported and inspired me throughout this research journey. Your encouragement, guidance, and constructive feedback have been invaluable, and I am grateful for your contributions. Thank you all for your support and for being part of my academic and personal growth.

ABSTRACT

Title: Advancing Fairness and Differential Privacy in Machine Learning for Socially Relevant Applications

This thesis investigates privacy-preserving machine learning techniques for socially relevant applications. Specifically, this work tackles three important problems: the detection and identification of medias with abuse content, with a special focus on child sexual abuse media (CSAM); the fairness impacts of utilizing private synthetic datasets in machine learning pipelines; and the generation of privacy-preserving synthetic data sets from distributed sources.

We address the challenge of developing machine learning-based solutions for CSAM detection while considering the ethical and legal constraints of using explicit imagery for model training. To circumvent these limitations, we propose a novel framework that leverages file metadata for CSAM identification. Our approach involves training and evaluating deployment-ready machine learning models based on file paths, demonstrating its effectiveness on a dataset of over one million file paths collected from actual investigations. Additionally, we assess the robustness of our solution against adversarial attacks and explore the use of differential privacy to protect the model from model inference attacks without sacrificing utility.

In the second part of this thesis, we investigate the opportunities and challenges of utilizing synthetic data generation in the context of increasing global privacy regulations. Synthetic data mimics real data without replicating personal information, and offers various possibilities for data analysis and machine learning tasks. This work addresses the impacts of using synthetic data sets in machine learning pipelines, especially when only synthetic data is available for training and evaluation. This thesis examines the relationship between differential privacy and machine learning fairness, exploring how different synthetic data generation methods affect the fairness and comparing the performance of models trained and tested with synthetic data versus real data. The findings contribute to a better understanding of synthetic data usage in machine learning pipelines and its potential to advance research across various fields.

The third and final part of this thesis proposes a protocol for generating privacy-preserving synthetic data sets from distributed data. This thesis proposes the first protocol for generation of synthetic data sets from distributed sources with differentially private guarantees, without the need for a trusted dealer. The goal of this approach is to enable data holders to share data without violating legal and ethical restrictions.

Keywords: machine learning, differential privacy, synthetic data, child sexual abuse media, algorithmic fairness, artificial intelligence

RESUMO

Título: Avanços em Equidade e Privacidade Diferencial em Aprendizado de Máquina para Aplicações Socialmente Relevantes

Esta tese investiga técnicas de aprendizado de máquina que preservam a privacidade para aplicações socialmente relevantes, focando em duas áreas específicas: detecção e identificação de Mídia de Abuso Sexual Infantil (CSAM) e geração de conjuntos de dados sintéticos que com foco em desenvolvimento ético e privado de inteligencia artifical.

Abordamos o desafio de desenvolver soluções baseadas em aprendizado de máquina para detecção de CSAM enquanto consideramos as restrições éticas e legais do uso de imagens explícitas para treinamento do modelo. Para contornar essas limitações, propomos uma nova estrutura que utiliza metadados de arquivo para identificação de CSAM. Nossa abordagem envolve o treinamento e avaliação de modelos de aprendizado de máquina prontos para implantação baseados em caminhos de arquivo, demonstrando sua eficácia em um conjunto de dados de mais de um milhão de caminhos de arquivo coletados em investigações reais. Além disso, avaliamos a robustez de nossa solução contra ataques adversáriais e exploramos o uso de privacidade diferencial para proteger o modelo de ataques de inferência de modelo sem sacrificar a utilidade.

Na segunda parte desta tese, investigamos as oportunidades e desafios do uso da geração de dados sintéticos no contexto do aumento da adoção de regulamentações globais de privacidade. Dados sintéticos são dados que imitam dados reais sem replicar informações pessoais, e oferecem diversas possibilidades para análise de dados e tarefas de aprendizado de máquina. No entanto, pouco se sabe sobre os impactos do uso de bancos de dados sintéticos em pipelines de aprendizado de máquina, especialmente quando apenas dados sintéticos estão disponíveis para treinamento e avaliação de modelo. Este estudo examina a relação entre privacidade diferencial e viés social dos algoritmos aprendizado de máquina, explorando como diferentes métodos de geração de dados sintéticos afetam o viés social dos algoritmos e comparando o desempenho de modelos treinados e testados com dados sintéticos versus dados reais. Os resultados contri-

buem para uma melhor compreensão do uso de dados sintéticos em pipelines de aprendizado de máquina e seu potencial para avançar o estado da arte em diversas áreas.

A terceira e última parte desta tese propõe um protocolo para a geração de bancos de dados sintéticos que preservam a privacidade a partir de dados distribuídos. Esta tese propõe o primeiro protocolo para a geração de bancos de dados sintéticos a partir de fontes distribuídas com garantias de privacidade diferencial, sem a necessidade de um negociante confiável. O objetivo desta abordagem é permitir que os detentores de dados compartilhem dados sem violar restrições legais e éticas.

Palavras-chave: aprendizado de máquina, dados sintéticos, privacidade diferencial, mídia de abuso sexual infantil, imparcialidade algorítimica, inteligência artificial

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

ϵ	privacy-loss parameter
δ	privacy-leak parameter
$\|x\|$	cardinality of x
$\|x\|_1$	l_1-norm of x
$\|x\|_2$	l_2-norm of x
Δf	l_1-sensitivity of f
$\Delta_2 f$	l_2-sensitivity of f
$\mathsf{Lap}(\lambda)$	Laplace distribution with mean 0 and scale λ
$\mathsf{N}(0,\sigma^2)$	Gaussian distribution with mean 0 and scale σ^2
∇	gradient
$[\![x]\!]$	secret sharing of x

GLOSSARY

2PC	Two-Party Computation
3PC	Three-Party Computation
4PC	Four-Party Computation
AI	Artificial intelligence
AIM	Adaptive and Interactive Mechanism
AUC	Area Under the Curve
BERT	Bidirectional Encoder Representations from Transformers
CNN	Convolutional Neural Network
CSAM	Child sexual abuse media
CTGAN	Conditional Tabular Generative Adversarial Network
DEO	Difference in Equality of Odds
DP	Differential Privacy
DP-SGD	Differential Privacy Stochastic Gradient Descent
DSP	Difference in Statistical Parity
FPR	False Positive Rate
GAN	Generative Adversarial Networks
GDPR	General Data Protection Regulation
LSTM	Long Short-Term Memory
ML	Machine learning
MPC	MultiParty Computation

MST	Maximum Spanning Tree
MWEM	Multiplicative Weights Exponential Mechanism
MWEM-PGM	Multiplicative Weights Exponential Mechanism-Probabilistic Graphical Model
NGO	Non-governmental organization
p2p	Peer-to-peer
PATE	Private Aggregation of Teacher Ensembles
SGD	Stochastic Gradient Descent
TF-IDF	Term Frequency-Inverse Document Frequency
TPR	True Positive Rate

CHAPTER 1

INTRODUCTION

The widespread use of digital technology has dramatically changed how people communicate, learn, work, and interact. However, it has also created new challenges for ensuring user safety and well-being, particularly in ubiquitous communication and data sharing. While technology companies bear responsibility for addressing harmful social behaviors facilitated by these platforms, the collection of user data can create socially impactful opportunities and drive research and knowledge advancement across many fields. This data can provide researchers with unparalleled insights into health, education, economy, psychology, sociology and other sciences, still, due to privacy restrictions, most of this data is kept away from researchers. The combination of data sharing, machine learning, and artificial intelligence has revolutionized many research areas enabling access to the vast amounts of data that are locked in data silos due to privacy restrictions will be key to research in many fields. This work focuses on a social good perspective of the challenges and opportunities presented by technology platforms that share and collect data by examining and proposing machine learning and statistical models while emphasizing the importance of balancing ethical and legal considerations.

On the challenges side, this work focus on social issues that significantly influence product development and customer relationships in large organizations, particularly those handling user-generated content like Pinterest, Facebook, Microsoft, Apple, and Google. One prominent example is the widespread presence of child sexual abuse material (CSAM) on digital platforms. Recognizing the seriousness of this issue, these companies have prioritized detecting and removing CSAM. Nevertheless, despite the collective efforts of non-profit organizations like Project VIC International[1], Thorn[2], and the Internet Watch Foundation[3], which focus on creating tools to fight CSAM, its production, and distribution continue to be pressing and expanding challenges. In the past decade, the amount of CSAM on digital platforms has grown exponentially,

[1] <https://www.projectvic.org>
[2] <https://www.thorn.org>
[3] <https://www.iwf.org.uk>

driven by the increasing popularity of online sharing platforms and social media. Given this context, artificial intelligence presents a potential solution for tackling this large-scale problem, providing innovative methods to curb the spread of harmful media and safeguard vulnerable individuals.

The COVID-19 pandemic triggered a significant increase in the distribution of CSAM via social media and video conferencing apps (SOLON, 2020). The identification of CSAM is a highly challenging problem. First, it can manifest in different types of material: images, videos, streaming, video conference, and online gaming, among others. Undiscovered and unlabeled CSAM on the internet is estimated to be magnitudes greater than the currently identified CSAM. Second, discovering new material is still highly dependent on human discovery. Despite the significant progress in machine learning models for CSAM identification with modern deep-learning architectures (VITORINO *et al.*, 2016; MACEDO *et al.*, 2018; Yiallourou *et al.*, 2017), these models rely on the availability of labeled images, which can lead to technical limitations. Training artificial intelligence (AI) models requires large datasets, and in the case of ethically and legally sensitive problems, such as CSAM, training data presents significant challenges since the content depicts illegal activities, and possessing it is a crime. This presents a substantial hurdle for data practitioners and researchers working to advance technology in social issues with ethical, legal and safety concerns.

As new material is created daily, we understand that utilizing complementary signals can advance the capability of digital platforms in detecting and removing illegal content. The use of metadata has been proposed in the past by (PEERSMAN *et al.*, 2016). This is an effective approach since distributors use coded language to communicate and trade links of CSAM hosted in plain sight on content sharing platforms, websites, newsgroups, bulletin boards, peer-to-peer networks, internet gaming sites, social networking sites, and anonymized networks [4]. In particular, peer-to-peer (p2p) file sharing networks is an environment where CSAM is actively hosted and shared (LATAPY *et al.*, 2013; FOURNIER *et al.*, 2014), and searches in p2p networks usually work by matching search terms with filenames and file paths.

Considering that those producing and sharing such content may actively work to evade detection tools and mechanisms is important. This leads to an adversarial situation in which

[4]https://www.thorn.org/child-pornography-and-abuse-statistics/

the offender attempts to trick the AI model to prevent CSAM detection. Identifying models less vulnerable to adversarial attacks is crucial in these circumstances. In addition to the security, privacy is another important aspect to consider. After training AI models, law enforcement agencies and NGOs, are often interested in collaborating and sharing their solutions. In this context, ensuring that models are trained with privacy safeguards is essential to protect against inference attacks, which could disclose the identities of victims included in the AI model's training dataset. AI solutions can be responsibly and effectively developed to identify and counteract CSAM by addressing these ethical, privacy, and security concerns

On the opportunities side, the data collected daily by technology systems for logs, records, and telemetry purposes helps researchers and industry understand our behavior better on both individual and collective levels, and allows important research studies in many disciplines, including health, education, and economy. At the same time, we see an increase in privacy regulations globally. Following the introduction of the GDPR,[5] more than 60 jurisdictions worldwide have proposed postmodern data privacy protection laws. By 2024, 75% of the world's population will have their personal information covered under modern privacy regulations (RIMOL, 2022). While privacy regulations are of extreme importance from an ethics perspective, they can potentially result in data stored in silos, compromising data usage and sharing, consequently stalling research.

Synthetic data generation is emerging as a paradigm to break this data logjam. While data synthesis is arguably best known as a means to create training examples for data hungry deep learning models (NIKOLENKO, 2021), it is increasingly acknowledged and proposed as a privacy-enhancing technology (PET) (JORDON *et al.*, 2018; MCKENNA *et al.*, 2021; Science and Technology Policy Office, 2022; TORKZADEHMAHANI *et al.*, 2019; WALONOSKI *et al.*, 2018; XIE *et al.*, 2018). When done well, synthetic data has the same distribution or characteristics as the underlying, real data, but, crucially, without replicating personal information. The latter is often formalized through Differential Privacy (DP) (DWORK *et al.*, 2006a), which intuitively means that the synthetic data should not reveal specifics about *individual* records in the underlying, real data.

Once synthetic data sets become publicly available, they can be used for any analysis and

[5]European General Data Protection Regulation <https://gdpr-info.eu/>

task. From simple statistical analysis, combining the data with other sources for a data augmentation, to training and evaluating machine learning models using solely synthetic data sets are some of the possibilities synthetic data sets offer. However, little is understood about the impacts of using such data sets for all these different type of tasks. For machine learning related tasks, there are many unknowns about the usage of synthetic data sets in machine learning pipelines, specially in scenarios where synthetic data is used for training and evaluation.

One important aspect that needs to be considered in machine learning pipelines that utilizes synthetic data, is how synthetic data might affect algorithmic fairness. Algorithmic fairness gained much attention with the increase of the utilization of machine learning (ML) models in decision making. In domains protected by anti-discrimination laws (HARDT *et al.*, 2016), while ML models can drastically improve the decision making process, it can also strengthen biases presented in the data, and even introduce new biases (BAROCAS *et al.*, 2017b). In 2016 a White House report (BAROCAS *et al.*, 2017a) introduced a requirement of "equal opportunity by design" for big data and machine learning systems in domains covered by anti-discrimination regulations. As a response, several works analysing the impacts of machine learning models in decision making emerged, where many of such works concluded that ML models can have disparate impacts on minoritized subgroups (WIENS *et al.*, 2019; COHEN *et al.*, 2020; RAJOTTE *et al.*, 2021). Unfairness in machine learning can happen, among other reasons, due to class imbalance and intrinsic bias in the underlying training dataset. It is known that differential privacy can affect fairness in machine learning models. However, despite significant work on addressing the relationship between differential privacy and ML fairness, fundamental questions remain unanswered. For instance, how different synthetic data generation methods affect fairness is unknown. Also, in the literature, it is usually assumed that real data is available for testing models trained on synthetic data before deployment. This is not a realistic assumption. In many scenarios, only synthetic data is available during training and testing. So, it is important to study the performance of a model trained and tested with synthetic data vs. its performance when tested against real data. We study this scenario to answer the question about impacts of synthetic data in machine learning pipelines.

Furthermore, what the existing approaches for generating synthetic data and publication of data with DP guarantees have in common, is that they all assume that the original, real data,

exists with one data holder, or, if the data originates from different data holders, that the latter are able to send their data to a central aggregator who in turn will use it as input for synthetic data generation or DP publication algorithms. Much of the valuable data in the world however is under the control of entities (companies, banks, hospitals, biomedical research institutes etc.) who cannot show their data to each other or to a central aggregator without raising privacy concerns. This is the bottleneck that we address, namely *how to generate synthetic data based on the combined data from multiple data holders that no one is allowed to see*. This includes data that is horizontally distributed, such as healthcare data across different hospitals, or financial data held by different banks, as well as data that is vertically distributed, such as advertising data where the publishers hold the input features while advertisers have the label, and many more. Finally, in addition to the cross-silo scenarios described above, our proposed solution makes a scenario practical where millions of users could provide their data to produce a synthetic dataset in a way that private, individual data would never be exposed in plaintext (i.e. without being encrypted) to any entity – practically implementing a "synthetic data as a service"model.

1.1 CONTRIBUTIONS

The objective of this thesis is to develop frameworks and provide analysis that can facilitate the practical, reliable, and ethical implementation of privacy-preserving AI tools in socially relevant scenarios. We explored important unresolved issues, such as ensuring robust AI deployment for detecting abusive media, examining the impact on utility and fairness when utilizing synthetic datasets in end-to-end machine learning pipelines, and generating privacy-preserving synthetic data from distributed sources.

By integrating these three research areas, this thesis offers practical frameworks for creating artificial intelligence solutions that address socially relevant problems, while upholding ethical standards, ensuring privacy in data sharing, and promoting algorithmic fairness.

1.1.1 A Framework for Metadata-based Detection of Child Sexual Abuse Material

In the first part of this work, we propose a comprehensive framework for designing, training and testing machine learning models that focus on child safety. Our approach carefully addresses ethical and privacy concerns and considers potential adversarial attacks by malicious actors. This framework encompasses the following aspects: Developing novel machine learning models tailored to detect and identify content related to child safety, while incorporating ethical and privacy-preserving techniques such as differential privacy. Implementing robust evaluation methodologies for these models, considering various scenarios, data distributions, and potential adversarial attacks. Investigating methods to counter adversarial attacks and improve the model's resilience against malicious attempts to bypass or manipulate the system.

The scarcity of frameworks for evaluating machine learning models for CSAM detection prevents a better understanding of model performance under multiple scenarios that can happen during deployment. Before deployment, organizations should test the CSAM detection model under different conditions. An evaluation scenario needs a real-world data set with similar data distributions to what the model will get exposed to after deployment. A critical scenario for analysis is testing the model on completely benign out-of-sample data sets. The burden caused by a high false-positive rate can halt the deployment of such systems. Furthermore, it is crucial to understand how adversarially modified data impact model performance.

We list the contributions as follows:

- We propose a framework for evaluating machine learning models for CSAM identification to prepare for deployment. Our framework proposes a testing pipeline that covers real-world scenarios that should be expected when deploying a machine learning model for CSAM detection: (i) test on CSAM and non-CSAM samples; (ii) test on adversarially modified CSAM samples to evade detection; (iii) test on benign samples from open data sources.

- We train and compare several machine-learning models that analyze file paths and file names from file storage systems and determine a probability that a given file has child sexual abuse content. Our experiments include traditional machine learning algorithms, deep neural network architectures, Transformers-based models and differentially private

models. We train our models on a real-world data set containing over one million file paths from apprehended hard drives during investigations. It is the most extensive file path data set composed solely of file paths from apprehended hard drives.

Our best classifier achieves recall rates over 0.94 and accuracy over 0.97 on holdout sets; it maintains a high recall rate in adversarially modified inputs; when tested against benign samples from other data distributions, it achieves a false-positive rate of ≈ 0.01.

Although previous works in the literature have proposed machine learning models for CSAM detection, we propose the first framework for *evaluating* CSAM detection systems that includes adversarial examples in the evaluation stage. Our results show that machine learning based on file paths can effectively detect CSAM in storage systems and achieve the aspired performance in all the proposed evaluation scenarios. Our work is also the first to train CSAM models with differentially private learning algorithms.

1.1.2 Assessment of differentially private synthetic data for utility and fairness in end-to-end machine learning pipelines for tabular data

The second part of this work explores the implications of utilizing synthetic datasets with privacy guarantees for training and evaluating machine learning models. Specifically, we investigate the following aspects: Analyzing the effects of synthetic datasets on algorithmic fairness, examining potential biases that may arise from using perturbed or anonymized data. Assessing the trade-offs between privacy preservation, data utility, and fairness when generating and using synthetic datasets for machine learning tasks.

This work studies the impacts of differentially private synthetic data on downstream classification with a focus on understanding the impacts on model performance and fairness. In this chapter, we investigate the impacts of differentially private synthetic data on downstream classification, where we focus on understanding the impacts on model utility and fairness. Our investigation focus on two aspects of such impact:

- What is the impact in model utility when utilizing synthetic data for training machine learning models? Can synthetic data also be used to evaluate utility of machine learning

models?

- What is the impact in model fairness when utilizing synthetic data for training machine learning models? Can synthetic data be used to evaluate fairness of machine learning models?

This work is the first to evaluate the fairness of machine learning models trained on DP synthetic data for the important case of tabular data.

1.1.3 Secure Multiparty Computation for Synthetic Data Generation from Distributed Data

In the third part of this work, we propose the first solution for synthetic data generation from multiple data holders, where data holders only share encrypted data for differentially private synthetic data generation. Data holders send shares to servers who perform Secure Multiparty Computation (MPC) computations while the original data stays encrypted. We instantiate this idea in an MPC protocol for the Multiplicative Weights with Exponential Mechanism (MWEM) algorithm to generate synthetic data based on real data originating from many data holders without reliance on a single point of failure.

We propose and implement the first Previous proposals for differentially private synthetic data generation from distributed databases use federated learning (FL) for training the data synthesizer (BEHERA *et al.*, 2022; XIN *et al.*, 2022; XIN *et al.*, 2020).

In these methods, each data holder sends model weights (without privacy protection) to a trusted aggregator, who computes the average of model weights and adds Laplacian noise.

Our proposal removes the need for data holders to disclose model parameters, and the need to rely on a single point of failure, by emulating the trusted aggregator with MPC. Additionally, previous works utilizing FL to train data synthesizers only account for horizontally partitioned data.

While MPC has emerged as a paradigm for privacy-preserving training of ML models over distributed data (e.g. (ADAMS *et al.*, 2022; AGARWAL *et al.*, 2019a; De Cock *et al.*, 2021; GUO *et al.*, 2022; MOHASSEL; ZHANG, 2017; WAGH *et al.*, 2019)) and privacy-preserving

inference with trained ML models (e.g. (De Cock *et al.*, 2019; FRITCHMAN *et al.*, 2018; LIU *et al.*, 2017; MISHRA *et al.*, 2020; PENTYALA *et al.*, 2021)), and it has been proposed for secure computation of histograms (e.g. (BELL *et al.*, 2022)), the idea of using MPC for privacy-preserving generation of synthetic data, as we propose here, is novel and a practical and secure technological solution.

The contributions presented in Chapter 3 were published at the *IEEE Transactions on Dependable and Secure Computing* (PEREIRA *et al.*, 2023). The contributions presented in Chapter 4 were first presented at the *Machine Learning for Data: Automated Creation, Privacy, Bias Workshop* at the *International Conference on Machine Learning (ICML)* (workshop without proceedings) (PEREIRA *et al.*, 2021b). The full version of this work was published at *PLOS ONE* (PEREIRA *et al.*, 2024), which significantly extends and sub sums the previous version. Finally, the contributions presented in Chapter 5 were featured at *SyntheticData4ML Workshop* at the *Neural Information Processing Systems (NeurIPS)* conference (PEREIRA *et al.*, 2022).

CHAPTER 2

PRELIMINARIES

2.1 DIFFERENTIAL PRIVACY

Differential privacy is a rigorous privacy notion used to protect an individual's data in a data set disclosure. We present in this section notation and definitions that we will use to describe our privatization approach. We refer the reader to (DWORK *et al.*, 2014), (MCSHERRY, 2009) and (DWORK *et al.*, 2006b) for detailed explanations of these definitions and theorems.

Differential privacy allows data analysts to extract insights from databases while protecting the privacy of the individuals whose data is included. It ensures that the analysis results do not reveal sensitive information about any single person in the databases.

The main idea behind differential privacy is to add carefully calibrated noise (random values) to the results of an analysis, making it difficult for anyone to confidently determine whether a particular individual's data was present in the computation. This noise is added in a way to ensure that the overall statistics and trends are preserved while the details about specific individuals remain protected.

Differential privacy allows data analysts to extract insights from databases while protecting the privacy of the data providers. It ensures the analysis results do not reveal sensitive information about any single database entry.

The main idea behind differential privacy is to add carefully calibrated noise (random values) to the results of an analysis, making it difficult for anyone to confidently determine whether a particular individual's data was present in the computation. This noise is added in a way to ensure that the overall statistics and trends are preserved while the details about specific individuals remain protected. Differential privacy strikes a balance between data privacy and the utility of the information derived from the dataset. By controlling the amount of noise added, one can fine-tune the level of privacy protection and the usefulness of the analysis

results.

The amount of noise added to the analysis output is computed based on the sensitivity of the function used and the desired privacy guarantees. The sensitivity of a function captures the maximum change in its output due to the addition or removal of a single individual's data in the input dataset. Adjacent databases are datasets that differ by just one individual's data. To compute the necessary noise, we first determine the sensitivity of the function. Different ways to measure sensitivity, such as l_1 and l_2 norms, are chosen depending on the function and the problem context. Lower sensitivity functions have a smaller impact on individual data, while higher sensitivity functions require more noise to ensure privacy.

Once the sensitivity is determined, we must set the desired privacy guarantee, represented by the parameter ϵ (epsilon). Smaller ϵ values provide stronger privacy protection but reduce the utility of the analysis results due to the increased noise. On the other hand, larger ϵ values offer less privacy protection but maintain higher utility. With sensitivity and ϵ defined, we can compute the amount of noise to be added using various noise-generating mechanisms. Two popular mechanisms are the Laplace Mechanism and the Gaussian Mechanism.

By adding noise to the output based on the sensitivity and the desired privacy guarantee, differential privacy ensures that the probability of the output remains similar across adjacent databases, providing a formal framework to balance privacy and utility in data analysis.

We now provide formal definitions for these concepts.

2.1.1 Distance between Databases and Norms

Database distance in the context of differential privacy refers to the similarity measure between two databases. It is used to compare how different two databases are by examining the presence or absence of a single individual's data. This concept plays a critical role in understanding how to design privacy-preserving algorithms that can resist privacy attacks and ensure that individual data remains protected.

Throughout this document, we will utilize two distance metrics: l_1-norm and l_2-norm.

2.1.1.1 The l_1-norm

The l_1-norm of a database D, given by the symbol $\|D\|_1$, measures the cardinality of the database. We have

$$\|D\|_1 = |D|. \tag{2.1}$$

We can utilize the l_1-norm to measure the distance between two databases D and D'. The l_1-distance is given by $\|D - D'\|_1$, and is the number of entries where D and D' are different.

Two databases are said to be adjacent, or neighboring, if their l_1 distance is equal to one, that is, they differ by a single entry.

2.1.1.2 The l_2-norm

We will also need the concept of l_2 norm. On a $d-$dimensional space, the length of a vector $x = (x_1, x_2, ..., x_d)$ can be measured by the l_2-norm, which is defined as:

$$\|x\|_2 = \sqrt{x_1^2 + x_2^2 + ... + x_d^2}. \tag{2.2}$$

2.1.2 Sensitivity

Sensitivity refers to the maximum possible change in the output of a function when the input database changes by adding or removing a single individual's data. It quantifies the impact of a single individual's data on the function's output.

In other words, the sensitivity is the greatest amount a function output can change when computed on adjacent databases. The norms l_1 and l_2 are commonly used to measure sensitivity. Such metrics provide different ways to measure the magnitude of these changes in the function's output.

2.1.2.1 The l_1-sensitivity.

The l_1-sensitivity of a function $f : \mathcal{D} \to \mathbb{R}$ is:

$$\Delta f = max_{D,D'} \parallel f(D) - f(D') \parallel_1, \tag{2.3}$$

where D and D' are neighboring databases.

Sensitivity is a crucial concept in differential privacy because it helps to determine the amount of noise that must be added to the output to ensure privacy protection.

For example, consider a function f that computes the sum of all salaries higher than U$100 in a database containing salaries. If we have two adjacent databases D and D' (i.e., they differ by a single individual's data), the sensitivity of f is the maximum difference in the sum of salaries higher than U$100 in D and D'. The sensitivity for such function f is one. This function is one example of what we call a count query, a type of query that computes the number of records in a dataset that satisfy a certain condition or meet specific criteria.

2.1.2.2 The l_2-sensitivity.

The definition of sensitivity easily generalizes to other norms, such as the l_2 norm.

The l_2-sensitivity of a d-dimensional function $f : \mathcal{D} \rightarrow \mathbb{R}^d$ is:

$$\Delta_2 f = max_{D,D'} \parallel f(D) - f(D') \parallel_2, \tag{2.4}$$

where D and D' are neighboring databases.

2.1.3 Differential Privacy Definitions

2.1.3.1 Pure Differential Privacy

A randomized mechanism $\mathcal{M} : \mathcal{D} \rightarrow \mathcal{Y}$ with data base domain $\mathcal{D}$ and output set $\mathcal{Y}$ is ϵ-differentially private if, for any output $Y \subseteq \mathcal{Y}$ and neighboring databases $D, D' \in \mathcal{D}$ (i.e., D and D' differ in at most one entry), we have

$$Pr[\mathcal{M}(D) \in Y] \leq e^{\epsilon} Pr[\mathcal{M}(D') \in Y]. \tag{2.5}$$

The privacy loss of the mechanism is defined by the parameter $\epsilon \geq 0$.

The definition of differential privacy captures the idea that given a specific outcome or function computed from a database, the probability of that outcome being computed from database D and the probability of the same outcome being computed from database D' are very close. The closeness of these probabilities is measured by the constant ϵ (epsilon), also known as the privacy budget. The smaller the value of ϵ, the more difficult it becomes to distinguish whether database D or D' was used to compute the result of the function. Since D and D' are adjacent databases (meaning they differ by just one individual's data), this ensures that any individual in the database used for computing the function has plausible deniability about their presence (or absence) in the database.

2.1.3.2 Approximate Differential Privacy

A randomized mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{A}$ with data base domain $\mathcal{D}$ and output set $\mathcal{A}$ is (ϵ,δ)-differentially private if, for any output $A \subseteq \mathcal{Y}$ and neighboring databases $D, D' \in \mathcal{D}$ (i.e., D and D' differ in at most one entry), we have

$$Pr[\mathcal{M}(D) \in A] \leq e^{\epsilon} Pr[\mathcal{M}(D') \in A] + \delta. \tag{2.6}$$

Approximate differential privacy is a relaxation of the original differential privacy definition, which allows for a small probability of large privacy leaks. This relaxation can accommodate a broader range of more practical scenarios and achieves a better balance between privacy and utility in certain situations. Approximate differential privacy is formalized by introducing an additional parameter, δ, in the privacy guarantee.

The introduction of approximate differential privacy is beneficial for obtaining differential privacy using Gaussian noise and for achieving good levels of privacy with less noise when repeatedly querying the same database (also known as an arbitrary composition of differential privacy).

2.1.4 Differential Privacy Mechanisms

2.1.4.1 Laplace Mechanism

The Laplace distribution with 0 mean and scale λ, denoted by $\mathsf{Lap}(\lambda)$, has a probability density function $\mathsf{Lap}(x|\lambda) = \frac{1}{2\lambda}e^{-\frac{x}{\lambda}}$. It can be used to obtain an ϵ-differentially private algorithm to answer numeric queries (DWORK *et al.*, 2006b).

Let $f : \mathcal{D} \rightarrow \mathbb{R}^n$ be a numeric query. Let x be the query input and ϵ the privacy parameter. The Laplace mechanism is defined as:

$$\mathcal{M}_L(x, f(\cdot), \epsilon) = f(x) + (\eta_1, \ldots, \eta_n) \tag{2.7}$$

where η_i are drawn from the Laplace distribution $\mathsf{Lap}(\frac{\Delta f}{\epsilon})$.

The Laplace mechanism preserves ϵ-differential privacy (DWORK *et al.*, 2006b).

2.1.4.2 Gaussian Mechanism

The Gaussian mechanism adds noise to a $d-$dimensional function $f : \mathcal{D} \rightarrow \mathbb{R}^d$, with l_2-sensitivity defined as $\Delta_2 f$, by drawing from the normal distribution $\mathsf{N}(0,\sigma^2)$ samples and adding to each of the d components of the output. For $c^2 > 2\ln\frac{1.25}{\delta}$, the Gaussian mechanism with parameter $\sigma \geq \frac{c\Delta_2 f}{\epsilon}$ is (ϵ,δ)-differentially private.

2.1.4.3 Exponential Mechanism

Let $s : \mathcal{D} \times \mathcal{K} \rightarrow \mathbb{R}$ be a quality scoring function where $s(D,k)$ denotes the quality of result k on dataset D and $\mathcal{K}$ is the set of possible results. The exponential mechanism (MCSHERRY; TALWAR, 2007) E selects k from $\mathcal{R}$ such that the probability that a particular k is selected is proportional to $\exp(\epsilon \cdot s(D,k)/2)$. In other words, the exponential mechanism samples k from the distribution satisfying

$$Pr[E(D) = k] \propto \exp(\epsilon \cdot s(D,k)/2) \tag{2.8}$$

To guarantee ϵ-differential privacy, the scoring function s is required to satisfy a stability property, where for each result k the difference $|s(D,k) - s(D',k)|$ is at most the number of records that would have to be added or removed to change D to D'.

2.2 DIFFERENTIALLY PRIVATE MACHINE LEARNING

Differential privacy is the gold standard technique for protecting against membership inference attacks in machine learning models. The process of transforming a machine learning algorithm into its differentially private version involves adding random noise to the training data, model parameters, or predictions to ensure that an attacker cannot extract any sensitive information about an individual from the model or the data used to train it.

In other words, a differentially private model is a model whose model parameters are differentially private releases made on the training data. The process of training a differentially private machine learning model usually involves bounding the contribution that each individual in the training data in the model parameters. By bounding the contribution of each individual, differentialy private noise can be computed and added to model parameter updates. This process happens in the differentially private stochastic gradient descent.

2.2.1 Differentially Private Stochastic Gradient Descent

Differentially private stochastic gradient descent (DP-SGD) (ABADI *et al.*, 2016) is an algorithm that allows training machine learning models on sensitive data sets while providing a strong guarantee of privacy. The modified version of the stochastic gradient descent algorithm clips per-sample gradients to bound the contribution of individual examples. Noise from a Gaussian distribution is sampled and added to the sum of the clipped gradients in a randomly selected subset of the data, known as a mini-batch.

The general formula for DP-SGD can be written as follows. For each iteration (epoch) t:

- select a random sample L from the training data with probability $\frac{L}{N}$, where N is the size of the training data.

- Compute the gradient of the loss function for the mini-batch of data:

$$g_t =\leftarrow \nabla_\theta \mathcal{L}(x_i, \theta_t) \tag{2.9}$$

  where $\mathcal{L}$ is the loss function, x_i are data points in L, and θ_t is the model parameters.

- Clip the gradient using a clipping bound C. The value C is chosen using public (or differentially private) information.

$$\overline{g_t} \leftarrow \frac{g_t}{\max(1, \frac{\|g_t\|_2}{C})} \tag{2.10}$$

- Add random noise to the gradient:

$$g'_t = \frac{1}{|L|}\overline{g_t} + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}) \tag{2.11}$$

  where $\mathcal{N}$ represents a Gaussian distribution with zero mean and variance $\sigma^2 C^2$, and $\mathbf{I}$ is the identity matrix.

- Update the model parameters using the noisy gradient:

$$\theta_{t+1} = \theta_t - \alpha * g'_t. \tag{2.12}$$

  where α is the learning rate. This step is equivalent to taking a step in the direction of the negative gradient of the noisy loss function.

By adding noise to the gradient in this way, DP-SGD provides a strong guarantee of differential privacy while still allowing the model to converge to a good solution. The amount of noise added is controlled by the privacy parameter ϵ and the sensitivity of the loss function to changes in the data, which is bounded by a value C. The noise variance σ^2 is chosen to ensure the algorithm satisfies the desired privacy level.

2.3 SYNTHETIC DATASETS

The modern era has given rise to an abundance of personal data, which is obtained from multiple sources like smartphones and medical devices. The utilization of such data for research purposes has become increasingly important. However, the privacy regulations in this context

make it difficult to ensure the fair and equitable usage of such data. While these regulations are of great importance from an ethical standpoint, they can result in data stored in isolated silos. This situation can subsequently impede research progress.

To overcome such situation, synthetic data sets have been proposed as a method for dissemination of data while protecting the privacy of individuals. These data sets have the potential to help researchers and industry understand our behavior better on both individual and collective levels, and have also allowed important research studies in many disciplines, including health, education, and economy.

The generation of synthetic data sets happens by replacing the observed data with synthetic values. The synthetic values are generated from models based on the original data. The risk of disclosure is reduced by replacing the original values with synthetic values.

2.3.1 Differentially Private Synthetic Data Generators

We use several differentially private (DP) synthetic data generators that have been specifically tailored for generating tabular data with the goal of enhancing their utility for learning tasks. We consider two broad categories of approaches: i) Marginal-based methods, ii) and Generative Adversarial Network (GAN) based models.

2.3.1.1 Marginal-based Methods

Multiplicative Weights with Exponential Mechanism Algorithm (HARDT *et al.*, 2012). The MWEM algorithm takes as input a dataset $D \subseteq \mathcal{D}$ and a set of linear queries Q (e.g. counting queries).[1] The MWEM algorithm aims to produce synthetic data generation algorithm by learning a distribution A over $\mathcal{D}$ such that the answers to the queries q in Q when run over A are similar to when run over D, i.e. the difference between $q(A)$ and $q(D)$ should be small.

This is achieved by repeatedly sampling a query based on it's approximation score, selecting the query with the highest score. The approximation score measures, for a query q how distant

[1] A linear query q is a function that maps data records in $\mathcal{D}$ to the interval $[-1, +1]$. By extension, the answer of a linear query q on a dataset D is defined as $q(D) = \sum_{x \in \mathcal{D}} q(x) \cdot D(x)$.

$q(A)$ is from $q(D)$. The greater the distance, the highest the score.

The algorithm then updates the weight that A places on each record x with the Multiplicative Weights update rule to better approximate the distribution of D w.r.t. q.

MWEM satisfies ϵ-DP by leveraging the exponential mechanism for query selection, and the Laplace mechanism to perturb the query results.

MWEM PGM (MCKENNA *et al.*, 2019) is a variation of the multiplicative weights with exponential mechanism algorithm (MWEM), which is an algorithm that generated synthetic data based on linear queries. The algorithm aims to produce a data distribution that produces query answers similar answers resulted when querying the real dataset. The MWEM PGM variation combines probabilistic graphical models (PGMs) with the MWEM algorithm. The structure of the graphical model is determined by the measurements, such that no information is lost relative to a full contingency table representation.

MST (MCKENNA *et al.*, 2021) is a synthetic data generation algorithm that acts selecting 2- and 3-way marginals for measurement. It combines one principled step, which is to find the maximum spanning tree (MST) on the graph where edge weights correspond to mutual information between two attributes, with some additional heuristics to ensure that certain important attribute pairs are selected, and a final step to select triples while keeping the graph tree-like.

AIM (MCKENNA *et al.*, 2022). The Adaptive and interactive mechanism (AIM) for synthetic data generation is a variation of the MWEM PGM algorithm that innovates in the way it selects the most useful measurements. The ability to produce data with lower error, in comparison to MWEM PGM, is because of the new proposed features in the select stage, which defines a quality score that helps determine the private selection of the next best marginal to measure. The quality score takes into account factors such as the current measure of the candidate marginal, expected improvement, relevance to the workload, and available privacy budget. The algorithm also includes other techniques like adaptive selection of rounds and budget-per-round, as well as intelligent initialization.

PrivBayes (ZHANG *et al.*, 2017). In order to improve the utility of the generated synthetic data, (ZHANG *et al.*, 2017) approximates the actual distribution of the data by constructing

a Bayesian network using the correlations between the data attributes. This allows them to factorize the joint distribution of the data into marginal distributions. Next, to ensure differential privacy, noise is injected into each of the marginal distributions and the simulated data is sampled from the approximate joint distribution constructed from these noisy marginals.

2.3.1.2 GAN-based Methods

Generative neural networks (GANs) are a type of artificial neural network used in machine learning for generating new data samples similar to a given training dataset. In the sense of game theory, generative adversarial networks are based on a game between two machine learning models, a discriminator model D and the generator G model. The goal of the generator is to learn realistic samples that can fool the discriminator, while the goal of the discriminator is to be able to tell generator generated samples from real ones (XIE *et al.*, 2018).

Conditional Tabular GAN (CTGAN) (XU *et al.*, 2019) is an approach for generating tabular data. CTGAN adapts GANs by addressing issues that are unique to tabular data that conventional GANs cannot handle, such as the modeling of multivariate discrete and mixed discrete and continuous distributions. It achieves these challenges by augmenting the training procedure with mode-specific normalization, employing a conditional generator and training-by-sampling that allows it to explore discrete values more evenly. When applying differentially private SGD (DP-SGD) (ABADI *et al.*, 2016) combined with CTGAN the result is a DP approach for generating tabular data.

The **PATE (Private Aggregation of Teacher Ensembles)** framework (PAPERNOT *et al.*, 2016) protects the privacy of sensitive data during training, by transferring knowledge from an ensemble of teacher models trained on partitions of the data to a student model. To achieve DP guarantees, only the student model is published while keeping the teachers private. The framework adds Laplacian noise to the aggregated answers from the teachers that were used to train the student models. CTGAN can provide differential privacy by applying the PATE framework. We call this combination PATE-CTGAN, which is similar to PATE-GAN for images (JORDON *et al.*, 2018). The original dataset is partitioned into k subsets and a DP teacher discriminator is trained on each subset. Further, instead of using one generator to

generate samples, k conditional generators are used for each subset of the data.

2.4 MACHINE LEARNING METRICS

Machine learning model evaluation by applying a trained model to data samples from a test data set. The test data set is an annotated data set, which means that the data contains the true values of the outcome variable. The evaluation is done by comparing the outputs of the machine learning model with the true values of the outcome variable.

We say that a model obtained a true prediction when the result of the machine learning model is equal to the true value. We say that a model obtained a false prediction when the result of the machine learning model is different from the true value of the outcome variable. For a binary machine learning model, predictions can be positive or negative. Positive predictions can be true positives (TP) or false positives (FP), and negative predictions con be true negatives (TN) or false negatives (FN).

Throughout this work we utilize several metrics to evaluate the proposed machine learning models. We define below the metrics utilized on this work.

2.4.1 Accuracy

The accuracy, acc, of a machine learning is given by:

$$acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.13}$$

2.4.2 Precision

The precision of a machine learning model is given by:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.14}$$

2.4.3 Recall

The recall of a machine learning model is given by:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.15}$$

2.4.4 Area under the curve - AUC

We utilize the area under the receiver operating characteristic curve as a metric for machine learning models. The receiving operating characteristic curve shows the performance of a classification model at all classification thresholds, by plotting the relationship between True Positive Rate (TPR) and False Positive Rate (FPR) for all classification thresholds.

$$\text{TPR} = \frac{TP}{TP + FN} \tag{2.16}$$

$$\text{FPR} = \frac{FP}{FP + TN} \tag{2.17}$$

The AUC measures the two-dimensional area underneath the ROC curve.

2.5 MACHINE LEARNING FAIRNESS

The term bias is often used to refer to demographic disparities in algorithmic systems that are objectionable for societal reasons. We will minimize the use of this sense of the word bias in this document, since different disciplines and communities understand the term differently, and this can lead to confusion. We refer to such disparities as algorithmic fairness.

We present the definition of two different algorithmic fairness metrics: Equal Opportunity (HEIDARI *et al.*, 2019) and Statistical Parity(BAROCAS *et al.*, 2017b). Given a dataset $W = (X, Y', C)$ with binary protected attribute C (e.g. race, sex, religion), remaining decision variables X and predicted outcome Y', we define Equal Opportunity and Statistical Parity as follows.

2.5.1 Equal Opportunity

Equal Opportunity (or Equality of Odds) requires equal True Positive Rate (TPR) across subgroups:

$$\Pr(Y' = 1|Y = 1,C = 0) = \Pr(Y' = 1|Y = 1,C = 1) \tag{2.18}$$

where Y' is the model output.

2.5.2 Statistical Parity

Statistical Parity requires positive predictions to be unaffected by the value of the protected attribute, regardless of true value of the outcome variable Y

$$\Pr(Y' = 1|C = 0) = \Pr(Y' = 1|C = 1), \tag{2.19}$$

We follow the approach of (XU *et al.*, 2021; PERRONE *et al.*, 2020) and utilize difference in Equal Oportunity (DEO) $= |\Pr(Y' = 1|Y = 1,C = 0) - \Pr(Y' = 1|Y = 1,C = 1)|$ and difference in Statistical Parity (DSP) $= |\Pr(Y' = 1|C = 0) - \Pr(Y' = 1|C = 1)|$ to measure model fairness.

2.6 SECURE MULTIPARTY COMPUTATION (MPC)

MPC protocols enable a set of parties to jointly compute the output of a function over the private inputs of each party, without requiring any of the parties to disclose their own private inputs (CRAMER *et al.*, 2000). MPC protocols are designed to prevent and detect attacks by an adversary corrupting one or more parties to learn private information or to cause the result of the computation to be incorrect. The adversary can have different levels of adversarial power. In the *semi-honest* model, all parties (even corrupted parties) follow the instructions of the protocol, but the adversary attempts to learn private information from the internal state of the corrupted parties and the messages that they receive. MPC protocols that are secure against semi-honest or *"passive"* adversaries prevent such leakage of information.

In the *malicious* adversarial model, the corrupted parties can arbitrarily deviate from the protocol specification. Providing security in the presence of malicious or *"active"* adversaries, i.e. ensuring that no such adversarial attack can succeed, comes at a higher computational cost than in the passive case.

The protocols that we propose are sufficiently generic to be used in settings with passive or active adversaries. This is achieved by changing the underlying MPC scheme to align with the desired security setting. We consider an honest-majority 3-party computing setting out of which at most one party can be corrupted (3PC) (ARAKI *et al.*, 2016; DALSKOV *et al.*, 2021), an honest-majority 4-party computing setting with one corruption (4PC) (DALSKOV *et al.*, 2021), and a dishonest-majority 2-party computation setting where each party can only trust itself (2PC) (CRAMER *et al.*, 2018). All these MPC schemes are based on secret sharing, where a secret input is split into shares that individually reveal no information about the original secret, but, when combined, can be used to recover the input. In secret sharing based MPC, the protocol's inputs are split into secret shares and these are distributed to the set of computing parties that run MPC protocols. Computations are performed over these secret shares, in our case to generate synthetic data. As all computations are done over the secret shared values, the servers do not learn the values of the inputs nor of intermediate results, i.e. MPC provides *input privacy*.

2.6.1 Secure Multiparty Computation Protocols

In the MPC schemes used in Chapter 5, all computations are done on integers modulo q, i.e., in a ring $\mathbb{Z}_q = \{0,1,\ldots,q-1\}$, with q a power of 2. As is common in MPC, any input real values from the data holders are converted to integers using a fixed-point representation (CATRINA; SAXENA, 2010). Below we give a high level description of the 3PC schemes used in this work. For more details and a description of the other MPC schemes, we refer to the papers about 2PC (CRAMER *et al.*, 2018), 3PC (ARAKI *et al.*, 2016; DALSKOV *et al.*, 2021), and 4PC (DALSKOV *et al.*, 2021).

Replicated sharing (3PC). In a replicated secret sharing scheme with 3 servers (3PC), a value x in $\mathbb{Z}_q$ is secret shared among servers (parties) S_1, S_2, and S_3 by picking uniformly

random shares $x_1, x_2, x_3 \in \mathbb{Z}_q$ such that $x_1 + x_2 + x_3 = x \mod q$, and distributing ($x_1$,$x_2$) to S_1, (x_2,x_3) to S_2, and (x_3,x_1) to S_3. Note that no single server can obtain any information about x given its shares. We use $[\![x]\!]$ as a shorthand for a secret sharing of x.

Passive security (3PC). The 3 servers can perform the following operations through carrying out local computations on their own shares: addition of a constant, addition of secret shared values, and multiplication by a constant. For multiplying secret shared values $[\![x]\!]$ and $[\![y]\!]$, we have that $x \cdot y = (x_1+x_2+x_3)(y_1+y_2+y_3)$, and so S_1 computes $z_1 = x_1 \cdot y_1 + x_1 \cdot y_2 + x_2 \cdot y_1$, S_2 computes $z_2 = x_2 \cdot y_2 + x_2 \cdot y_3 + x_3 \cdot y_2$ and S_3 computes $z_3 = x_3 \cdot y_3 + x_3 \cdot y_1 + x_1 \cdot y_3$. Next, the servers obtain an additive secret sharing of 0 by picking uniformly random u_1,u_2,u_3 such that $u_1 + u_2 + u_3 = 0$, which can be locally done with computational security by using pseudorandom functions, and S_i locally computes $v_i = z_i + u_i$. Finally, S_1 sends v_1 to S_3, S_2 sends v_2 to S_1, and S_3 sends v_3 to S_2, enabling the servers S_1, S_2 and S_3 to get the replicated secret shares (v_1,v_2), (v_2,v_3), and (v_3,v_1), respectively, of the value $v = x \cdot y$. This protocol only requires each server to send a single ring element to one other server, and no expensive public-key encryption operations (such as homomorphic encryption or oblivious transfer) are required. This MPC scheme was introduced by Araki et al. (ARAKI *et al.*, 2016).

Active security (3PC). In the case of malicious adversaries, the servers are prevented from deviating from the protocol and gain knowledge from another party through the use of information-theoretic message authentication codes (MACs). For every secret share, an authentication message is also sent to authenticate that each share has not been tampered in each communication between parties. In addition to computations over secret shares of the data, the servers also need to update the MACs appropriately, and the operations are more involved than in the passive security setting. For each multiplication of secret shared values, the total amount of communication between the parties is greater than in the passive case. We use the MPC scheme `SPDZ-wiseReplicated2k` recently proposed by Dalskov et al. (DALSKOV *et al.*, 2021) that is available in MP-SPDZ (KELLER, 2020).

MPC primitives. The MPC schemes listed above provide a mechanism for the servers to perform cryptographic primitives through the use of secret shares, namely addition of a constant, multiplication by a constant, and addition of secret shared values, and multiplication of secret shared values (denoted as π_{MUL}). Building on these cryptographic primitives, MPC

protocols for other operations have been developed in the literature. We use (KELLER, 2020):

- Secure random number generation from uniform distribution $\pi_{\mathsf{GR-RANDOM}}$: In $\pi_{\mathsf{GR-RANDOM}}$, each party generates l random bits, where l is the fractional precision of the power 2 ring representation of real numbers, and then the parties define the bitwise XOR of these l bits as the binary representation of the random number jointly generated.

- Secure random bit generation $\pi_{\mathsf{GR-RNDM-BIT}}$: In $\pi_{\mathsf{GR-RNDM-BIT}}$, each party generates the secret share of a single random bit, such that the generated bit is either 0 or 1 with a probability of 0.5.

- Secure equality test π_{EQ} : At the start of this protocol, the parties have secret sharings $[\![x]\!]$; at the end if $x = 0$, then they have a secret share of 1, else a secret sharing of 0.

- Secure less than test π_{LT} : At the start of this protocol, the parties have secret sharings $[\![x]\!]$ and $[\![y]\!]$ of integers x and y; at the end of the protocol they have a secret sharing of 1 if $x < y$, and a secret sharing of 0 otherwise.

- Secure greater than test π_{GT} : At the start of this protocol, the parties have secret sharings $[\![x]\!]$ and $[\![y]\!]$ of integers x and y; at the end of the protocol they have a secret sharing of 1 if $x > y$, and a secret sharing of 0 otherwise.

- Other primitives : We use secure maximum protocol (π_{MAX}), secure exponential protocol (π_{EXP}) and secure logarithm protocol (π_{LN}) as the building blocks for our protocols. π_{LN} uses the polynomial expansion for computing logarithm and π_{EXP} in turn uses the π_{LN} to compute exponential. π_{MAX} inherently uses the π_{GT} repeatedly over a list by employing variant of Divide-n-Conquer approach. At the start of all of these primitives, parties hold the secret sharings $[\![x]\!]$ and at the end of the protocol they hold the secret shares of the corresponding computed values.

MPC protocols can be mathematically proven to guarantee privacy and correctness. We follow the universal composition theorem that allows modular design where the protocols remain secure even if composed with other or the same MPC protocols (CANETTI, 2000).

2.6.1.1 Implementing DP in MPC.

Keeping in mind the dangers of implementing DP with floating point arithmetic (MIRONOV, 2012), we stick with the best practice of using fixed-point and integer arithmetic as recommended by, for example, OpenDP [2]. We implement all our DP mechanism using their discrete representations and use 32 bit precision to ensure correctness.

[2] https://opendp.org/

CHAPTER 3

ROBUST AND PRIVATE MACHINE LEARNING FOR CHILD SEXUAL ABUSE MEDIA DETECTION

Building machine learning systems for detecting CSAM media is a complex task. Due to the associated legal constraints, systems that rely on metadata for detecting and blocking the distribution of CSAM can expedite the hard work of NGOs and content moderators.

This chapter describes a framework for training and evaluating machine learning models for CSAM detection based solely on file metadata. Our framework provides guidelines for evaluating CSAM detection models against adversarial attacks and models' ability to perform on different data distributions

Our framework is general, and can be replicated by researchers and organizations. Our frameworks can also be utilized for machine learning tasks other than file path classification, such as website content classification, and search terms classification. We list our contributions as follows:

- We propose a framework for evaluating machine learning models for CSAM identification to prepare for deployment. Our framework, illustrated in Figure 3.1, proposes a testing pipeline that covers real-world scenarios that should be expected when deploying a machine learning model for CSAM detection: (i) test on CSAM and non-CSAM samples; (ii) test on adversarially modified CSAM samples to evade detection; (iii) test on benign samples from open data sources.

- We train and compare several machine-learning models that analyze file paths and file names from file storage systems and determine a probability that a given file has child sexual abuse content. Our experiments include traditional machine learning algorithms, deep neural network architectures, and Transformers-based models. We train our models on a real-world data set containing over one million file paths from apprehended hard

drives during investigations. It is the most extensive file path data set composed solely of file paths from apprehended hard drives. Our best classifier achieves recall rates over 0.94 and accuracy over 0.97 on holdout sets; it maintains a high recall rate in adversarially modified inputs; when tested against benign samples from other data distributions, it achieves a false-positive rate of ≈ 0.01.

To our knowledge, our work is the first to propose a framework for the evaluation of CSAM detection systems that include adversarial examples in the evaluation stage. Our results show that machine learning based on file paths can effectively detect CSAM in storage systems and achieve the aspired performance in all the proposed evaluation scenarios.

Finally, we remark that our solution has been deployed by the non-profit Project VIC[1] and it is currently in use.

3.1 RELATED WORK

Identification of CSAM via statistical algorithms is a reasonably recent approach. In the early 2000s, the US and the UK introduced laws targeting the online exploitation of minors (COPA in the US, Crime and Disorder Act UK)(DAVIDSON; GOTTSCHALK, 2010). However, only in 2008 the first widely used technology for CSAM identification was released.

PhotoDNA Hash. PhotoDNA Hash (PDNA) is a widely used technique for automated identification of CSAM. The PDNA uses a fuzzy hash algorithm to convert a CSAM image to a long string of characters. The converted hashes are compared against other hashes to find identical or similar images. PDNA technology enabled a faster discovery of CSAM while protecting the victim's identity. This system is still one of the most widely used methods for detecting CSAM images worldwide. Search engines, social networks, and image-sharing services utilize databases of hashed CSAM images to eradicate harmful content from their platforms. PDNA is a signature-based technology; it recalls only known CSAM. Therefore, identifying new CSAM in a PDNA-based system requires manual labeling.

Machine Learning for Image Identification. Since PDNA's first development, compu-

[1] A non-profit organization whose technologies are used by thousands of law enforcement officers worldwide - https://www.projectvic.org/vic-point

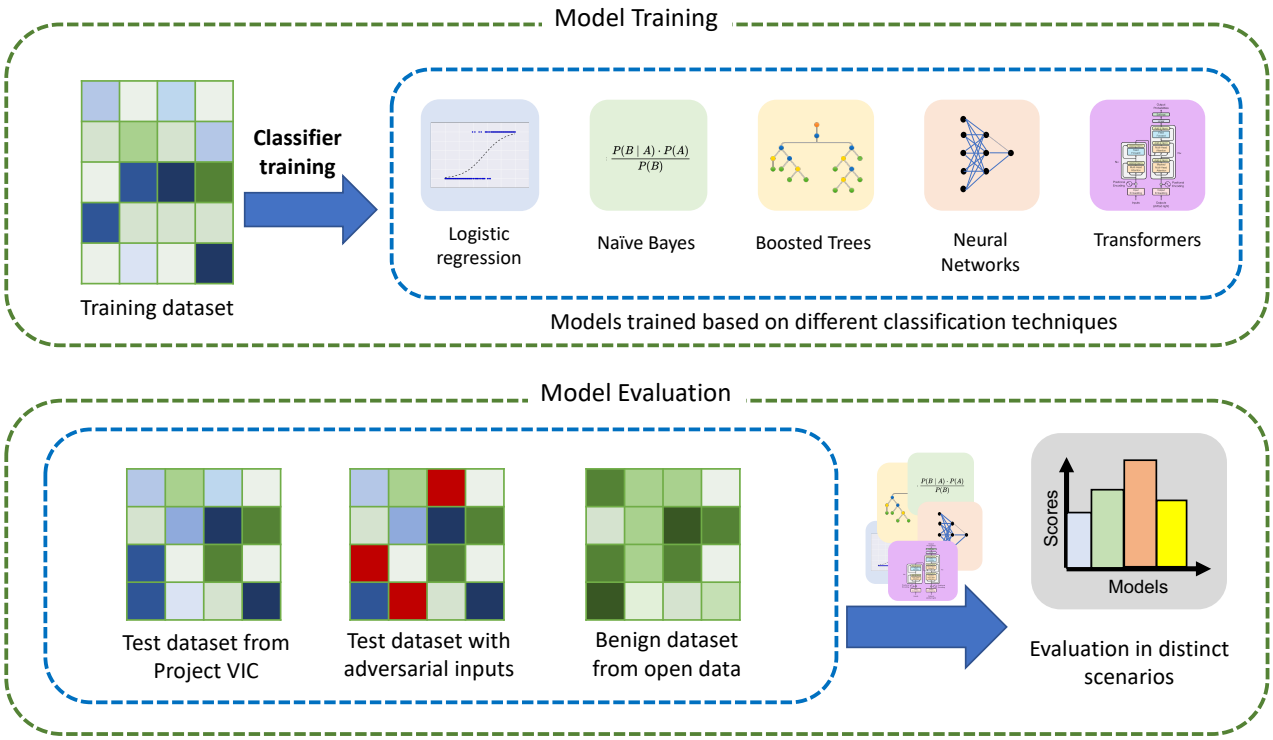

Figure 3.1: Pipeline for model training and evaluation of machine learning models for CSAM detection. (i) During model training, we train models utilizing several machine learning techniques, such as logistic regression, Naive Bayes, boosted trees, and deep neural networks, including Transformers. (ii) We construct different testing data sets to model performance in different circumstances of practical relevance during the model evaluation. We propose a testing framework where the model is tested under three scenarios: File paths from out-of-sample hard drives, file paths intentionally modified by an adversary to evade detection, and file paths from benign sources (open data).

ter vision models have undergone a revolution resulting in novel machine learning-based models for pornography and CSAM detection (NIAN *et al.*, 2016; MACEDO *et al.*, 2018; Yiallourou *et al.*, 2017; PEERSMAN *et al.*, 2014). The current approaches either combine a computer vision model to extract image descriptors (VITORINO *et al.*, 2016), train computer vision models on pornography data (GANGWAR *et al.*, 2017), perform a combination of age estimation and pornography detection (MACEDO *et al.*, 2018) or synthetic data (Yiallourou *et al.*, 2017). However, due to legal restrictions in maintaining a database of CSAM images, all current works are based on either unrealistic images (Yiallourou *et al.*, 2017), or validated by authorities in small data sets (VITORINO *et al.*, 2016; MACEDO *et al.*, 2018; GANGWAR *et al.*, 2017) that hardly represent the true data distribution in the internet (BURSZTEIN *et al.*, 2019).

Adversarially Modified Data Samples. Adversarial inputs are intentionally crafted small perturbations to elude detection from a model. For text applications, this can include injecting random noise that does not dramatically alter the understanding by a human. Substitutions such as replacing "before"with "b4", homoglyph substitutions, and other substitutions, such as using "Lo7ita"instead of "Lolita"(WOODBRIDGE *et al.*, 2018). The effects of adversarial modifications in text classification have been explored for different natural language processing (NLP) techniques, including classification (AGARWAL *et al.*, 2007), machine translation (BELINKOV; BISK, 2017) and word embeddings (HEIGOLD *et al.*, 2017). Depending on what kind of information is available to the adversary, it distorts portions of the text most likely to contain a signal important to the classification task.

CSAM File Metadata Classification. While significant efforts have focused on the images themselves, some researchers have looked for complementary signals to help CSAM identification. Such measures include queries that return CSAM in search engines, file metadata, and conversations that imply grooming or exchange of CSAM (THORN,). Other efforts have used textual signals to identify where CSAM might be located, such as keywords related to website content(Westlake *et al.*, 2012), using NLP analysis (PEERSMAN *et al.*, 2016; PEERSMAN, 2018; NABKI *et al.*, 2020; AL-NABKI *et al.*, 2020; PANCHENKO *et al.*, 2013; DU; SCANLON, 2019), conversations(BOGDANOVA *et al.*, 2014). Our work falls into this category. Previous works have found that perpetrators use a specific CSAM vocabulary to name files (PEERSMAN *et al.*, 2016). For this reason, using file paths, which is the combination of

the file location and file name, is a promising approach for CSAM identification.Other related works aim to identify CSAM based solely on file path (NABKI *et al.*, 2020; AL-NABKI *et al.*, 2020). However, these works do not address important questions such as classifiers' robustness against adversarial examples and performance in out-of-sample benign data sets.

Recent Works. The recent publication of a survey on detecting and preventing online child sexual abuse material (NGO *et al.*, 2022) compares over 35 studies on the topic. The findings of this survey highlight the problem's complexity and the need to combine computer vision and natural language processing techniques to combat this heinous crime. In (SILVA *et al.*, 2022), the authors propose a pipeline for extracting signals from data, which they claim is highly valuable in highlighting essential aspects of the overall distribution of data. This pipeline can provide valuable insights into databases that cannot be disclosed. Moreover, in (WOODHAMS *et al.*, 2021), the authors used real-world data from the United Kingdom to study the behavior and preferences of 53 anonymous CSAM suspects who were active on the dark web and noticed by the police. This research provides a unique perspective into the minds of these dangerous individuals and can be used to develop more effective strategies to prevent them from exploiting innocent children online.

3.2 METHODS

In this section we describe the data set, methods, and algorithms utilized in our experiments.

3.2.1 Training Data Set

Our supervised learning approach to identify CSAM file paths utilizes a binary labeled data set. To separate the data set into independent training and test sets, we split the data by storage system information (e.g., driver designations) in order to not leak information from the training to test set, which is also known as model leakage (KAUFMAN *et al.*, 2012). Our data set consists of real file paths collected by Project VIC International[2]. The data consists of 1,010,000 file paths from 55,312 unique storage systems. File paths are strings that contain location information of a file (folders) in a storage system and the file name. In Table 3.1, we

[2]https://www.projectvic.org.

present details on the different types of content that constitute the data set and the number of samples for each type.

Table 3.1: Project VIC data set description. The data set is used for model training and model testing. It contains non-pertinent (label 0) file paths and different types of file paths of child exploitative and child sexual abuse material (label 1).

Content Type	Samples	Label
Non-pertinent	717,448	0
Child exploitative and child sexual abuse	292,552	1

The Project VIC data set used in our experiments contains 717,448 non-pertinent file paths and 292,552 file paths containing child exploitative and child sexual abuse material. We note that all the 1,010,000 file paths in our data set were extracted from hardware apprehended for investigations. The training and testing data sets accurately represent the data in a deployment scenario.

3.2.1.1 File Path Characteristics

The distribution of file path length helps us define the size of the character embedding vectors in our deep neural networks models and the size of the word vectors used as input to the transformers-based model we fine-tune to the task of CSAM file path identification.

When analyzing the distribution of file path lengths in the data set, we observe that 95% of file paths have 300 characters or less. Limiting the size of the character embedding layer helps increase time and memory efficiency of the model. We set our character embedding layer size to 300 characters. For file paths with more than 300 characters, we truncate the file path by discarding the initial characters and keeping only the last 300 characters. We pad with zeros on the left for file paths with less than 300 characters.

The transformers-based model also takes as input a fixed-sized vector, in this case, a vector of words. We consider a word to be a sequence of alphanumeric characters that are separated by a dash, slash, colon, underscore, or period. By counting the number of words in each file path, we see that over 99% of file paths have at most 64 words. More precisely, in our data set,

there is only one file path with more than 60 words. For this reason, we set the input vector size for the transformers-based model to 64 words.

3.2.1.2 Cross Validation Data Split

We use a K-Fold Cross Validation methodology in our experiments with K=10. The file path contains information about the storage system in our data set. If a storage system contains a high volume of CSAM files, the model could learn that files from certain storage systems have a high probability of being CSAM. This is known as model leakage (KAUFMAN *et al.*, 2012). Leakage in machine learning modeling consists of introducing information about the target of a machine learning problem at training time. To avoid model leakage, we split the data by storage system information. In each cross-validation fold is the data is divided into 80% for training, 10% for validation, and and 10% for testing.

The information before the first backlash of a file path specifies the external storage system or a laptop/desktop. This information is used to partition the data set for cross-validation.

3.2.2 Text Vectorization

We present the concepts utilized for text vectorization: term-frequency inverse-document-frequency (TF-IDF), character-based quantization, and word vectors that will serve as input to the transformers-based model.

3.2.2.1 TF-IDF

This technique attributes weights to words (or sequences of characters) in a text (JONES, 1972). First, it computes the term-frequency (TF), which is the number of times a term occurs in a given document. The inverse-document frequency-component (IDF) is computed as:

$$\mathrm{IDF}(t) = \log \frac{1+n}{1+df(t)} + 1. \tag{3.1}$$

Where n is the total number of documents in the document set, and $df(t)$ is the number of documents in the document set that contains the term. For each term, the product of the TF

and IDF components is computed. The resulting TF-IDF vectors are then normalized by the Euclidean norm.

When vectorizing a text with TF-IDF, the terms in a text can be words or sequences of characters. We investigate both approaches in our work. When using *words* as terms in TF-IDF, we refer to the text vectorization as bag-of-words (BoW). When vectorizing the text as BoW, for each file path, we consider a *word* to be a sequence of alphanumeric characters that are separated by a dash, slash, colon, underscore or period. The bag-of-words model is constructed by selecting the 5,000 most frequent *words* from the training subset. We utilize this text representation in combination with TF-IDF. The data set of vectorized file paths is used as input to three different learning algorithms: logistic regression, naive Bayes, and boosted decision trees.

When using sequence of characters as terms in TF-IDF, we refer the text vectorization as bag-of-n-grams, or simply n-grams. When vectorizing the text as n-grams, we extract from each file path string its n-grams, for $n \in \{1,2,3\}$. The set of n-grams of a string s is the set of all substrings in s of length n. We construct the bag-of-n-grams models by selecting the 50,000 most frequent n-grams (up to 3-grams) from the training data set. We utilize this text representation in combination with TF-IDF. The data set of vectorized file paths is used as input to three different learning algorithms: logistic regression, naive Bayes, and boosted decision trees.

3.2.2.2 Character-based Quantization

This type of text representation defines an alphabet of size m as the input language, quantized using 1-of-m encoding. Each textual input, of length l, is then transformed into a sequence of such m-sized vectors with fixed length l. Texts with more than l characters are truncated, and the exceeding initial characters are discarded. If the text is shorter than l, it is padded with zeroes on the left. Characters that are not in the alphabet are quantized as all-zero vectors. The alphabet used in our models consists of $m = 802$ characters, including English letters, Japanese characters, Chinese characters, Korean characters, and special alphanumeric characters. The alphabet is the set of all unique characters in the training data.

3.2.2.3 Word Vectors for Pre-trained Models

We utilize transformers-based models in our experiments. We utilize bidirectional encoder representation from transformers, BERT (DEVLIN *et al.*, 2018) pre-trained model. To prepare the text to serve as an input to the pre-trained BERT model, we represent the file path as a sequence of words by removing dash, slash, colon, underscore, and periods. We limit the sequence of words to 64 as indicated in section 3.2.1.1.

3.2.3 Learning Algorithms

We use several learning algorithms that have been successfully applied to short text classification. We consider two broad approaches: i) Traditional machine learning models, ii) and Neural networks models.

3.2.3.1 Traditional ML on Extracted Features

Logistic Regression. This classification algorithm is a discriminative classifier that models the posterior probability $P(Y|X)$ of the class Y given the input features X by fitting a logistic curve to the relationship between X and Y. Model outputs can be interpreted as probabilities of the occurrence of a class (NG; JORDAN, 2001).

Naive Bayes. Conditional probability model that assumes independence of features: given a problem instance to be classified, represented by a vector $\mathbf{x} = (x_1, \ldots, x_n)$ representing some n features, it assigns to this instance probabilities $P(C_k \mid x_1, \ldots, x_n)$ for each of K possible outcomes or classes C_k. The problem with the above formulation is that if the number of features n is large or if a feature can take on a large number of values, then basing such a model on probability tables is infeasible. The model is reformulated to become more tractable. Using Bayes' theorem and assuming independence of the feature variables, the conditional probability can be decomposed as:

$$P(C_k \mid \mathbf{x}) = \frac{P(C_k)\ P(\mathbf{x} \mid C_k)}{P(\mathbf{x})}$$

Boosted Decision Trees. Model based on ensembles of trees, where each tree is trained using a boosting process in which each subsequent tree is built with weighted instances which

were misclassified by the previous tree (FREUND; SCHAPIRE, 1997). Classification of a new instance with a trained ensemble of trees is based on a simple majority vote of the individual trees.

3.2.3.2 Deep Neural Networks on Learned Embeddings

All neural network architectures start with an embedding layer that represents each character by a numerical vector. The embedding maps semantically similar characters to similar vectors, where the notion of similarity is automatically learned based on the classification task at hand. The variant of long short-term memory (LSTM) network architecture used in our work is the common "vanilla"architecture as used in (WOODBRIDGE *et al.*, 2016).

Convolutional Neural Networks. One-dimensional Convolutional Neural Networks (CNNs) are a good fit when the input is text, treated as a raw signal at the character level (ZHANG *et al.*, 2015). The CNN automatically learns filters to detect patterns that are important for prediction. The presence (or lack) of these patterns is then used by the quintessential neural network (multilayer perceptron) to make predictions. These filters, (also called kernels) are learned during backpropagation. Figure 3.2 shows detailed information on the CNN-based architecture used in our experiments. The architecture in Figure 3.2 includes two one-dimensional convolution layers, followed by a dense layer. The dimensions of the CNN architecture presented were tuned during the validation phase of the training experiments.

Long Short-Term Memory Networks. This flexible network architecture generalizes manual feature extraction via n-grams by learning dependencies of one or multiple characters, whether in succession or with arbitrary separation. The long short-term memory network (LSTM) layer can be thought of as an implicit feature extraction instead of explicit feature extraction (e.g., n-grams) used in other approaches. Rather than represent file paths explicitly as a bag of n-grams, for example, the LSTM learns patterns of tokens that maximize the performance of the second classification layer. Figure 3.3 shows detailed information on the LSTM-based architecture used in our experiments, including data dimensions and weights in each layer. The dimensions of the LSTM architecture presented were tuned during the validation phase of the training experiments.

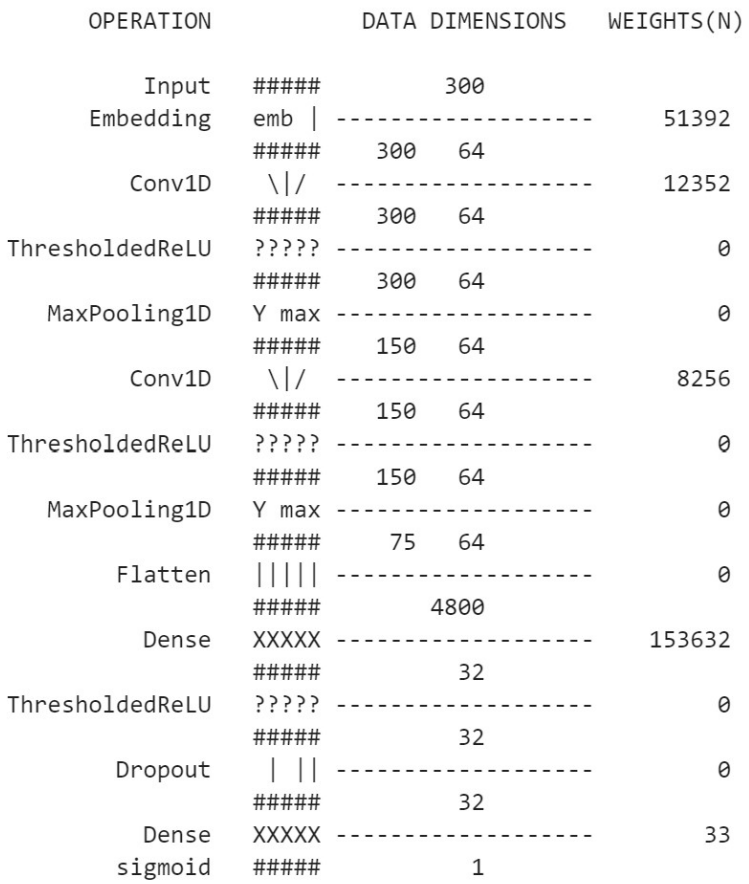

Figure 3.2: Diagram of the deep neural network architecture with CNN layers used for training one of our CNN-based model. All data dimensions and number of weights in each layer of our CNN model are indicated in the above diagram.

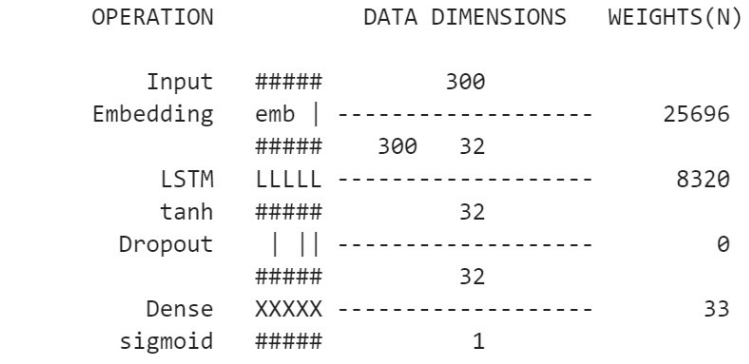

Figure 3.3: Diagram of the deep neural network architecture with LSTM layer used for training one of our LSTM-based model. All data dimensions and number of weights in each layer of our LSTM model are indicated in the above diagram.

Transformer-based model. Transformer (VASWANI *et al.*, 2017) is a model architecture that dismisses recurrence and relies solely on an attention mechanism to derive global dependencies between input and output. BERT's model architecture is a multilayer bidirectional Transformer (DEVLIN *et al.*, 2018). BERT's framework comprises two steps: pre-training and fine-tuning. Pre-trained BERT models are trained on unlabeled data over different pre-training tasks and can be easily fine-tuned to several downstream tasks. We utilize pre-trained `bert-base-uncased` from the Hugging Face Transformer library [3]. We use `BertForSequenceClassification` class from the same library for fine-tuning, which is the downstream task suitable for our classification problem. We fine-tune the BERT model using the hyperparameters suggested by (DEVLIN *et al.*, 2018).

3.2.4 File Path-Based CSAM Classifiers

Our work investigates four approaches for CSAM file path classification.

1 **Bag-of-words.** This approach encodes the file path string into a vector of words. The weights of the words are attributed using TF-IDF. We utilize the resulting vectors as input to traditional machine learning classifiers (logistic regression, boosted decision trees, and Naive Bayes).

2 **Character N-grams.** A list of character sequences on size N encodes the file path. The weights of the sequences are attributed using TF-IDF. The resulting vectors of the character sequences are used with traditional machine learning classifiers (logistic regression, boosted decision trees, and Naive Bayes);

3 **Character-based Neural Networks.** Sequences of encoded characters are used with a convolutional neural network (CNN) and a long short-term memory network (LSTM).

4 **Pre-trained BERT Model.** Pre-trained `bert-base-uncased` model is fine-tuned for downstream sequence classification task.

Table 3.2: Model evaluation. Experiments with traditional machine learning and neural networks using Project VIC's data set. We evaluate the AUC-ROC, accuracy, precision, and recall. These results were measured across 10-folds in a cross-validation setting. For each metric, we report the mean (μ), and the standard deviation (σ). We highlight the best results, which were achieved by the character-based CNN.

Model	**AUC**		**Accuracy**		**Precision**		**Recall**	
	μ	σ	μ	σ	μ	σ	μ	σ
BoW Logistic Reg.	0.967	0.035	0.922	0.062	0.904	0.090	0.787	0.202
BoW Naive Bayes	0.972	0.011	0.927	0.032	0.875	0.070	0.859	0.114
BoW Boosted Trees	0.982	0.013	0.934	0.062	0.903	0.096	0.827	0.203
N-grams Logistic Reg.	0.980	0.021	0.931	0.060	0.919	0.088	0.793	0.202
N-grams Naive Bayes	0.958	0.023	0.929	0.032	0.839	0.083	0.913	0.085
N-grams Boosted Trees	0.983	0.015	0.931	0.060	0.906	0.094	0.822	0.203
Character-based CNN	**0.990**	0.011	**0.968**	0.019	**0.938**	0.034	**0.943**	0.060
Character-based LSTM	0.982	0.017	0.953	0.044	0.937	0.064	0.903	0.129
BERT	0.987	0.013	0.955	0.035	0.934	0.048	0.896	0.100

3.3 MODEL EVALUATION

We present our results for all our classifiers in Table 3.2. All performance metrics were measured using a 10-fold cross-validation on the Project VIC data set.

For each of our classifiers, we report the mean and the standard deviation over the folds for the area under the ROC curve (AUC), accuracy, precision, and recall for predicting CSAM files. We focus on two primary metrics for model comparison: Recall and AUC. Additionally, we assess all machine learning models' generalization by looking into the standard deviations over the cross-validation folds.

[3]https://huggingface.co/bert-base-uncased

3.3.1 Traditional Machine Learning Models

There are significant advantages of traditional machine learning models compared to deep neural networks. Understanding how well these models perform can help scientists and investigators leverage such models' most remarkable characteristic: feature interpretability. The most relevant predictive tokens, or n-grams, can give clues about vocabulary words in the data set and utilize them in other CSAM detection systems. Table 3.2 shows that the model trained with bag-of-words and bag-of-n-grams operates in similar AUC and accuracy ranges. When analyzing recall rates of traditional models, we note that both naive Bayes models have the highest average rates and lowest standard deviations. The naive Bayes with bag-of-n-grams features presents the best recall of all traditional models, of about 0.91. Among the other models trained using bag-of-n-grams, naive Bayes presents a much smaller recall standard deviation ($\sigma = 0.085$) when compared to logistic regression ($\sigma = 0.20$) and boosted decision trees ($\sigma = 0.20$).

Although the evaluation of CSAM classification models heavily relies on recall rates, when deploying a model in an environment that potentially analyzes hundreds on thousands of file systems, and consequently millions of file paths, precision becomes a significant metric. The burden of having several thousands of false positives can result in an inefficient process and potentially delay investigations and the discovery of true positives. The AUC metric captures the ability of a classifier to operate with high recall when low false positive rates are necessary. By analyzing the traditional models' AUC, we observe that boosted decision trees perform better than the two other techniques.

3.3.2 Deep Neural Networks and Transformers-based Models

We achieved the best performance across all categories with deep neural network architecture. We trained three different architectures: a layered CNN, an LSTM-based model, and BERT. The LSTM model achieves results very similar to the fine-tuned BERT model. Both models present accuracy above 0.95, precision over 0.93, and recall ≈ 0.9. However, our CNN model consistently outperforms all the other models in mean performance metrics across all folds and in the lowest standard deviation.

3.3.3 Comparison with Previous Works

Although several previous works have proposed using file paths for CSAM detection, they lack rigorous methodology in which test data correctly emulates the data during deployment. Without a test data set with a similar distribution to the data during deployment, it is hard to evaluate the true model performance. In (AL-NABKI *et al.*, 2020), although the authors did achieve a recall rate of 0.98, the training and testing data utilized in the experiments are not an accurate representation of data in a deployment scenario. CSAM and non-CSAM file paths come from entirely different data sources and do not accurately represent file paths' data distribution in real deployment scenarios. In (DU; SCANLON, 2019), the authors propose a sound methodology for collecting CSAM and non-CSAM file paths from a pool of Windows disk images. However, accuracy, precision, and recall rates of 1.00 indicate that model leakage and data set size might be leading to model overfitting. A third work focus on detecting CSAM (PANCHENKO *et al.*, 2013) using filenames. Once again, it does not provide a testing scenario that reasonably represents a deployment setting. The work explores only traditional machine learning techniques achieving a maximum accuracy of 0.97.

3.4 MODEL EVALUATION WITH ADVERSARIAL EXAMPLES

In classical machine learning applications, we assume that the underlying data distribution is stationary at test time. However, a testing pipeline of models aimed at the detection of illegal activities should anticipate an intelligent, adaptive adversary actively manipulating data. We know perpetrators purposely add typos and modifications to file identifiers (PEERSMAN *et al.*, 2016) to evade blocklists and machine learning-based detection mechanisms. We modify our test data set to simulate an adversary actively changing the file paths to elude the classifiers.

Our CSAM file path detector assumes that file paths contain information about file contents; therefore, we can detect CSAM files by only analyzing file paths. This is because CSAM files are often shared among perpetrators, and the file name is usually used to identify file contents in many scenarios, including peer-to-peer systems (LATAPY *et al.*, 2013; FOURNIER *et al.*, 2014). In such case, the adversary wants to make the maximum possible changes without compromising others' ability to search the file.

The proposed adversarial analysis certainly has limitations, as criminals could adopt alternate coding methods or randomly name their CSAM files to evade detection. However, the alarming reality is that the online search for CSAM content has seen a significant surge in recent years, with CSAM content being openly advertised online [4]. Given the widespread nature of this issue, it is reasonable to assume that adversaries will opt for subtle modifications and obfuscation in the file paths rather than completely concealing them. Our experiments incorporate the most commonly observed adversarial techniques in evading text-based classification, as documented in previous works (KUCHIPUDI *et al.*, 2020; DAWSON, 2022) . It is important to acknowledge that this is not an exhaustive list of possible attacks, but it provides a solid foundation for understanding the threat landscape.

3.4.1 Threat Model

In our threat model, the attacker is unaware of the model architecture and parameters and does not have access to the confidence scores predicted by the model. The attacker attempts to cause an integrity violation in the model by modifying the input under bounded perturbation size (GOODFELLOW *et al.*, 2014). The only knowledge the adversary has about the model is the input space and the output space. Notably, the attacker cannot immediately observe the output for a given input, so cannot directly optimize for a worst-case outcome. But, since the attacker generally understands that filenames are being monitored, she uses heuristics, randomly applied, that attempt to evade.

We assume the adversary has access to a list of CSAM and non-CSAM trigger words, similar to the adversarial attacks proposed in (KUCHIPUDI *et al.*, 2020) to evade spam email detection models (see Figure 3.4 for a comparison between an adversary with access trigger words and an adversary without access). We create a list of trigger words, i.e., words that are highly correlated with CSAM and non-CSAM file paths using odds ratio. Odds ratio is a widely used technique in information retrieval which is used for feature selection and interpretation of text classification models (MLADENIĆ, 1998).

We calculate the odds of the keyword being part of a CSAM file path and the odds of the

[4] https://www.thorn.org/child-sexual-exploitation-and-technology/

keyword being part of a non-CSAM file path for all keywords. The Odds ratio of a word w is computed as:

$$\text{Odds Ratio} = \frac{\text{odds of } w \text{ appear in CSAM file}}{\text{odds of } w \text{ appear in non-CSAM}} \tag{3.2}$$

The CSAM lexicon comprises all keywords with an Odds Ratio greater than two. An analogous process identifies the non-CSAM trigger words. We assume the list of trigger words is available to the adversary.

3.4.2 Random Character Replacement

The adversarial examples are generated by randomly selecting a position in the file path string and substituting the character in the selected position with a random alphanumeric character. This technique has been previously used to attack language models (BELINKOV; BISK, 2017). We evaluate our models for three character replacement rates: 10%, 15%, and 20% of file path length.

3.4.3 Homoglyph Replacement

The homoglyph replacement attack (JÁÑEZ-MARTINO *et al.*, 2022) obfuscates words by modifying words in a text while keeping them readable. This attack replaces characters of CSAM trigger words in file paths with homoglyphs, i.e. a character with identical or very similar shapes. We utilize the homoglyph dictionary from (DAWSON, 2022) to make the character replacements.

3.4.4 Synonym Replacement

In (KUCHIPUDI *et al.*, 2020), authors presented this attack as an effective way to evade spam classification models. The synonym replacement attack finds trigger words in CSAM file paths and replaces them with a synonym. We use the natural language toolkit (NLTK)[5]

[5]NLTK is a platform for building Python programs to work with human language data. <https://www.nltk.org>

module to find word synonyms. This attack intends to modify the file path without modifying its meaning.

3.4.5 CSAM Word Spacing

In (KUCHIPUDI *et al.*, 2020), authors propose the spacing attack. The spacing attack adds spaces between characters of trigger words. In our attack, since file paths comprise our data, we add an underscore "_"between every character of words from the CSAM trigger word list. Intuitively, the text parser and n-gram sequences of the trigger words would not recognize the new modified word, while a human would still be able to read and recognize the keyword.

3.4.6 Non-CSAM Word Injection

Manipulating the file paths by adding words that are more likely to appear in non-CSAM file paths was adapted from the *ham word injection attack* presented in (KUCHIPUDI *et al.*, 2020). This attack consists in selecting a word from the non-CSAM trigger word list and injecting this word into the file path. We evaluate our models when injecting one, two and, three non-CSAM words in the file paths.

3.4.7 Experimental Results

We evaluate the impact of adversarial modifications in test samples on the model's performance. The adversarial modifications are done in the test fold of the cross-validation on the Project VIC data set. Details on how cross-validation is done are in section 3.2.1. We are interested in i) understanding which machine learning techniques are more robust when the data is adversarially modified at test time, and ii) how much the performance of the models changes. All attacks target only CSAM file paths; therefore, we only evaluate the variation in recall rates.

Under the random character replacement attack, an adversary randomly modifies a percentage of the file path by randomly selecting characters and replacing them with random

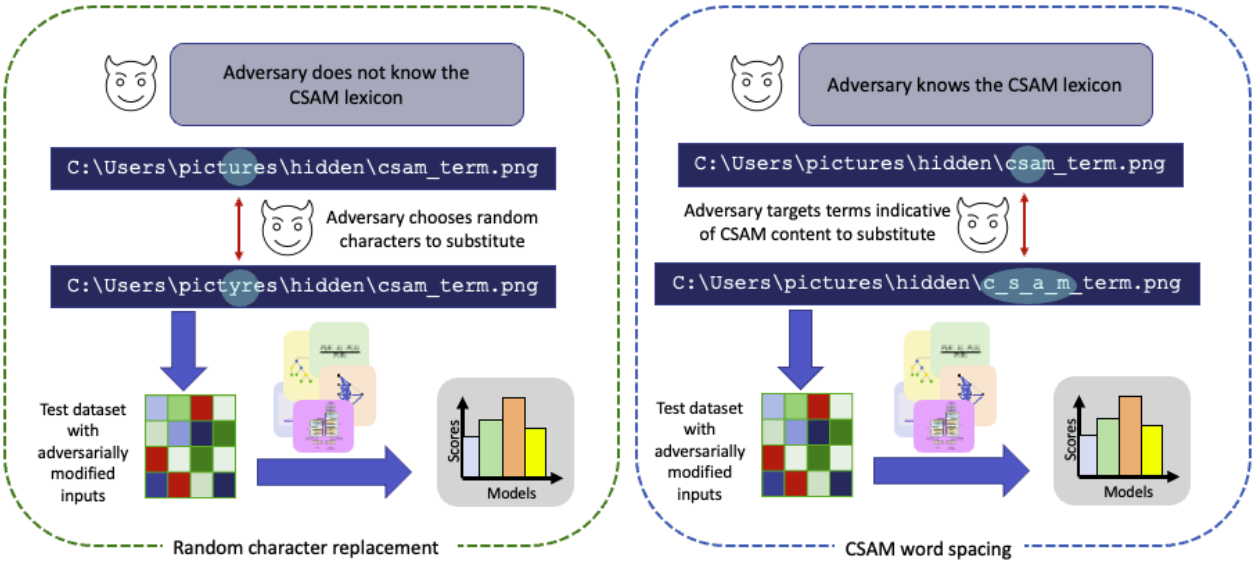

Figure 3.4: Example of adversarial inputs generation. We generate adversarial inputs based on several different adversarial attacks. Here we illustrate two attacks: (1) In the random character replacement attack, the adversary chooses random positions in the file path string and replace the character with a randomly chosen character. (2) The CSAM word spacing attack allows the adversary access to a CSAM lexicon. The adversary adds spacing between characters in words that are present in the CSAM lexicon.

characters. A reasonable adversary budget in this scenario is between 10% and 15%. Previous works have also considered this same percentage range for perturbing text strings (JIN *et al.*, 2019). Since most file paths have a length between 40 and 200 characters, this changes between 6 and 30 characters in each file path. To stress-test our models, we also analyze the performance of our models under a 20% change.

Table 3.3 demonstrates the variation in recall rates for different categories of adversarial attacks. In the presence of file paths generated via the character replacement attack, both variations of logistic regression models will present a small decay in recall rates when 10% of the file path is modified. We see the recall rates decrease as we increase the percentage of the file path that is modified. Logistic regression combined with bad-of-words text representation achieves a decrease of almost 25 points percentage in the recall rate, compared to the original recall, when 20% of the file path is modified.

Boosted tree models also see a sharp decrease in recall rates as we increase the percentage of modified file paths. When compared to the original recall rate of 0.83, a drop of 16 points percentage is observed in the lowest level of the attack, when 10% of the file path is modified. Recall rates get to 0.43 at the highest level of the attack when 20% of the file path is modified.

Table 3.3: Recall evaluation of model performance in the presence of adversarial examples. We evaluate changes in the recall rate of several machine learning models under the following attacks: random character replacement, homoglyph replacement, synonym replacement and CSAM word spacing. For all experiments, we report the average recall. We highlight the most robust results, which were achieved by the n-gram naive Bayes model.

	Character replacement					
Model	10%	15%	20%	homoglyph	synonym	spacing
BoW Logistic Regression	0.75	0.66	0.54	0.61	0.81	0.62
BoW Naive Bayes	0.85	0.83	0.80	0.81	0.90	0.81
BoW Boosted Trees	0.66	0.55	0.43	0.42	0.85	0.42
N-grams Logistic Regression	0.78	0.74	0.69	0.64	0.83	0.63
N-grams Naive Bayes	**0.92**	**0.93**	**0.94**	**0.88**	**0.92**	**0.88**
N-grams Boosted Trees	0.69	0.61	0.53	0.40	0.85	0.39
Character-based CNN	0.86	0.83	0.79	0.71	0.91	0.80
Character-based LSTM	0.82	0.79	0.76	0.76	0.90	0.77
BERT	0.85	0.83	0.82	0.84	0.91	0.77

At a recall rate of 0.43, more than half of the modified file paths are able to evade the classifier. CNN, LSTM, and BERT models also present decreases in recall rates as we increase the number of modifications in the file paths. The BERT model presents the smallest overall decrease of the three models. The naive Bayes model presented surprising results. The recall rates did not decrease for any level of the attack. In addition, we notice a small recall increase as the number of modifications in the file paths increased. We believe that the randomness added to the CSAM file paths helped the naive Bayes classifier identify better which file paths were CSAM.

The homoglyph attack is very common in scenarios where an adversary wants to evade spam email classifiers (DENG *et al.*, 2020). We evaluate the performance of all models to understand which models are more resilient to this kind of attack. Logistic regression and boosted trees models all see a significant reduction in recall rates. We observe that n-grams boosted tress

Table 3.4: Recall evaluation of model performance in the presence of adversarial examples. We evaluate changes in the recall rate of several machine learning models under the non-CSAM word injection attack. For all experiments, we report the average recall. We highlight the best results, which were achieved by the character-based CNN

	non-CSAM word injection		
Model	one word	two words	three words
BoW Logistic Regression	0.78	0.71	0.63
BoW Naive Bayes	0.80	0.68	0.58
BoW Boosted Trees	0.78	0.71	0.69
N-grams Logistic Regression	0.80	0.73	0.65
N-grams Naive Bayes	0.80	0.64	0.49
N-grams Boosted Trees	0.80	0.73	0.66
Character-based CNN	**0.89**	**0.86**	**0.84**
Character-based LSTM	0.87	0.84	0.81
BERT	0.83	0.74	0.65

recall rate achieve 0.40 under the homoglyph attack, which leads us to conclude that this model is susceptible to this kind of attack. CNN and LSTM models also present a decline in recall rate, going down to 0.71 and 0.74, respectively. Once again, the n-grams naive Bayes model is the best performing under attack, presenting a recall rate of 0.88, followed by BERT, which presented a recall rate of 0.84.

The synonym attack was the attack that overall impacted less the models' performance. Most models did knot see a decrease, and several models experienced a small increase in recall rates. The spacing attack impacted logistic regression and boosted trees, decreasing the recall rates of these models by more than 20 points. CNN, LSTM, and BERT suffered decreases in recall rates of approximately 10 points percentage. N-grams naive Bayes presented, once again, the smallest decrease in recall rates, with a decrease of 3 points percentage.

Non-CSAM word injection was the attack that most impacted the n-gram naive Bayes model, as presented in Table 3.4. By injecting only one word highly correlated with non-CSAM

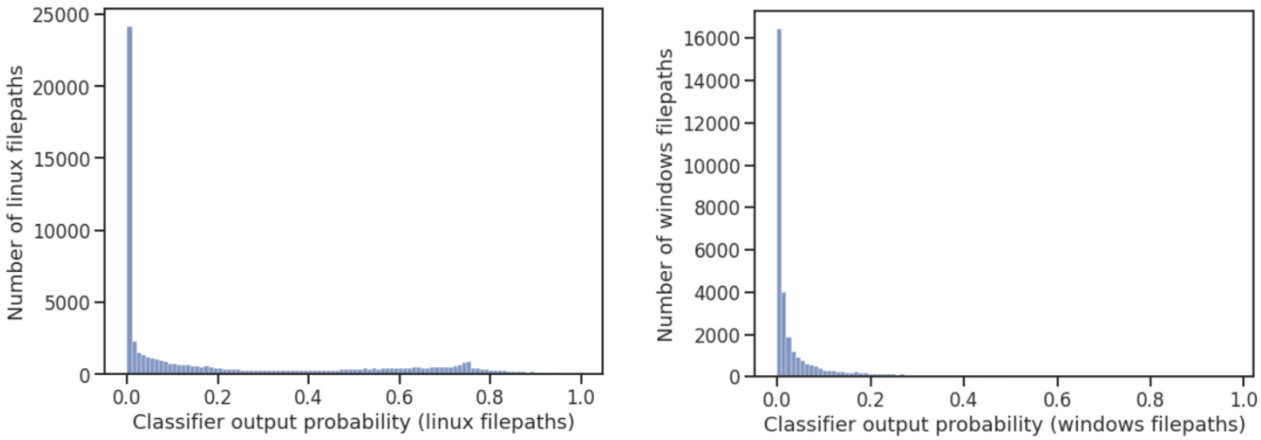

Figure 3.5: Model confidence scores when evaluating file paths from common crawl. Our best-performing model, a CNN-based model, exhibits a low number of files with a confidence score over 0.2. The false positive rate (FPR) is 0.03 for a confidence threshold of 0.5, but achieves an FPR of 0.002 for a confidence threshold of 0.9. The model presents a higher FPR on Linux file paths, where at a confidence level of 0.5 it exhibits an FPR of 0.24. However, it drops significantly for a higher confidence threshold, achieving an FPR of 0.008 at a confidence level of 0.9. At this confidence level, out of 73k file paths from the Linux set, only 584 would be identified as CSAM by our model. We highlight the most robust model, which is the n-grams naive Bayes model.

files, we already see a drop in recall rate to 0.80. As we increase the number of words, the recall rate decreases to 0.49 when injecting three non-CSAM words into a file path. Logistic regression, boosted trees, and BERT models also presented a reduction in recall rates with the addition of non-CSAM words. Even though we see a decrease in recall rates in all models, CNN and LSTM models presented the most stable behavior under this attack.

By exploring multiple attacks and evaluating the performance of different models under these attacks, we can identify which learning techniques produce models that are sensitive to the attacks, such as logistic regression and boosted trees, and which models are more resilient to attacks, such as n-gram naive Bayes and CNN models. We leave the improvement of the performance of models under non-CSAM word injection attacks as a future extension of this work.

3.5 MODEL EVALUATION WITH FILE PATHS FROM COMMON CRAWL

Our training data set perfectly represents the deployment scenario for model deployment: our model is currently used to identify CSAM files in apprehended hard drives. Our training

data comprises positive and negative examples from apprehended hard drives. The training data contains only file paths from hard drives that were *suspect* in the first place.

We collect file paths from common crawl data [6]. For this exercise, we assume all file paths from the common crawl are benign. However, understanding how the model behaves in other data distributions, i.e., data sets formed by file paths that come solely from benign sources and not *suspect* hard drives, is also essential. Given that we only evaluate the model performance on benign file paths, the metric we use for evaluating is the false positive rate (FPR). When a positive detection occurs, detection systems trigger an action, usually human data review. Suppose the false positive rate is poorly understood and the model operation is not correctly calibrated. In that case, it can cause a burden on content moderators and jeopardize the deployment of the system.

3.5.1 Common Crawl data set

The benign samples data set was constructed using the publicly available common crawl data set. We collected data from the Common Crawl index CC-MAIN-2021-10. The WARC (Web ARChive) files utilized to construct our data set are:

- Linux file paths: we parsed the first 200 WARCs (00000-00199, inclusive) resulting in over 73k unique paths.
- Windows file paths: we parsed 11821 WARCS (00000-12000, inclusive) resulting in 32K unique paths.

We parsed the raw HTML, treating it as a Latin-encoded string. In each HTML, regular expression functions for identifying Windows and Linux file paths are the following:

```
Windows_file path_with_ext = r"([a-z]:\\([a-z0-9() ]*\\)*[a-z0-9()]*\.
(jpg|jpeg|png|gif|mp4|mov|m4a|m4v|mpg|mpeg|wmv|avi|flv|3gp|3gpp|3g2|
3gp2|doc|docx|xls|xlsx|ppt|pptx|pdf))"

Linux_file path_with_ext = r"(/([a-zA-Z0-9()]*/)*[a-zA-Z0-9()]*\.
```

[6] https://commoncrawl.org

```
(jpg|jpeg|png|gif|mp4|mov|m4a|m4v|mpg|mpeg|wmv|avi|flv|3gp|3gpp|3g2|
3gp2|doc|docx|xls|xlsx|ppt|pptx|pdf))"
```

After collecting the data set using the functions above, we filtered Windows filenames to exclude ":\u002F". In Linux filenames, we only keep the file paths that begin with: /usr/, /home, /etc, /tmp, and /var.

The evaluation of model performance in independent data sets is essential to understand model generalization. We test our best-performing model against a data set containing only benign file paths. We measured the false positive rate for the best-performing model, CNN, at different confidence thresholds.

As we can observe from Figure 3.5, there is a very small number of files that the model attributes a confidence score above 0.8, with a false positive rate is less than 0.01. For example, for Linux file paths, a decision threshold of 0.8 results in an FPR of ≈ 0.03, whereas a decision threshold of 0.95 results in an FPR of ≈ 0.001. For Windows file paths, a threshold of 0.8 prompts an FPR less than 0.01, while a decision threshold of 0.95 leads to an FPR less than 0.001. High decision thresholds are common design choices in detection systems, where only the high confidence samples are flagged and sent for human review.

Based on this evaluation, we recommend a careful choice of model threshold when using the model in general scenarios, i.e., scenarios where file paths do not come exclusively from suspect hard drives. The volume of data to be analyzed should also be considered when defining the model threshold.

3.5.2 Differentially Private CSAM Classification

To fine-tune a BERT model using DP-SDG in PyTorch, we loaded a pre-trained BERT model and added an output layer specific to the CSAM file path identification task. The main advantage of using a pre-trained BERT model is that the model is mostly trained on public data, resulting in having the privacy budget used only in the fine-tuning task.

The model can then be fine-tuned on a labeled data set using DP-SDG. The DP-SDG algorithm modifies the gradients computed during backpropagation to satisfy DP constraints.

Table 3.5: Evaluation of differentially private model performance. We fine tuned a BERT model using DP-SGD optimization algorithm and the CSAM data set.

epsilon	accuracy	precision	recall
0.1	0.88	0.8	0.72
0.5	0.9	0.82	0.77
1.0	0.89	0.82	0.76
5.0	0.9	0.83	0.8

In our model, the DP guarantees provide file path privacy. This means that for any given file path, an adversary cannot infer whether the file path was part of the training data or not.

The DP-SGD algorithm ensures that the model's parameters are updated in a way that preserves the privacy of each sample training data.

From Table 3.5 we can observe that fine-tuning using a differentially private will result in a decrease in model utility. Note that the overall utility decrease is about 0.05 points for accuracy, 0.1 points for precision and recall, which is at the same order of magnitude of the standard deviation is the models presented in 3.2.

One interesting observation is that there is not a significant difference in model utility when training models with privacy parameters $\epsilon = 0.5, 1.0$ or 5.0.

Our conclusion is that a model trained with differential privacy algorithms, although it has a decrease in performance, it performs with comparable precision and recall to non-DP bag-of-words and bag-of-n-grams models.

3.6 DISCUSSION

We presented in this chapter a framework for robust and private training of CSAM detection model, based solely on metadata. The presented framework, along with the experiments, provide a solid understanding of which are a the necessary analysis for deployment of robust machine learning models for CSAM detection. The experiments with privacy-preserving CSAM detection models show that adding differential privacy to the training process incurs a very small decrease in utility, enforcing that deployment of DP CSAM detection models can be practical.

CHAPTER 4

FAIRNESS AND UTILITY IMPACTS IN MACHINE LEARNING PIPELINES CAUSED BY SYNTHETIC DATASETS

Differential privacy (DP) is the standard for privacy-preserving statistical summaries (DWORK *et al.*, 2006a). Companies such as Microsoft (PEREIRA *et al.*, 2021a), Google (AKTAY *et al.*, 2020), Apple (TANG *et al.*, 2017), and government organizations such as the US Census (ABOWD, 2018), have successfully applied DP in machine learning (HAMIDA *et al.*, 2022; HAMIDA *et al.*, 2023) and data sharing scenarios. The popularity of DP is due to its strong mathematical guarantees. Differential Privacy guarantees privacy by ensuring that the inclusion or exclusion of any particular individual does not significantly change the output distribution of an algorithm.

In areas ranging from health care, humanitarian action, education, and socioeconomic studies, the publication and sharing of data is crucial for informing society and fostering scientific collaboration. However, the disclosure of such datasets can often reveal private, sensitive information. Privacy-preserving data publishing aims at enabling such collaborations while preserving the privacy of individual entries in the dataset. Tabular/categorical data about individuals are relevant in many applications, from health care to humanitarian action. Privacy-preserving data publishing for such data can be done in the form of a synthetic data table that has the same schema and similar distributional properties as the real data. The aim here is to release a perturbed version of the original information, so that it can still be used for statistical analysis, but the privacy of individuals in the database is preserved.

The biggest advantage of synthetic datasets is that, once released, all data analysis and machine learning tasks are performed in the same way it is done with real data. As noted by (QIAN *et al.*, 2023), the switch between real and synthetic data in data analysis and machine learning pipelines is seamless - the same analysis tools, libraries and algorithms are applied in

the same manner in both datasets. Other privacy-preserving technologies, such as federated learning, requires expertise and appropriate tools to perform data analysis and model training.

Due to all the potential benefits of synthetic data, understanding the impacts of synthetic data in downstream classification tasks have become of extreme importance. A trend observed in recent studies is to evaluate performance of synthetic data generators of two types: marginal-based synthesizers (MOVAHEDI *et al.*, 2023) and generative adversarial networks (GAN) based synthesizers (CHENG *et al.*, 2021; GANEV, 2021; QIAN *et al.*, 2023). Marginal-based synthetic data generators are suitable for tabular data only, and have gained increased popularity after the algorithm MST won the NIST competition in 2018 (MCKENNA *et al.*, 2021). Marginal-based synthesizers are named as such due to the fact that they learn approximate data distributions by querying noisy marginals from the real data. Notable marginal-based algorithms are MST (MCKENNA *et al.*, 2021), MWEM PGM (MCKENNA *et al.*, 2019), AIM (MCKENNA *et al.*, 2022) and PrivBayes (ZHANG *et al.*, 2017). GAN-based synthesizers, on the other hand, are flexible algorithms, and are suitable for tabular, image and other data formats. GANs learn patterns and relationships from the input data based on a game, in the sense of game theory, between two machine learning models, a discriminator model and the generator model. Among popular differentially private GAN architectures we list DP-GAN (XIE *et al.*, 2018), DP-CTGAN (ROSENBLATT *et al.*, 2020a) , PATE-GAN (JORDON *et al.*, 2018) and PATE-CTGAN (ROSENBLATT *et al.*, 2020a).

One of the major applications of synthetic data is for training machine learning models. Therefore, it is paramount to understand how exchanging real data for synthetic data impacts the performance of the trained machine learning models. By performance, we mean not only the utility of the model (its accuracy, for example) but also how well the model performs for different subgroups of the dataset - the fairness of the model. The impact of machine learning models on minorities subgroups is an active area of research, and several works have investigated the trade-offs among model accuracy, bias, and privacy (WIENS *et al.*, 2019; BAGDASARYAN *et al.*, 2019; CALMON *et al.*, 2017; RAJOTTE *et al.*, 2021). However, only recently bias caused by the use of synthetic data in downstream classification received attention (GANEV *et al.*, 2022; MOVAHEDI *et al.*, 2023; GILES *et al.*, 2022). This problem becomes particularly relevant in the context of synthetic datasets generated with differential privacy guarantees. It is known

that differential privacy can affect fairness in machine learning models (BAGDASARYAN *et al.*, 2019). Despite recent work investigating the impact of synthetic data in downstream model fairness (GANEV *et al.*, 2022; CHENG *et al.*, 2021), there are important questions that remain unanswered:

- There is no published work that systematically studies the utility and fairness of machine learning models trained on several GAN-based and marginal-based synthetic tabular dataset generation algorithms;
- Previous studies have not evaluated machine learning models trained on synthetic dataset generation algorithms for multiple definitions of fairness;
- In previous studies, it was always assumed that real data was available for evaluating the fairness of models trained on synthetic data. Here, we propose and evaluate a pipeline where no such assumption is necessary.

Contributions In this chapter, we investigate the impacts of differentially private synthetic data on downstream classification, where we focus on understanding the impacts on model utility and fairness. Our investigation focus on two aspects of such impact:

- What is the impact in model utility when utilizing synthetic data for training machine learning models? Can synthetic data also be used to evaluate utility of machine learning models?
- What is the impact in model fairness when utilizing synthetic data for training machine learning models? Can synthetic data be used to evaluate fairness of machine learning models?

In our investigations we also evaluate if there are clear differences in performance between marginal-based and GAN-based synthetic data, and if there is a synthesizer algorithm type that produces data that clearly outperform others.

Our research work evaluates the impact of utilizing synthetic datasets for both training and testing in machine learning pipelines. We empirically compare the performance of marginal-based synthesizers and GAN-based synthesizers within the context of a machine learning pi-

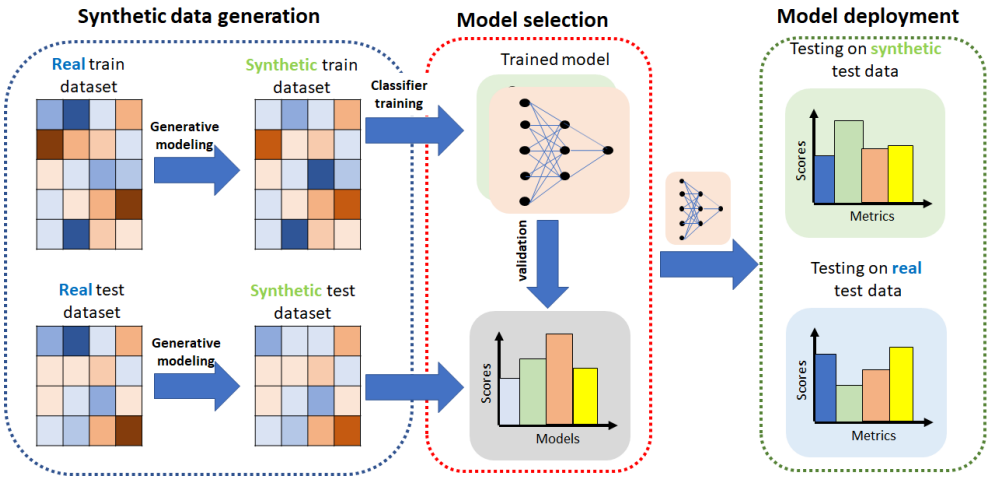

Figure 4.1: Pipeline for model training and evaluation using synthetic data (1) We generate Synthetic datasets for model training and model testing utilizing differentially private synthesizers. (2) We train models utilizing differentially private synthetic data and evaluate on a differentially private synthetic test data. Model selection is made during this phase. (3) Based on the previous phase results, model is trained using synthetic data and deployed. Model is applied to real (test) data in production phase.

peline for classification tasks. Our experiments yield a comprehensive analysis, encompassing utility and fairness metrics. Our main contributions are:

- We propose a training and evaluation framework that does not assume that real data is available for testing the utility and fairness of machine learning models trained on synthetic data.
- We present an extensive analysis of synthetic dataset generation algorithms in terms of privacy-loss, utility and fairness when used for training machine learning models. In particular, this is the first systematic comparison of several marginal-based and GAN-based algorithms for fairness and utility of the resulting machine learning models.
- This is the first of such studies that includes several different definitions of fairness.

Main Findings:

1 **Marginal-based synthetic data can accurately train machine learning models for tabular data.** Marginal-based synthetic data can train models with similar utility to models trained on real data. Our experiments show that for a privacy-loss parameter $\epsilon > 5.0$, models trained with AIM (AUC = 0.683), MWEM PGM (AUC = 0.684), MST

(AUC = 0.662) and Privbayes (AUC = 0.668) provides utility very similar to models trained on real data (AUC = 0.684). Additionally, we evaluated models using synthetic data, and found that marginal-based synthetic provides a good evaluation, with synthetic data providing an AUC = 0.666 versus AUC = 0.684 (measured using real data).

2 **Synthetic datasets generated with AIM and MWEM PGM have the potential be used for accurate model training and fairness evaluation in the case of tabular data.** Our experiments show that AIM and MWEM PGM synthetic data can train models that achieves very similar utility and fairness characteristics of models trained with real data. Additionally, the synthetic data generated by AIM algorithm, in our experiments, showed very similar behavior to real data when used to evaluate utility and fairness of machine learning models. This is the first study that presents evidence, from the perspective of utility and fairness, that synthetic data can be a substitute for real datasets in end-to-end machine learning pipelines for tabular data. It is interesting to investigate how these results generalize to larger data sets.

4.1 RELATED WORK

Synthetic data generation is a promising practice for privacy-preserving data sharing and publishing, understanding the impacts of utilizing synthetic data in machine learning pipelines is of significant importance. Although previous works have advised against using synthetic data to train and evaluate any final tools deployed in the real world (JORDON *et al.*, 2022), in very sensitive scenarios, such as human trafficking data (RESEARCH, 2022), and electronic health records (HERNADEZ *et al.*, 2023; YAN *et al.*, 2022), synthetic data is seen as a way to drastically increase the availability of research data. Particularly in health care, synthetic data can unlock research in areas like etiology of diseases, personalization of medicine, and healthcare administration assessment.

The promises synthetic data brings generated an interest in understanding impacts of utilizing synthetic in data analysis and machine learning. Some of these works include analyzing the utility of differentially private synthetic data in different tasks (TAO *et al.*, 2021), investigating if training models with differentially private synthetic images can increase subgroup disparities

Publication	Evaluation of synthetic data as training data	Evaluation of synthetic data as testing data	Evaluation of Algorithmic Fairness
(TAO *et al.*, 2021)	yes	no	no
(GANEV *et al.*, 2022)	yes	no	only subgroup accuracy
(MOVAHEDI *et al.*, 2023)	yes	yes	no
(GILES *et al.*, 2022)	yes	no	no
(ABAY *et al.*, 2019)	yes	no	no
(YOON *et al.*, 2020)	yes	no	no
This work	yes	yes	yes

Table 4.1: Previous works evaluating differentially private synthetic data generation in machine learning pipelines for tabular data. The works presented in this table all focus on understanding the impact of utilizing differentially private synthetic datasets in machine learning pipelines either from a perspective of utility or from a perspective of algorithmic fairness.

(CHENG *et al.*, 2021), the impacts different types of synthetic data can have in model fairness (GANEV *et al.*, 2022), utility of synthetic data in downstream health care classification systems (MOVAHEDI *et al.*, 2023), and whether feature importance can be accurately analyzed using differentially private synthetic data (GILES *et al.*, 2022). The evaluation of impacts of synthetic datasets in machine learning pipelines is made by comparing models trained with real data with models trained on synthetic data. The comparison is performed by testing both models on real data.The comparison can be performed using utility metrics (AUC-ROC, F1-score, accuracy) and also fairness metrics (subgroup accuracy, statistical parity, equality of odds). A complete survey of evaluation metrics for synthetic datasets can be found in (HERNADEZ *et al.*, 2023; YAN *et al.*, 2022).

Many of these works have made important findings in impacts of synthetic data in model utility and algorithmic fairness. In (TAO *et al.*, 2021) a comparison among different types of differentially private synthetic data generation algorithms found that marginal-based algorithms outperform all other types of DP synthetic data generators when training machine learning classifiers, with performance nearly matching the performance of a classifier trained on real data. The paper (GANEV *et al.*, 2022) finds that marginal-based synthetic data (PrivBayes) impacts machine learning pipelines by decreasing model bias, while GAN-based synthetic data increases model bias. All these works are ultimately trying to answer the same question: to which extent can we substitute real data with synthetic data, and which are the best synthetic data generation techniques for model training?

However these works still left questions unanswered. First of all, there hasn't been a systematic study of impacts of using synthetic datasets in end-to-end machine learning pipelines, which means evaluating the use of synthetic data for model training and model evaluation. Additionally, there has been a lot of focus on image classification tasks (CHENG *et al.*, 2021; GANEV *et al.*, 2022) where the disparity in accuracy are largely attributable to the class imbalance in these datasets: i.e disadvantaged classes are also rare classes in the dataset thereby leading to worse performance on these. In contrast, our work studies these issues in the context of tabular datasets and in settings where the data has an intrinsic bias against sub-populations that are not necessarily rare in the dataset. We summarize in Table 4.1 how previous works have evaluated the impacts of synthetic tabular data in machine learning pipelines, and how our work differentiates from previous analysis. Although several works have assessed the performance of machine learning models trained with synthetic datasets (TAO *et al.*, 2021; GANEV *et al.*, 2022), this is the first study to analyze if synthetic datasets can be used for model assessment, and how close to reality such assessment is from the point of view of utility and fairness. Moreover, our work focus on comparing two types of data synthesizing algorithm families: marginal-based and GAN-based data synthesizers. While, these two type of data synthesizing algorithms have been previously compared for utility (TAO *et al.*, 2021), no such extensive comparative analysis exists for fairness.

We are the first to extensively study the differences of applying data generated by these two families types of data synthesizing algorithms in end-to-end machine learning pipelines for utility and multiple fairness metrics.

4.2 DATASETS

We now describe the datasets used in our work. These datasets are commonly used in the literature for benchmarking algorithmic fairness in classification tasks (NGONG *et al.*, 2020; CALMON *et al.*, 2017; CELIS *et al.*, 2021).

4.2.1 Adult dataset

In the Adult dataset (32561 instances), the features were categorized as protected variable (C): gender (male, female); and response variable (Y): income (binary); decision variables (X): the remaining variables in the dataset. We map into categorical variables all continuous variables.

4.2.2 Prison Recidivism dataset

From the COMPAS dataset (7214 instances), we select severity of charge, number of prior crimes, and age category to be the decision variables (X). The outcome variable (Y) is a binary indicator of whether the individual recidivated (re-offended), and race is set to be the protected variable (C). We utilize a reduced set of features as proposed in (CALMON *et al.*, 2017).

4.2.3 Fair Prison Recidivism dataset

We construct a "fair"dataset based on the COMPAS recidivism dataset by employing a data preprocessing technique for learning non-discriminating classifiers from (KAMIRAN; CALDERS, 2012), which involves changing the class labels in order to remove discrimination from the dataset. This approach selects examples close to the decision boundary to be either 'promoted', i.e label flipped to the desirable class, or 'demoted', i.e label flipped to the undesirable class (ex: the 'recidivate' label in the COMPAS dataset is the undesirable class). By flipping an equal number of positive and negative class examples, the class skew in the dataset is maintained.

4.3 RESULTS

One potential outcome of synthetic data sharing is the utilization of synthetic data for training and evaluating an ML model. The trained model could be deployed without assessing its performance on real data, due to lack of data access. However, it is important to acknowledge that these trained models are ultimately applied to real data. This scenario is illustrated in

Figure 4.1. In our experiments, we address the concern that there may be substantial disparities in performance between the evaluation phase (employing synthetic data) and the deployment phase (utilizing real data). We refer to the experiments emulating the evaluation phase as *train on synthetic, test on synthetic (TSTS)*, and the experiments emulating the deployment phase as *train on synthetic, test on real (TSTR)*. We compare the performance of machine learning models trained with differentially private synthesizers, focusing on two performance dimensions: utility and fairness. We follow the approach of (GANEV *et al.*, 2022) and use logistic regression for downstream classification evaluation to avoid another layer of stochasticity. The utilization of a linear model allows us to better focus on the effects of different synthetic data generators in algorithmic fairness and model utility and reduce the effect of randomness in the training algorithms.

To assess the utility performance, we employ the AUC-ROC metric, which quantifies trade-off between the recall and false positive rate. We examine fairness performance through three different perspectives. Previous research (BAGDASARYAN *et al.*, 2019) has indicated that differentially private machine learning models tend to perform worse on minority groups. To this point we evaluate the decay in accuracy for the different subgroups in the protected attribute. We also measure the difference in equality of odds (DEO) and the difference in statistical parity (DSP). These metrics allow us to assess any disparities or bias in the model's predictions across different groups. Furthermore, we also investigate the extent to which one can accurately assess a model utilizing synthetic datasets. Again, we evaluate two performance dimensions: utility and fairness.

We utilized multiple differentially private marginal-based synthesizers (AIM, MST, MWEM-PGM, and PrivBayes) as well as GAN-based synthesizers (DP-GAN, DP-CTGAN, PATE-GAN, and PATE-CTGAN) to generate synthetic data. In our experiments, we generated datasets utilizing each synthetic data generation technique in combination with four different privacy-loss budgets $\epsilon = \{0.5, 1.0, 5.0, 10.0\}$. The privacy-loss budget quantifies the privacy risk associated with the publication of the synthetic data set, as defined in Chapter 2. The choice of these budgets is based on previous research in synthetic data analysis and published synthetic datasets (GANEV *et al.*, 2022; RESEARCH, 2022). Previous studies showed that budgets at and lower than $\epsilon = 0.1$ (GANEV *et al.*, 2022; ROSENBLATT *et al.*, 2020a) result in synthetic data with

very low utility, so our experiments focused on budgets greater than 0.5. The selection of $\epsilon = 10.0$ as the maximum budget aligns with other works in the literature on differentially private synthetic data generation (GANEV, 2021; GANEV *et al.*, 2022; MOVAHEDI *et al.*, 2023) . We also observed this magnitude of privacy-loss budget in published synthetic datasets, such as the Global victim-perpetrator synthetic data, which was generated with a privacy-loss budget of $\epsilon = 12$ (RESEARCH, 2022).

We divide the real dataset into 10 random 80/20 data splits, separating the data into generator and test datasets. For the TSTR experiments, we run 10 rounds of synthetic DP data generation on the 80% splits (generator data), used to generate the synthetic train datasets. We use the remainder 20% split as test data in the TSTR experiments. For the TSTS experiments, we run 10 rounds of synthetic DP data generation on the 80% splits (generator data), where we generate synthetic train datasets. We use the same generator data to generate the synthetic test data used in the TSTs experiments. We utilize the SmartNoise Library (VADHAN *et al.*, 2019) and DiffPrivLib (HOLOHAN *et al.*, 2019) implementations of the synthesizers, and approximate-DP approaches use the library's default value of δ.

We train Logistic Regression models using the generated DP synthetic train datasets. In experiments where we test the trained models on real data, model performance is evaluated on the real test data (the 20% test split from the real data). In experiments where we test the trained models on synthetic data, models are evaluated using the synthetic test datasets.

We report, for each technique and each value of privacy loss parameter, the mean across 10 rounds. The mean across multiple rounds serve to capture the behavior of each synthesizer and attenuate the effects of randomness. A similar approach was used in (GANEV *et al.*, 2022). Our experiments use three datasets: the UCI Adult dataset (DUA; GRAFF, 2017) and ProPublica's COMPAS recidivism data (BARENSTEIN, 2019), and a fair COMPAS dataset as defined in Section 4.2.3. The fair COMPAS dataset provides a way to evaluate synthetic data generation performance in fair and biased versions of the same dataset.

4.3.1 Utility analysis: impacts of synthetic data in machine learning pipelines

We evaluate the quality of models trained with synthetic datasets by measuring AUC and accuracy of the protected class. We consider privacy-loss budgets of $\epsilon = 0.5$, 1.0, 5.0 and 10.0. We compare the AUC obtained in our experiments with the AUC measured by training models with the real (non-synthetic) Adult, COMPAS, and fair COMPAS datasets.

Figure 4.2 shows AUC for different privacy losses and different synthesizers. The plots show the variation of AUC as a function of privacy-loss parameter ϵ for marginal-based and GAN-based synthesizers. The first row refers to marginal-based synthesizers in the TSTR mode. Experiments with COMPAS and fair COMPASS datasets showed that models trained on marginal-based synthetic data perform similarly to the baseline model (trained on real data). For all four synthesizers, we see an increase in AUC as we increase ϵ. Experiments with Adult dataset showed that AIM synthesizer outperformed all other synthesizer in both experimental settings: TSTR and TSTS. For COMPAS dataset (which has a small dimension) the performance of marginal-based synthetic datasets as training data is very close to the performance of the real data. The second row of figure 4.2 presents the performance of GAN-based synthetic data. Overall, the performance of GAN-based synthesizer is worse and the performance of the marginal-based synthesizer. The utility of data produced by GAN-based synthesizers fluctuated as we increased privacy-loss budget. This phenomenon had been previously observed in (ROSENBLATT *et al.*, 2020a). With AUC mostly fluctuating around ≈ 0.5, we can say that GAN-based synthetic data do not do much better than random guessing (for various values of ϵ). We attribute the fluctuations to the fact that GAN-based synthesizers are known to be data hungry and not capture well the intrinsic relationships between features when using small data sets for training data synthesizers (DHAMI *et al.*, 2021). The inferior performance of GAN-based synthesizers was also noted by (TAO *et al.*, 2021), which showed that models trained on GAN-based synthetic data perform worse than models trained on marginal-based synthetic data.

In third and fourth rows of Figure 4.2 we present the plots of variation of AUC for different values of epsilon for TSTS models. The plots in the third row refer to performance of models trained on marginal-based synthesizers, the the plots in the fourth row refer to GAN-based synthesizers. By comparing the models trained with marginal-based synthetic data when eva-

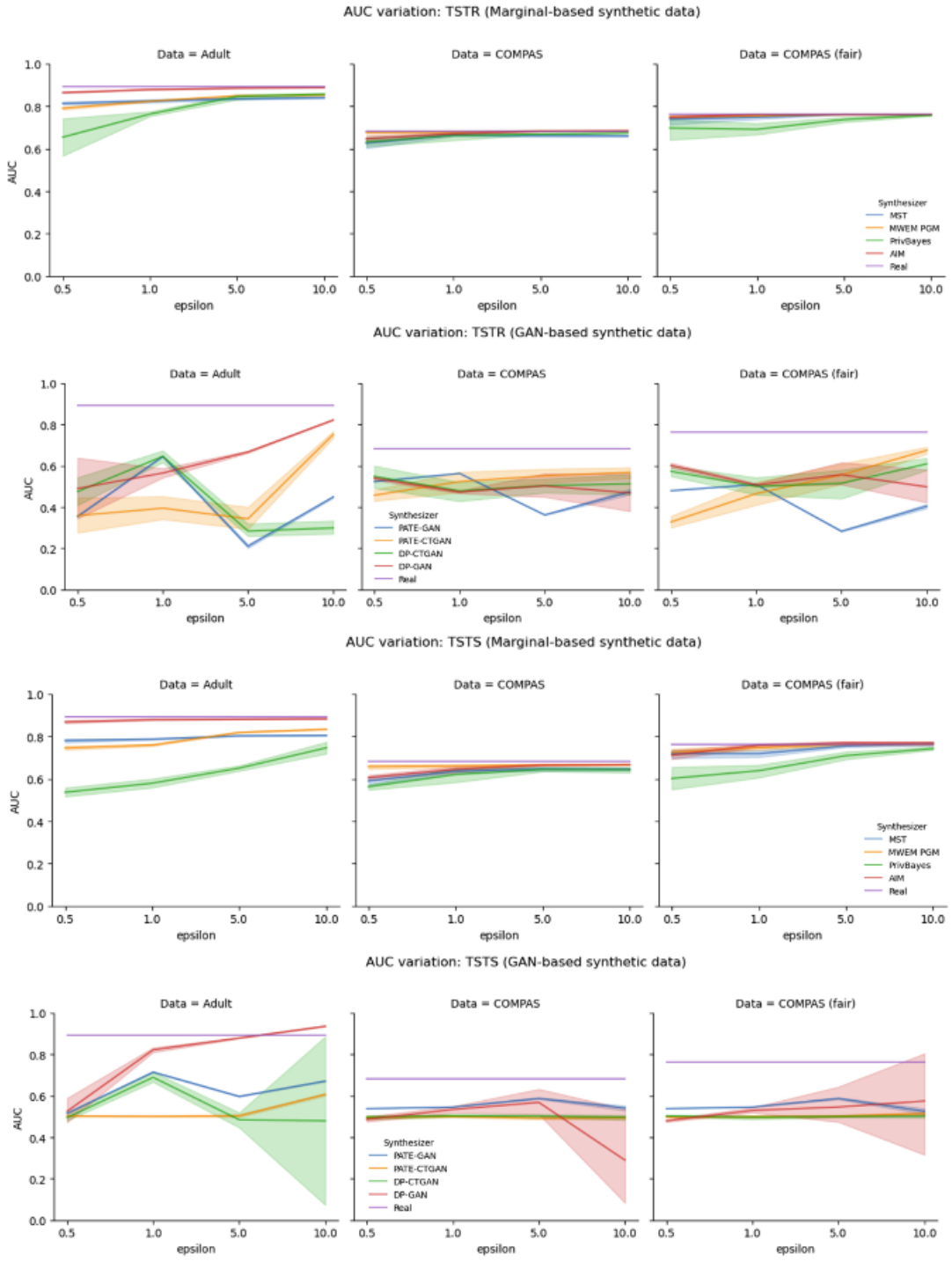

Figure 4.2: Impact in utility caused by the use of differentially private synthetic data in model training and testing. In the first two rows we show the decay in model utility when utilizing marginal-based and GAN-based synthetic datasets for model training. In the third and fourth rows we show what is the measured model utility when the instrument for measuring model performance is a synthetic dataset.

luated in different modes - TSTR and TSTS , we see that the assessment is very similar in both cases when the synthesizers are MST, AIM and MWEM PGM. When assessing with synthetic data, we notice that PrivBayes present a large difference in assessment results when assessing model trained on Adult and fair COMPAS synthetic data. GAN-based synthetic data, once again, present inconsistent behavior when used for model assessment. When comparing the assessments TSTR and TSTS, we notice that using DP-GAN sythetic data for model assessment can over estimate model AUC. Overall, GAN-based synthetic data made assessments that are as good as random guessing.

4.3.2 Fairness analysis: impacts of synthetic data in machine learning pipelines

4.3.2.1 Impacts on subgroup accuracy

In the previous section, we showed that adding privacy by utilizing synthetic datasets in machine learning pipelines results in a utility decrease in most cases. We now proceed to perform a fairness analysis. In the first experiment, presented in Table 4.2, we analyzed model accuracy for different groups in the protected class. The goal of the experiment is to understand whether the addition of privacy to the data pipeline harms model utility more for the minority class than it does for the privileged class. Results in Table 4.2 refer to the Adult, COMPAS and COMPAS (fair) datasets.

We first note that the model accuracy decay when training models with marginal-based (AIM) Adult synthetic data is smaller for the minority subgroup (Female), which presented an accuracy decay of 0.005, than it is for the privileged subgroup (Male), which presented an accuracy decay of 0.01. Models trained with marginal-based COMPAS synthetic data presented a slightly larger accuracy decay for the minority subgroup (Black) when compared to the accuracy decay for the privileged subgroup (Caucasian). Models trained on synthetic COMPAS fair dataset did not show accuracy decay in any of the subgroups. Overall, marginal-based synthesizers do not further accentuate subgroup accuracy disparities.

In the case of models trained with GAN-based synthetic datasets, no clear pattern of subgroup accuracy disparity was observed. For models trained with GAN-based Adult synthetic data, accuracy decay of the minority class (Female) was smaller than accuracy decay for the

Synthesizer	Accuracy of different subgroups			
	minority (R)	minority (S)	privileged (R)	privileged (S)
Adult data				
Real	0.924	–	0.804	–
AIM	0.919	0.916	0.794	0.807
MWEM PGM	0.909	0.898	0.779	0.770
MST	0.914	0.895	0.756	0.765
PrivBayes	0.892	0.713	0.709	0.648
DP-GAN	0.733	0.929	0.585	0.855
PATE-CTGAN	0.892	0.938	0.695	0.942
DP-CTGAN	0.889	0.999	0.693	0.999
PATE-GAN	0.892	0.874	0.695	0.854
COMPAS data				
Real	0.632	–	0.644	–
AIM	0.630	0.610	0.645	0.633
MWEM PGM	0.630	0.627	0.644	0.598
MST	0.616	0.614	0.631	0.616
PrivBayes	0.619	0.598	0.639	0.622
DP-GAN	0.497	0.514	0.451	0.452
PATE-CTGAN	0.536	0.497	0.377	0.499
DP-CTGAN	0.499	0.463	0.527	0.450
PATE-GAN	0.466	0.370	0.624	0.422
COMPAS (fair) data				
Real	0.690	–	0.679	–
AIM	0.690	0.693	0.679	0.701
MWEM PGM	0.690	0.678	0.679	0.707
MST	0.691	0.685	0.704	0.699
PrivBayes	0.674	0.632	0.672	0.656
DP-GAN	0.513	0.366	0.542	0.474
PATE-CTGAN	0.471	0.499	0.437	0.510
DP-CTGAN	0.491	0.524	0.489	0.528
PATE-GAN	0.528	0.389	0.562	0.442

Table 4.2: Accuracy comparison for different subgroups of the protected attribute. The comparison presented accounts for synthetic data generated with privacy-loss parameter $\epsilon = 5.0$. We show a comparison of model accuracy for the different groups measured with real data (R), and model accuracy measured with synthetic data (S).

privileged class (Male). In the case of models trained with GAN-based COMPAS and COMPAS (fair) synthetic data, accuracy of both subgroups were close to 0.5, confirming previous results, that showed model trained with GAN-based data acting like random classifiers. What we confirmed with this experiment is that this phenomenon happens for all subgroups.

4.3.2.2 Impacts on statistical parity

A model presents statistical parity if the percentage of positive predictions are the same for all subgroups. The goal of the experiments in this section is to measure whether models trained with synthetic data preserve the characteristics of models trained on real data.

Our experiments measure the difference in statistical parity (DSP) of models. We measure DSP of models using real data - DSP(R), and using synthetic data - DSP(S). We present a detailed comparison of DSP for all three datasets and all synthesizers on Table 4.3. We notice from our experiments that several models trained on synthetic data seem to be less biased than the model trained on real data. In terms of training models that performs similarly to models trained with real data, AIM synthesizer outperformed all other algorithms, followed by MWEM PGM synthesizer. AIM presented the best results in preserving statistical parity, based on experiments with all three datasets: Adult, COMPAS and COMPAS fair. GAN-based synthesizers, overall presented an intriguing performance: in some cases it seems like it has achieved perfect fairness.

To understand better what is behind this apparent fairness provided some GAN-based synthetic datasets, we investigate the percentage of positive labelled samples in the training data, evaluation data and predictions of models on TSTR and TSTS modes. We present percentages for minority and privileged classes in Table 4.4.

As we investigate GAN-based synthetic data, we observe in Table 4.4 that synthetic data generated with PATE-GAN and PATE-CTGAN presents very similar percentages of samples with positive labels for each subgroup that belongs to the protected attribute. At a first sight, this seems like a dataset with promising fairness capabilities. However, when training models with such data, in most cases there were no positive predictions resulting from the model scoring. The model trained with PATE-GAN and PATE-CTGAN data acts like a majority baseline

Data	Synthesizer	DSP(R)	DSP(S)	DSP delta
Adult	AIM	0.193	0.184	0.009
	MST	0.083	0.072	0.011
	MWEM PGM	0.168	0.159	0.009
	PrivBayes	0.051	0.043	0.008
	DP-CTGAN	-0.001	0.000	-0.001
	DP-GAN	0.346	0.253	-0.093
	PATE-CTGAN	0.000	0.000	0.000
	PATE-GAN	0.000	0.000	0.000
	Real	**0.189**		
COMPAS	AIM	-0.207	-0.204	-0.002
	MST	-0.182	-0.101	-0.082
	MWEM PGM	-0.218	-0.190	-0.028
	PrivBaeys	-0.211	-0.153	-0.058
	DP-CTGAN	-0.034	0.001	-0.034
	DP-GAN	0.072	-0.089	0.161
	PATE-CTGAN	-0.008	-0.009	0.001
	PATE-GAN	0.000	-0.001	0.001
	Real	**-0.205**		
COMPAS	AIM	0.009	0.020	-0.010
(fair)	MST	-0.185	-0.090	-0.095
	MWEM PGM	-0.018	0.015	-0.032
	PrivBayes	-0.065	0.005	-0.060
	DP-CTGAN	-0.034	-0.004	-0.030
	DP-GAN	0.066	0.096	-0.030
	PATE-CTGAN	0.000	0.000	0.000
	PATE-GAN	0.000	0.000	0.000
	Real	**-0.025**		

Table 4.3: Difference in statistical parity (DSP) of models trained with synthetic data. We measure the DSP of models using real test data - DSP(R) and synthetic test data DSP(S). DEO delta quantifies the difference between DSP(R) and DSP(S). All synthetic data where generated using privacy-loss parameter $\epsilon = 5.0$.

classifier for all groups. The datasets generated with DP-CTGAN presented an accentuated disparity in positive labels percentages between minority and privileged classes. In the real Adult data 30% of privileged class contains positive labels, while only 10% of minority class contains positive labels. Although DP-GAN synthesizer generates data where 31% of privileged class with positive labels (a value similar to the one presented in the real data - 30%), there is a significant decrease in the percentage of positive class in the minority class, which is $\approx 6\%$. This imbalance is even further accentuated by the models trained with DP-GAN synthetic data. Model predictions resulted in over half of samples from the privileged class being classified with positive labels (versus 20% of minority class). For models trained with COMPAS and COMPAS

	Ratio of positive labels					
Synthesizer	Generated data		Predictions(R)		Predictions(S)	
	Adult data					
	Female	Male	Female	Male	Female	Male
Real	0.109	0.303	0.055	0.244		
AIM	0.110	0.303	0.049	0.242	0.056	0.239
MWEM PGM	0.120	0.307	0.042	0.209	0.043	0.202
MST	0.123	0.297	0.032	0.115	0.031	0.102
PrivBayes	0.259	0.342	0.004	0.060	0.102	0.143
PATE-GAN	0.125	0.144	≈ 0	≈ 0	≈ 0	≈ 0
PATE-CTGAN	0.056	0.058	≈ 0	≈ 0	≈ 0	≈ 0
DP-GAN	0.061	0.307	0.199	0.545	0.016	0.269
DP-CTGAN	≈ 0	0.002	0.227	0.130	≈ 0	≈ 0
	COMPAS data					
	Black	Caucasian	Black	Caucasian	Black	Caucasian
Real	0.504	0.402	0.499	0.294		
AIM	0.503	0.405	0.504	0.297	0.500	0.297
MWEM PGM	0.504	0.403	0.514	0.294	0.498	0.302
MST	0.477	0.443	0.567	0.384	0.538	0.433
PrivBayes	0.489	0.436	0.566	0.352	0.550	0.387
PATE-GAN	0.231	0.196	0.397	≈ 0	≈ 0	≈ 0
PATE-CTGAN	0.548	0.541	0.715	0.975	0.981	0.949
DP-GAN	0.745	0.583	0.442	0.908	0.004	≈ 0
DP-CTGAN	0.471	0.455	0.302	0.218	0.217	0.179
	COMPAS (fair) data					
	Black	Caucasian	Black	Caucasian	Black	Caucasian
Real	0.454	0.493	0.488	0.463		
AIM	0.453	0.493	0.487	0.487	0.478	0.492
MWEM PGM	0.454	0.491	0.480	0.463	0.466	0.478
MST	0.485	0.446	0.495	0.310	0.478	0.393
PrivBayes	0.450	0.497	0.561	0.491	0.530	0.520
PATE-GAN	0.232	0.194	0.397	≈ 0	≈ 0	≈ 0
PATE-CTGAN	0.606	0.598	0.397	≈ 0	≈ 0	≈ 0
DP-GAN	0.593	0.664	0.560	0.836	0.865	0.744
DP-CTGAN	0.581	0.576	0.492	0.398	0.421	0.401

Table 4.4: Ratio of samples with positive labels for each subgroup in the protect class in the Adult , COMPAS and COMPAS (fair) datasets. We compare percentages present in the true labels of the real data and the predicted labels. Analogously, we measure the ratio of samples with positive label present in the synthetic generated data and predicted labels for datasets generated using distinct synthesizer techniques. Predictions(R) represents ratio of positive prediction labels of an experiment where model trained on synthetic data was evaluated on real data, and Predictions(S) ratio of positive prediction labels of an experiment where model trained on synthetic data was evaluated on synthetic data.

fair synthetic datasets, similar behavior was observed.

AIM once again was the best overall performing model, as it preserves similar percentages of positive labels for all groups, 11% and 30% (compared to 11% and 30% in real data). Models trained with AIM also presented similar metric to models trained with real data, and even presenting slightly improvement in fairness. The runner-up synthetic data generator in preserving the ratio of positive labels was the MWEM algorithm.

The DSP delta presented in Table 4.3 quantifies the difference in DSP observed during model evalution with real data and model evaluation with synthetic data. For Adult dataset, a positive DSP delta means that evaluation with synthetic data observed fairer results than evaluation with real data. For COMPAS and fair COMPAS data, a negative DSP delta means that evaluation with synthetic data observed fairer results than evaluation with real data.

Across all datasets, models trained with AIM and MWEM PGM presented DSP metrics very similar to models trained with real data, this is captured by the DSP(R) metric.

4.3.2.3 Impacts on equal opportunity

Equal Opportunity requires equal True Positive Rate (TPR) across subgroups. Difference in equal opportunity (DEO) measures the difference of privileged group TPR and minority group TPR.

We perform a thorough analysis to understand two points related to equal opportunity. First, what is the DEO of models trained with synthetic datasets, and how does it compare with models trained with real data? Second, given that true positive rate is the foundation for understanding equal opportunity, we investigate whether synthetic data preserves true positive rates across all subgroups.

We present in Table 4.5 experiment results comparing DEO of models trained with differentially private synthetic datasets ($\epsilon = 5.0$). These experiment are similar to the statistical parity experiments, we use real data - DEO(R) - to measure DEO of models trained on synthetic data, as well as synthetic data - DEO(S).

Model trained with AIM and MWEM PGM synthetic data were the only ones that presented a similar DEO to the baseline model, outperforming all other models trained with synthetic

Data	Synthesizer	DEO (R)	DEO (S)	DEO delta
Adult	AIM	0.209	0.200	0.009
	MST	0.038	0.076	-0.037
	MWEM PGM	0.206	0.200	0.006
	PrivBayes	0.094	0.026	0.067
	DP-CTGAN	-0.002	≈0.00	-0.002
	DP-GAN	0.527	0.641	-0.116
	PATE-CTGAN	0.000	0.000	0.000
	PATE-GAN	0.000	0.000	0.000
	Real	**0.173**		
COMPAS	AIM	-0.201	-0.195	-0.006
	MST	-0.150	-0.089	-0.061
	MWEM PGM	-0.215	-0.224	0.009
	PrivBayes	-0.177	-0.127	-0.051
	DP-CTGAN	-0.031	-0.000	-0.031
	DP-GAN	-0.075	0.020	0.055
	PATE-CTGAN	-0.011	-0.009	-0.002
	PATE-GAN	0.000	-0.001	0.001
	Real	**-0.204**		
COMPAS	AIM	0.007	0.013	-0.006
(fair)	MST	-0.181	-0.073	-0.107
	MWEM PGM	-0.019	0.037	-0.056
	PrivBayes	-0.057	0.005	-0.062
	DP-CTGAN	-0.030	-0.005	-0.026
	DP-GAN	0.097	0.087	0.010
	PATE-CTGAN	0.000	0.000	0.000
	PATE-GAN	0.000	-0.001	-0.000
	Real	**-0.027**		

Table 4.5: Difference in equal opportunity (DEO) of models trained with synthetic data. We measure the DEO of models using real test data - DEO(R) and synthetic test data DEO(S). DEO delta quantifies the difference between DEO(R) and DEO(S). All synthetic data where generated using privacy-loss parameter $\epsilon = 5.0$.

data. Note that our comparison, as in the DSP case, focus on understanding which synthetic datasets can train models that behave as close as possible to models trained with real data. Models trained with MST, which presented promising utility metrics and subgroup accuracy, did not capture as well the difference in equality on odds in experiments with the Adult data. For experiments with COMPAS and fair COMPAS data, MST performs better, but still worse than AIM and MWEM PGM, as we can see on Table 4.5.

As we investigate the details of variation in TPR it becomes clear AIM algorithm is the the best technique for training models that preserve fairness characteristics of models trained with real data, followed by MWEM PGM algorithm. Experiments with Adult data (Figure 4.3) show that the difference between the privileged group TPR and the minority group TPR of models trained with AIM data is very similar to the difference between subgroups TPR of models trained with real data, for all values of privacy-loss parameter ϵ. Similar conclusion is achieved by observing experiments with COMPAS and COMPAS fair data (Figures 4.4 and 4.5). Not only the difference between the subgroup TPR of the model trained with AIM and MWEM PGM synthetic data is close to that of the model trained with real data, but the true positive rates of the subgroups are also very similar to the TPR of the model trained with real data. Figures 4.3 , 4.4 and 4.5 show that models trained with marginal-based synthetic data outperforms models trained with GAN-based synthetic data for our tested datasets.

We make a similar analysis when evaluating how good synthetic datasets are for assessing TPRs. Figures 4.3, 4.4 and 4.5 also present plots of TPR when synthetic data is used during model assessment. Models trained with AIM and MWEM PGM data present very similar assessment when using both real and synthetic data as test data. Models trained on MST and PrivBayes present greater discrepancies. Models trained on GAN-based data present even greater discrepancies between assessments made with real and synthetic data as test data.

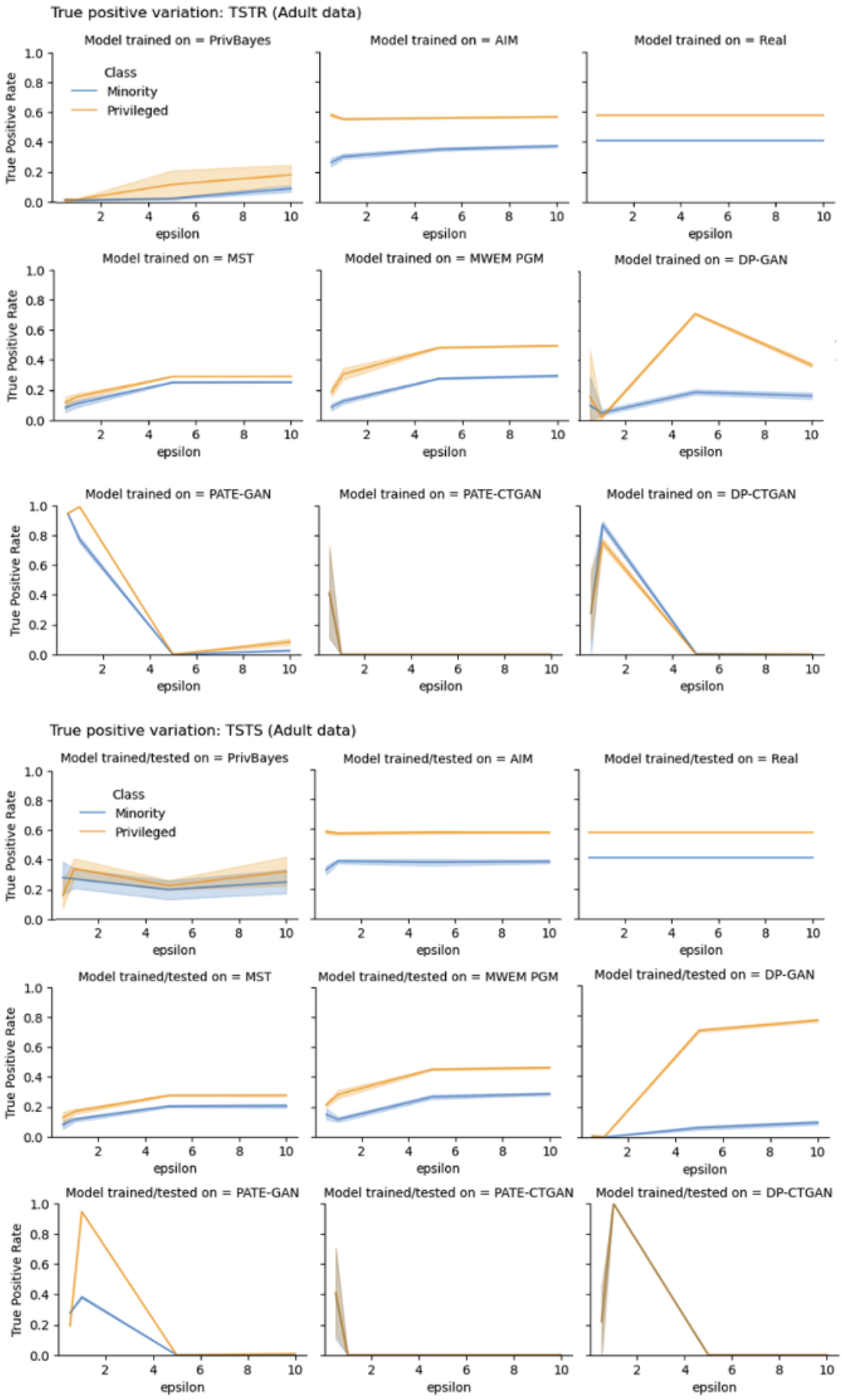

Figure 4.3: True positive rate (TPR) variation of different subgroups of the protected attribute of the Adult data. The top three rows shows TPR variation for different values of privacy-loss parameter ϵ, TSTR mode. The bottom three rows shows TPR variation for different values of privacy-loss parameter ϵ, TSTS mode.

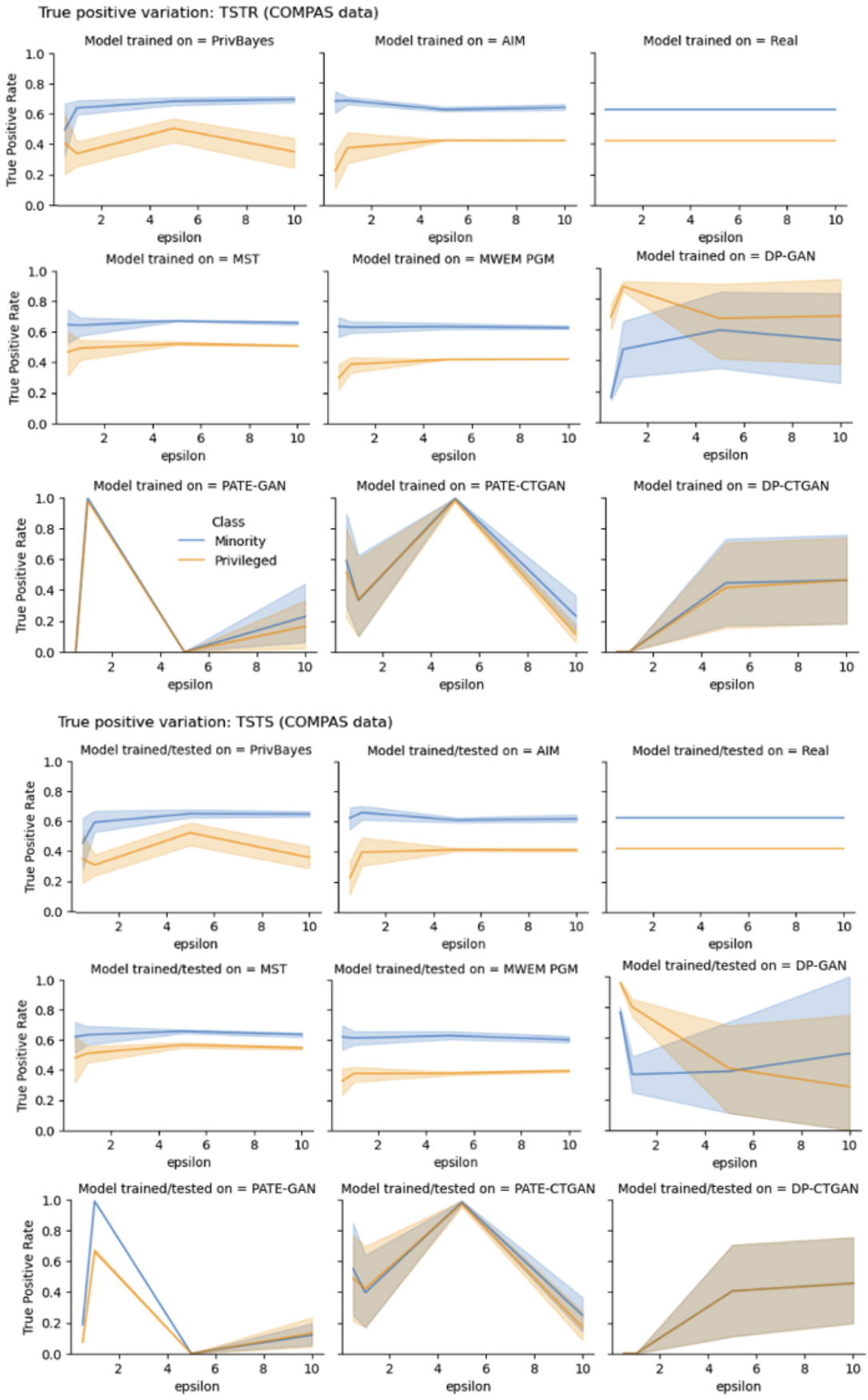

Figure 4.4: True positive rate (TPR) variation of different subgroups of the protected attribute of the COMPAS data. The top three rows shows TPR variation for different values of privacy-loss parameter ϵ, TSTR mode. The bottom three rows shows TPR variation for different values of privacy-loss parameter ϵ, TSTS mode.

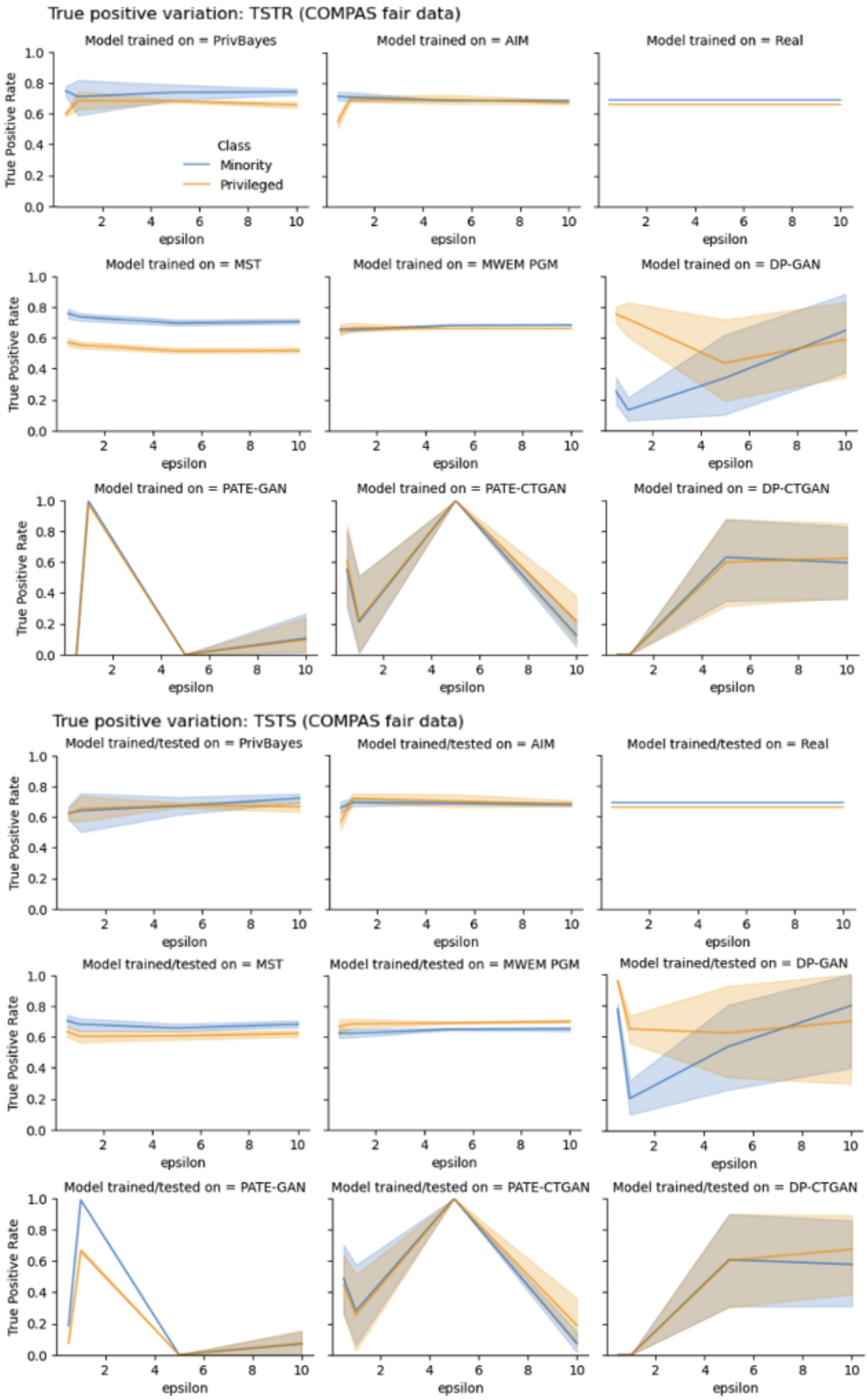

Figure 4.5: True positive rate (TPR) variation of different subgroups of the protected attribute of the COMPAS (fair) data. The top three rows shows TPR variation for different values of privacy-loss parameter ϵ, TSTR mode. The bottom three rows shows TPR variation for different values of privacy-loss parameter ϵ, TSTS mode.

	Adult		COMPAS		COMPAS Fair	
Synthesizer	Rank	AUC (R/S)	Rank	AUC (R/S)	Rank	AUC (R/S)
AIM	1st	0.886/0.882	2nd	0.683/ 0.666	2nd	0.761/0.771
MWEM PGM	2st	0.850/0.820	1st	0.684/ 0.666	1st	0.762/0.762
MST	3rd	0.836/0.804	4th	0.662/0.647	3rd	0.763/0.756
PrivBayes	4th	0.846/0.650	3rd	0.668/0.645	4th	0.738/0.710
DP-GAN	5th	0.667/0.880	7th	0.503/0.568	5th	0.557/0.546
PATE-CTGAN	6th	0.343/0.504	5th	0.552/0.492	6th	0.556/0.502
DP-CTGAN	7th	0.284/0.485	6th	0.504/0.502	7th	0.515/0.501
PATE-GAN	8th	0.210/0.597	8th	0.362/0.587	8th	0.283/0.588

Table 4.6: Synthesizer utility comparison. We compare and rank all synthesizers by their ability to generate quality training data and evaluation data for machine learning pipelines. The comparison presented accounts for synthetic data generated with privacy-loss parameter $\epsilon = 5.0$. In addition to present a performance ranking for Adult, COMPAS data and COMPAS (fair) data, we show a comparison of model AUC measured in TSTR mode - AUC(R), and model AUC measured in TSTS mode - AUC(S).

4.4 DISCUSSION

4.4.1 Marginal-based synthetic data does better at training and assessing utility of models.

The results in section 4.3.1, showed that models trained marginal-based synthetic data can have similar performance to models trained on real data. We observed the AIM synthetic data generation algorithm generated data that performed very closely to real data when training and evaluating machine learning models. The AIM data synthesizer presented a consistent performance across all datasets and for all values of privacy-loss parameter ϵ. To showcase a clear comparison between marginal-based and GAN-based synthesizers, we ranked the utility performance of all synthesizers taking based on two criteria: ability to generate synthetic data for model training and ability to generate synthetic data for model assessment. We ranked the synthesizers for each dataset used in our experiments. Table 4.6 shows the ranking of synthesizers when generating training and assessment data for the Adult data, COMPAS data and COMPAS (fair) data. The table also shows a comparison of model AUC measured in TSTR mode - AUC(R), and model AUC measured in TSTS mode - AUC(S). All table results accounts for synthetic data generated with privacy-loss parameter $\epsilon = 5.0$.

Synthetic data generated with the AIM algorithm outperforms (or tie with) all other synthe-

tic data for both tasks: utility as training data for machine learning models and utility as evaluation data for machine learning models. The performance of synthetic datasets generated with AIM was very similar to real data, both when using the synthetic data for model training and model assessment. For model training, when comparing the AUC achieved by model trained with the real Adult dataset (AUC = 0.892) to the metrics achieved by models trained with AIM Adult synthetic data (AUC= 0.886) and MWEM PGM Adult synthetic data (AUC = 0.850), the decrease in performance is small. The synthetic datasets also present a good performance as assessment data. The model assessment with AIM generated data showed good results, with an assessment of AUC = 0.882. Assessment with other marginal-based synthesizers, MST data (AUC = 0.804) and MWEM PGM data (AUC = 0.820), also presented consistent results, with a small decay. Although PrivBayes data presents good performance in model training (AUC = 0.846), there is a significant discrepancy between assessment utilizing real data and assessment utilizing synthetic data. We reached similar conclusions when analysing results for COMPAS and COMPAS (fair) data. Overall, our experiments using GAN-based data as training data resulted in models with utility very close to random guess. DP-GAN synthetic data performed slightly better than the rest of GAN-based datasets. We believe that the fact that the datasets used in our experiments are relatively small (less than 50k rows), GAN-based synthesizers do not have enough data samples to capture correctly the relationships between features. Although experiments with larger datasets can be useful to uderstand whether GAN-based synthesizers could do better with more data, the datasets used in our experiments are great representations of datasets found in the real world. Such datasets are rarely larger than a couple of thousand rows.

4.4.2 Marginal-based synthetic data preserves and better assess model fairness

We evaluated the performance of the synthetic datasets based on two key model fairness tasks: the ability to mirror the behavior of actual data in downstream model fairness, and the ability to produce synthetic data for assessing model fairness. Our analysis includes a rigorous assessment of model fairness, which includes measuring subgroup accuracy, the difference in statistical parity(DSP) and the difference in equal opportunity (DEO). Beyond measuring the classical fairness metrics, we also assess the Positive Predictive Value (PPV) and True Positive

Metric	Best Synthesizer	Runner up
Subgroup accuracy	AIM	MWEM PGM
Difference in statistical parity	AIM	MWEM PGM
Difference in equality of odds	AIM	MWEM PGM
PPV accross subgroups	AIM	MWEM PGM
TPR accross subgroups	AIM	MWEM PGM

Table 4.7: Best synthesizers for each fairness metric evaluated in the experiments: subgroup accuracy, difference in statistical parity and difference in equality of odds. We also present the synthesizers that best preserve PPV and TPR accross subgroups. We present the two best synthetic data generator for each task. We selected best synthesizer and runner up based on experiments with privacy-loss budget $\epsilon = 5.0$.

Rate (TPR) for each subgroup within the protected class. The significance of evaluating PPV and TPR lies in understanding if the model upholds fairness because it accurately represents PPV and TPR for all subgroups, or if it does so merely by acting as a random classifier.

Table 4.7 shows the best synthesizers in end-to-end machine learning pipelines when evaluating for fairness metrics. All table results accounts for synthetic data generated with privacy-loss parameter $\epsilon = 5.0$.

Throughout fairness experiments we observed that marginal-based synthetic datasets performed better than GAN-based synthetic dataset across all algorithmic fairness metrics. AIM and MWEM PGM synthetic data generation algorithms not only outperformed all other synthetic data generation algorithms, but these synthesizers generated data that performed similarly to real data in the three fairness metrics, and in our deeper investigations on PPV an TPR. This advantage was observed for multiple values of privacy-loss parameter ϵ, when synthetic data was used as a training dataset as well as when used as a testing dataset.

The investigation of subgroup PPV and TPR metrics clarified our observations regarding model fairness performances. We note that AIM and MWEM PGM synthetic data presents a ratio of positive labels comparable to that obtained with real data (Table 4.4), for all subgroups. When evaluating the ratio of positive labels in prediction for all subgroups in the Adult data (female and male) and in the COMPAS and COMPAS (fair) data (black and caucasian) in Table 4.4, we see that AIM and MWEM PGM also results is metrics that are the closest to real data.

The evaluation of true positive rate provides more insights into the bias introduced by synthetic dataset in end-to-end machine learning pipelines. Figures 4.3, 4.4 and 4.5 shows

the variation of TPR for different values of ϵ, in experiments with Adult, COMPAS and fair COMPAS, respectively. For COMPAS dataset, AIM provides the best performance, comparable the real dataset in an end-to-end analysis. For Adult data, $\epsilon > 1$ provides comparable metrics. Other algorithms, such as PrivBayes, that presented utility results (AUC metric) comparable to real data, showed low performance in terms of TPR. Finally, marginal-based synthesizers presented similar performance from the point of view of utility and fairness for both biased and fair versions of the COMPAS dataset.

CHAPTER 5

SECURE MULTIPARTY COMPUTATION FOR SYNTHETIC DATA GENERATION FROM DISTRIBUTED DATA

We live in an era of abundant data, where enormous amounts of personal data are collected daily via smartphones, social media, smartwatches, medical devices, among many other services. These datasets have helped researchers and industry understand our behavior better on both individual and collective levels, and have also allowed important research studies in many disciplines, including health, education, and economy. At the same time, we see an increase in privacy regulations globally. Following the introduction of the GDPR,[1] more than 60 jurisdictions around the world have proposed postmodern data privacy protection laws. By 2024, 75% of the world's population will have its personal information covered under modern privacy regulations (RIMOL, 2022). While privacy regulations are of extreme importance from an ethics perspective, they can potentially result in data stored in silos, compromising data usage and data sharing, and stalling research.

Synthetic data generation is emerging as a paradigm to break this data logjam. While data synthesis is arguably best known as a means to create training examples for data hungry deep learning models (NIKOLENKO, 2021), it is increasingly acknowledged and proposed as a privacy-enhancing technology (PET) (JORDON *et al.*, 2018; MCKENNA *et al.*, 2021; Science and Technology Policy Office, 2022; TORKZADEHMAHANI *et al.*, 2019; WALONOSKI *et al.*, 2018; XIE *et al.*, 2018). When done well, synthetic data has the same distribution or characteristics as the underlying, real data, but, crucially, without replicating personal information. The latter is often formalized through the notion of Differential Privacy (DP) (DWORK *et al.*, 2006a), which intuitively means that the synthetic data should not reveal specifics about *individual* records in the underlying, real data.

[1] European General Data Protection Regulation <https://gdpr-info.eu/>

The contributions of this paper are the following: (1) We introduce a framework for synthetic data generation from distributed databases that utilizes Secure Multiparty Computation (MPC) (CRAMER *et al.*, 2015) protocols that are run by two or more computing parties to emulate a trusted curator. This simulation enables the generation of synthetic data from training data held by multiple data holders, without requiring these data holders to disclose their data to anyone in an unencrypted manner. (2) We modify the Multiplicative Weights with Exponential Mechanism (MWEM), to generate synthetic data with DP guarantees, based on real data originating from many data holders, and without reliance on a single point of failure. (3) We propose an MPC protocol for secure sampling from distributed data using the exponential mechanism.

5.1 CONTRIBUTIONS.

Previous proposals for differentially private synthetic data generation from distributed databases use federated learning (FL) for training the data synthesizer (BEHERA *et al.*, 2022; XIN *et al.*, 2022; XIN *et al.*, 2020). In (BEHERA *et al.*, 2022; XIN *et al.*, 2022; XIN *et al.*, 2020), synthetic data is generated utilizing generative adversarial networks (GANs) in combination with FL. (XIN *et al.*, 2020) proposes a method based on serial training, and (XIN *et al.*, 2022) presents a variation of the framework to account for non-IID datasets. In (BEHERA *et al.*, 2022), the training of the data synthesizer occurs in parallel, resulting in more efficient results. In these methods, each data holder sends model weights (without privacy protection) to a trusted aggregator, who computes the average of model weights and adds Laplacian noise. Our proposal removes the need for data holders to disclose model parameters, and the need to rely on a single point of failure, by emulating the trusted aggregator with MPC. Additionally, previous works utilizing FL to train data synthesizers only account for horizontally partitioned data.

While MPC has emerged as a paradigm for privacy-preserving training of ML models over distributed data (e.g. (ADAMS *et al.*, 2022; AGARWAL *et al.*, 2019a; De Cock *et al.*, 2021; GUO *et al.*, 2022; MOHASSEL; ZHANG, 2017; WAGH *et al.*, 2019)) and privacy-preserving inference with trained ML models (e.g. (De Cock *et al.*, 2019; FRITCHMAN *et al.*, 2018; LIU *et al.*, 2017; MISHRA *et al.*, 2020; PENTYALA *et al.*, 2021)), and it has been proposed for

Algorithm 1: The MWEM algorithm (Hardt et al. [2012])

Input : Dataset D over a universe $\mathcal{D}$, set of linear queries Q, number of iterations T, and privacy parameter $\epsilon > 0$.
Let n denote $|D|$, the number of records in D.
Let A_0 denote n times the uniform distribution over $\mathcal{D}$.

1 **for** $i \in \{1, \ldots, T\}$ **do**

2 **Exponential Mechanism:** sample a query $q_i \in Q$ using the Exponential Mechanism parametrized with epsilon value $\epsilon/2T$ and the score function: $s_i(D,q) = |q(A_{i-1}) - q(D)|$

3 **Laplace Mechanism:** Let measurement $m_i = q_i(D) + \mathsf{Lap}(2T/\epsilon)$

4 **Multiplicative Weights:** Let A_i be n times the distribution whose entries satisfy

$$A_i(x) \propto A_{i-1}(x) \times \exp(q_i(x) \times (m_i - q_i(A_{i-1}))/2n)$$

5 **end**

Output: $A = \text{avg}_{i<T} A_i$

secure computation of histograms (e.g. (BELL *et al.*, 2022)), the idea of using MPC for privacy-preserving generation of synthetic data, as we propose here, is novel and a practical and secure technological solution.

5.2 METHODS

5.2.1 MWEM algorithm

As aeen in Chapter 2, the MWEM algorithm is a marginal-based synthetic data generation algorithm that takes as input a dataset $D \subseteq \mathcal{D}$ and a set of linear queries Q (e.g. counting queries).

The algorithm aims to produce a distribution A over $\mathcal{D}$ such that the answers to the queries q in Q when run over A are similar to when run over D, i.e. the difference between $q(A)$ and $q(D)$ should be small. This is achieved by repeatedly sampling a query for which the difference is still large (line 2 in Alg. 1), and updating the weight that A places on each record x with the Multiplicative Weights update rule to better approximate the distribution of D w.r.t. q (line 4). Furthermore, MWEM satisfies ϵ-DP by leveraging the exponential mechanism for query selection, and the Laplace mechanism to perturb the query results.

In the MWEM algorithm, the set of "results" to be selected from at each iteration is the set of queries $Q = \{q_1, q_2, \ldots, q_N\}$, and the value of the scoring function for query q_i is $s(D,q_i) = |q_(A) - q_i(D)|$, i.e. the difference in the answer for query q_i when run over the approximate

data A vs. when run over the real data D. Alg. 2 provides pseudocode for the exponential mechanism for query selection (HARDT *et al.*, 2012; ROSENBLATT *et al.*, 2020b). Lines 1–6 generate the probability distribution over the set of results (queries) as per Eq. (2.8), while lines 8–13 sample a result (query).

Algorithm 2: Algorithm for sampling a query using the Exponential Mechanism

Input : Answers to linear queries for synthetic data $q(A)$ and real data $q(D)$, number of linear queries N, and privacy parameter ϵ'.

1 // Compute the probability distribution over the set of queries
2 **for** $i \leftarrow 1$ **to** N **do**
3 $err[i] = 0.5 \cdot \epsilon' \cdot$ abs$(q_i(A) - q_i(D))$ //Note : $s(D,q_i) = |q_i(A) - q_i(D)|$
4 **end**
5 max_err = max(err)
6 **for** $i \leftarrow 1$ **to** N **do**
7 $err[i] = \exp(err[i] -$ max_err$)$
8 **end**
9 // Sample the query
10 e_s $= \sum_{i=1}^{N}(err[i])$
11 r = random value drawn from uniform distribution in [0,1]
12 c = 0
13 **for** $i \leftarrow 1$ **to** N **do**
14 c = c + $err[i]$
15 **if** c > r $\cdot$ e_s **then**
16 **return** i
17 **end**
18 **end**
19 **return** N

5.2.2 Distributed MWEM algorithm

We address the scenario where, instead of residing with one entity, the dataset D that we wish to give as input to the MWEM algorithm is distributed among multiple data holders who cannot disclose their data to anyone in an unencrypted manner. We distinguish between the *data holders* who hold the data sets, and the *computing parties* who run the MPC protocols for synthetic data generation and noise addition. Our solution works in scenarios in which each data holder (e.g. hospital or bank) is also a computing party, as well as in scenarios where the data holders outsource the computations to untrusted servers (computing parties) instead. The data holders send secret shares of their data to a set of computing parties.

Protocol 3: π_{QEM} - Protocol for secure sampling a query using the Exponential Mechanism

Input : The number N of queries in Q, secret-shared true query answer $[\![q_i(D)]\!]$ and approximate query answer $q_i(A)$ for each q_i in Q, privacy parameter $\epsilon' = \epsilon/(2T)$

1 Initialize a vector **err** of length N
2 **for** $i \leftarrow 1$ **to** N **do**
3 $\quad [\![\text{diff}]\!] \leftarrow [\![q_i(D)]\!] - q_i(A)$
4 $\quad [\![\text{sign}]\!] \leftarrow \pi_{\mathsf{LT}}([\![\text{diff}]\!], 0)$ // with secure comparison protocol π_{LT}
5 $\quad [\![\text{abs\_diff}]\!] \leftarrow \pi_{\mathsf{MUL}}(1 - 2 \cdot [\![\text{sign}]\!], [\![\text{diff}]\!])$ //with secure multiplication protocol π_{MUL}
6 $\quad [\![err[i]]\!] \leftarrow [\![\text{abs\_diff}]\!] \cdot 0.5 \cdot \epsilon'$
7 **end**
8 $[\![\text{max\_err}]\!] \leftarrow \pi_{\mathsf{MAX}}([\![\mathbf{err}]\!])$ // with secure maximum protocol π_{MAX}
9 **for** $i \leftarrow 1$ **to** N **do**
10 $\quad [\![err[i]]\!] \leftarrow \pi_{\mathsf{EXP}}([\![err[i]]\!] - [\![\text{max\_err}]\!])$ // with secure exponentiation protocol π_{EXP}
11 **end**
12 // Get random threshold to sample query
13 es $\leftarrow 0$
14 Initialize a vector **c** of length N
15 **for** $i \leftarrow 1$ **to** N **do**
16 $\quad [\![\text{es}]\!] \leftarrow [\![\text{es}]\!] + [\![err[i]]\!]$
17 $\quad [\![c[i]]\!] \leftarrow [\![\text{es}]\!]$
18 **end**
19 $[\![\text{r}]\!] \leftarrow \pi_{\mathsf{GR-RANDOM}}(0,1)$ // with protocol for random number generation $\pi_{\mathsf{GR-RANDOM}}$
20 $[\![\text{t}]\!] \leftarrow \pi_{\mathsf{MUL}}([\![\text{es}]\!],[\![\text{r}]\!])$
21 $s \leftarrow 0$
22 **for** $i \leftarrow 1$ **to** N **do**
23 $\quad [\![\text{cnd}]\!] \leftarrow \pi_{\mathsf{GT}}([\![c[i]]\!], [\![\text{t}]\!])$
24 $\quad [\![s]\!] \leftarrow [\![s]\!] + [\![\text{cnd}]\!]$
25 **end**
26 $[\![\text{cnd}]\!] \leftarrow \pi_{\mathsf{EQ}}([\![s]\!], 0)$
27 $[\![k]\!] \leftarrow N - \pi_{\mathsf{MUL}}([\![s]\!] - 1, 1 - [\![\text{cnd}]\!])$

Output: Secret-sharing $[\![k]\!]$ of the index of the selected query

Without loss of generality, we assume that the computing parties have secret shares of $[\![D]\!]$ of D, which they can use to compute a secret-sharing of the query result $[\![q(D)]\!]$ for each q in Q using primitive MPC protocols for addition and multiplication.

The execution of the overall MWEM algorithm can be coordinated by one of the data holders or any other entity interested in generating the synthetic data. Indeed, there are only two crucial steps in Alg. 1 that rely directly on the encrypted data $[\![D]\!]$, or rather $[\![q(D)]\!]$, hence requiring MPC computations involving all computing parties: (1) the query selection in line 2; and (2) the measurement in line 3. Note that the output of the computations in line 2 and line 3 is protected with DP guarantees. In other words, if we let the computing parties run MPC protocols for the computations and the DP mechanisms, then they can publicly reveal the selected query (line 2) and the perturbed query result (line 3), which can subsequently be used for further computations. Furthermore, there is no need to encrypt the synthetic data

distribution A, as it is not based on any information from D that is not already protected with DP. This is a welcome observation because it means that query evaluations need to be done only once over encrypted data, namely to compute $[\![q(D)]\!]$, and any further query evaluations on new versions of A can be done in-the-clear, i.e. without the need for encryption.

Description of π_{QEM}. For the secure query sampling on line 2 of Alg. 1, we propose MPC-protocol π_{QEM} (see Prot. 3) which is called with privacy budget $\epsilon' = \epsilon/(2T)$. π_{QEM} consists of two parts: on lines 1–9 the parties compute secret shares of the probability distribution over the queries, while on lines 10–23 the parties subsequently sample a query q_k from that distribution. Pseudocode for a corresponding algorithm in-the-clear, i.e. without regards for privacy, and the MPC primitives used in MWEM algorithm are described in Chapter 2.

The number and the kind of operations to construct the probability distribution and to compute the threshold (lines 1–9 in Alg. 2) are deterministic in the sense that they do not depend on the value of the data, hence their MPC counterpart in Prot. 3 is relatively straightforward. The implementation of the counterpart of the for-loop that starts on line 11 in Alg. 2 requires more care, as exiting the for-loop prematurely could allow an adversary to infer the value of the returned index from the runtime.

The code in line 18-23 in Prot. 3 is written to prevent such side-channel attacks. To understand this part of the code, note that we have a list $c[1..N]$ of non-decreasing values, i.e. the cumulative probability sums, and we – or rather the computing parties – have to find the first index i in $c[1..N]$ for which $c[i] > \mathrm{t}$. In a mock example with $N = 10$, and assuming that the first such $c[i]$ value is at position 7, the tests on line 20 will generate the results 0,0,0,0,0,0,1,1,1,1. On line 21, these results are accumulated in s, which eventually becomes 4, and the desired index is computed as $N - (s - 1) = 10 - 3 = 7$. Lines 22–23 take care of the edge case when $c[i] \leq \mathrm{t}$ for all i (i.e. s is 0). We protect the value of s by employing MPC primitives for multiplication to simulate a conditional statement.

Description of π_{LAP}. For the measurement computed in line 3 of Alg. 1, we design π_{LAP} (see Prot. 4) to securely sample noise from from the Laplacian distribution and add to the secret sharing of $q_i(D)$. The noise is sampled as $b \cdot \ln \mathrm{x} \cdot \mathrm{c}$ where $b = 2T/\epsilon$ is the privacy budget, x is a random value drawn from the uniform distribution in [0,1] and c is a random value selected from $\{-1,1\}$. On lines 1–2, the parties straightforwardly compute x and its natural log. To

Protocol 4: π_{LAP} - Protocol for Laplace mechanism

Input : Secret shared true query answer $[\![q_i(D)]\!]$ and $b = 2T/\epsilon$

1 $[\![\mathrm{x}]\!] \leftarrow \pi_{\mathsf{GR-RANDOM}}(0,1)$ // with protocol for random number generation $\pi_{\mathsf{GR-RANDOM}}$

2 $[\![\mathrm{ln\_x}]\!] \leftarrow \pi_{\mathsf{LN}}([\![\mathrm{x}]\!])$ // with secure logarithm protocol π_{LN}

3 $[\![\mathrm{r}]\!] \leftarrow \pi_{\mathsf{GR-RNDM-BIT}}()$ // with protocol for random bit generation $\pi_{\mathsf{GR-RANDOM}}$

4 $[\![\mathrm{c}]\!] \leftarrow 2 \cdot [\![\mathrm{r}]\!] - 1$

5 $[\![m_i]\!] \leftarrow [\![q_i(D)]\!] + b \cdot \pi_{\mathsf{MUL}}([\![\mathrm{ln\_x}]\!], [\![\mathrm{c}]\!])$ // with secure multiplication protocol π_{MUL}

Output: Secret sharing of measurement $[\![m_i]\!]$ for the query q_i, with $m_i = q_i(D) + \mathsf{Lap}(2T/\epsilon)$

compute c, the parties, on line 3, generate secret shares of a random bit $[\![\mathrm{r}]\!]$, i.e. a value $\in \{0,1\}$ is chosen, where each value has a chance of 50% to be chosen. On line 4, the parties transform r to a value $\in \{-1,1\}$ using the logic c = 2· r −1. Line 5 is straightforward where the parties compute secret shares of measurement $[\![m_i]\!]$ for the query q_i, which is then made public for further computations in Alg. 1.

5.3 EXPERIMENTS

We evaluate MPC-MWEM against a centralized version of MWEM (HARDT *et al.*, 2012) using two publicly available datasets, namely the Car and Adult datasets, which have been featured in previous DP synthetic dataset generation analyses (HARDT *et al.*, 2012; ROSENBLATT *et al.*, 2020b). In all the results below, **centralized** refers to the setting in which all data holders disclose their data to a central, trusted curator who runs the MWEM algorithm over all the data combined, while **distributed** refers to the setting in which the data holders secret share their data with computing parties who run MPC protocols. The distributed setting protects the privacy of the inputs, while the centralized setting does not. The results for the centralized setting are obtained with an implementation of MWEM in SmartNoise (ROSENBLATT *et al.*, 2020b). For the distributed setting, we implemented our MPC protocols π_{QEM} and π_{LAP} in the MPC framework MP-SPDZ (KELLER, 2020).

Experimental Settings. We empirically validate the utility of the produced synthetic data and measure performance by training logistic regression (LR) models using synthetic data and testing the models on real data, as in (ROSENBLATT *et al.*, 2020b). We evaluate model performance using AUC-ROC. We compare the performance of models trained on synthetic data generated in the centralized mode, and synthetic data generated in the distributed mode using MPC where the data is split horizontally across data holders. We repeat the comparison

process for different privacy parameter values. We measure runtimes of our method for different numbers of MWEM iterations T and compare with the centralized setting, while keeping other parameters constant.[2] We use a maximum number of iterations of 1000 and other default parameters of LR available in Scikit-learn (PEDREGOSA *et al.*, 2011) to train the models.

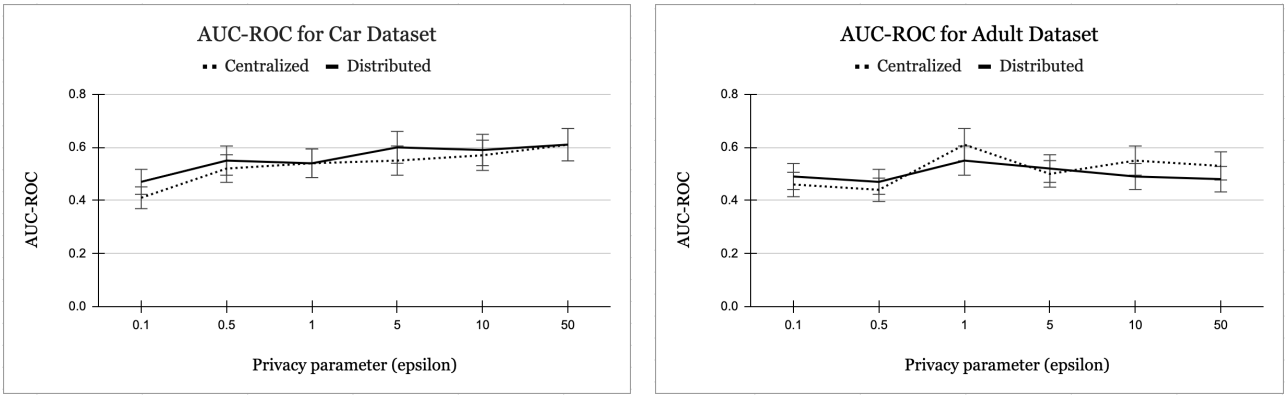

Figure 5.1: AUC-ROC of LR models trained on synthetic data generated by two different modes (centralized and distributed) with varying privacy budget. The results presented are averaged over 10 runs.

Quantitative Analysis of Utility. In Fig. 5.1, we investigate the trade-off between the privacy parameter ϵ and utility of models trained with synthetic data generated by the two different modes (centralized and distributed). The models perform similarly in terms of AUC-ROC, for the different values of ϵ. Additionally, the trends are also consistent for both datasets. In the experiments using the Car dataset we see a upward trend for both modes, whereas for the adult dataset we see a small spike for $\epsilon = 1$ for both modes. Based on similar trendlines for both settings, we conclude that MPC emulates the centralized mode of operation. The small differences observed in the plots are a result of the noise introduced by the DP mechanisms. The results in Fig. 5.1 are averaged over 10 runs.

Quantitative Analysis of Iterations and Runtime. We measure runtime for different values of the number of iterations T, which is a hyperparameter of MWEM. Previous works have demonstrated the trade-off between the number of iterations and quality of the synthetic data (HARDT *et al.*, 2012). Tab. 5.1 shows the runtime for different choices of T averaged over 3 runs for the centralized setting and for the distributed setting with 2, 3, and 4 computing parties. All MPC based computations were done in ring $\mathbb{Z}_q$ with $q = 2^{64}$. As observed, the

[2]We use the same parameters (such as number of queries, etc.) as in the SmartNoise tutorial notebooks. Similarly, for the Adult dataset, we use only the categorical columns as per the notebook (ROSENBLATT *et al.*, 2020b).

runtimes increase with T. We note that the runtimes further depend on the dimensions of the datasets and the number of queries, as shown in (HARDT *et al.*, 2012). The increased runtimes for the distributed setting when compared to their corresponding centralized setting are due to the runtimes of the MPC protocols. For example, in a 3PC passive security setting for $|T| = 1$, each call to π_{QEM} adds ∼0.74 secs for $|Q| = 400$ and π_{LAP} adds ∼0.006 secs to the synthetic generation process. The differences in runtime observed across different security settings are in line with existing literature (DALSKOV *et al.*, 2021). All the experiments were run on Azure D8ads_v5 8 vCPUs, 32Gib RAM.

Table 5.1: Runtime for different values of T (MWEM iterations). Central: Centralized setting runs the MWEM algorithm; Other columns: Distributed setting with 2 data holders and MPC protocols run on different number of computing servers with different security settings: 2PC, 3PC, 4PC. $|Q|$ is the number of queries, (a x b) denotes the dataset dimension.

Dataset	T	Central	2PC passive	3PC passive	3PC active	4PC active
Car (1,728 x 7) $\lvert Q = 400\rvert$	10	0.33 sec	12.14 sec	10.09 sec	20.52 sec	11.81 sec
	20	0.71 sec	23.50 sec	20.26 sec	43.01 sec	23.98 sec
	30	1.30 sec	37.86 sec	31.91 sec	66.4 sec	36.22 sec
	40	2.13 sec	51.20 sec	43.60 sec	87.53 sec	51.85 sec
Adult (12,499 x 12) $\lvert Q = 500\rvert$	10	3.96 sec	156.62 sec	39.98 sec	75.88 sec	111.75 sec
	20	4.95 sec	161.20 sec	41.70 sec	78.78 sec	115.72 sec
	30	6.45 sec	168.89 sec	44.80 sec	83.13 sec	121.59 sec
	40	8.53 sec	178.84 sec	48.80 sec	89.07 sec	129.77 sec

5.4 DISCUSSION

Marginal-based synthetic data generation algorithms, such as MWEM, are flexible synthetic data generation algorithms that can be adapted to generate synthetic data from distributed sources. The usage of MPC protocols for generating synthetic data without relying in a centralized authority does increase the running time of the data generation process, as shown in Table 5.1 , but also presents many advantages. The main advantage of using MPC, rather than other frameworks such as federated learning, is that MPC computations do not result in an utility reduction, as seen in Figure 5.1 (which is not the case of federated learning).

CHAPTER 6

CONCLUSION

While digital technology has transformed society in numerous ways, it has also created challenges for ensuring user safety and preserving privacy. Technology companies must address harmful social behaviors facilitated by their platforms, but they also have the potential to collect valuable data that can drive research and knowledge advancement in many fields. The combination of data sharing, machine learning, and artificial intelligence has already revolutionized research in several areas. However, privacy restrictions often limit researchers' access to the vast amounts of data that are locked in data silos. This work has focused on exploring and proposing machine learning and statistical models that can navigate these challenges while prioritizing ethical and legal considerations. By striking a balance between the potential of technology and the need for responsible use, we can advance research and make valuable contributions to numerous fields while also ensuring user safety and privacy.

6.1 DETECTING CHILD SEXUAL ABUSE MEDIA

The first part of this work proposes several machine learning models for CSAM file path detection and an evaluation framework for preparing machine learning models for CSAM file detection for deployment. Our evaluation framework covers real-world scenarios that surface when deploying a machine learning model for CSAM detection.

The proposed system for CSAM identification based solely on file paths has the advantage of not working directly with CSAM photos or videos. The classifier is a medium agnostic CSAM detector of easy maintenance and reduced legal restrictions for acquiring training data. Our classifier achieves precision and recall rates over 0.90 in out-of-sample hard drives. Our experiments also show that our models generalize well to identifying CSAM content in file storage systems and preserve low FPR in out-of-sample negative samples. Additionally, we

present a testing framework to evaluate model robustness to adversarial attacks introduced at test time.

The proposed framework is an essential addition to the available tools for CSAM detection. The community can leverage the proposed framework to train and evaluate models for CSAM metadata and short-text classification tasks, such as file path classification and CSAM search terms classification.

Operationally, using a CNN can dramatically reduce the burden on human evaluation. For example, using a threshold of 0.5, the CNN achieves a TPR of 0.94 and an FPR of 0.02. This suggests human review of results will discover 94% of actual CSAM examples, with an estimate of 24 false positives files for every 1000 non-CSAM files the model scans. While this does not remove the burden of human review, it significantly improves the status quo.

In combination with PhotoDNA hash, computer vision tools, and other forensics tools, our CSAM file path classifier integrates a global toolset that enables organizations to fight the distribution of CSAM.

Online child sexual abuse imagery falls into a category of content that should not be distributed or be present in file storage systems. The distributed nature of the internet makes CSAM detection a complex problem to solve. Automated tools and machine learning-based systems can help technology companies and investigation agencies rapidly identify such content and take the appropriate actions.

6.2 UNDERSTANDING IMPLICATIONS OF THE UTILIZATION OF SYNTHETIC DATA IN ML PIPELINES

As the privacy-preserving research community develops new and more sophisticated techniques for privacy-preserving data publishing, the natural question of fairness impacts arises. The second part of this work investigates the implications in model fairness when utilizing differentially private synthetic data for model training. We observe that model utility continuously decreases as we increase the privacy guarantees of synthetic data. However, fairness performance seems to be synthesizer dependent. Additionally, we observe that models trained with differently private synthetic data tend to perform more unfairly when tested on real data

versus when tested on synthetic data. This is an important observation as we see synthetic data techniques becoming more accepted as the standard data publishing approach in domains such as health care, education, and other population studies.

Our research comprehensively evaluates the impact of differentially synthetic datasets for training and testing machine learning pipelines in the case of tabular datasets. Specifically, we compare the performance of marginal-based and GAN-based synthesizers within a machine-learning pipeline and analyze various utility and fairness metrics for tabular datasets, across multiple values privacy-loss parameter ϵ.

Our main findings are as follows: Marginal-based synthetic data demonstrated comparable utility to real data in end-to-end machine-learning pipelines. AIM and MWEM PGM synthetic data generators provided the best utility across experiments, for various values of ϵ. AIM synthetic data, in particular, performed provided utility very close to models trained on real data, for multiple values of epsilon, for all datasets: Adult (AUC(R) = 0.892 vs AUC(S) = 0.886), COMPAS (AUC(R) = 0.684 vs AUC(S) = 0.683) and COMPAS fair (AUC(R) = 0.762 vs AUC(S) = 0.761). Furthermore, we show that model evaluation using synthetic data also provides similar results to evaluation using real data, for tabular data. The metrics obtained when utilizing AIM marginal-based synthetic data are comparable to real data, across all datasets and for multiple values of epsilon. Synthetic datasets trained with AIM and MWEM PGM synthetic data do not increase model bias and can provide a realistic fairness evaluation. Our study reveals that AIM and MWEM PGM synthetic data can train models that achieve similar utility and fairness characteristics as models trained with real data. Additionally, when used to evaluate the utility and fairness of machine learning models, our experiments showed that the synthetic datasets generated by the AIM algorithm exhibits behavior very similar to real data, for various values of ϵ .

One important point to raise is that, across all datasets used in our experiments (Adult, COMPAS and COMPAS fair) marginal-based algorithms (AIM and MWEM PGM specifically) were the best performing algorithms in terms of utility and fairness. From our experiments we gained evidence about an important fact: that synthesizer performance is independent from fairness characteristics of the original dataset.

These findings highlight synthetic data's potential reliability and viability as a substitute

for real datasets in end-to-end machine learning pipelines for tabular data. Furthermore, our research sheds light on the implications of model fairness when utilizing differentially private synthetic data for model training.

One crucial observation is that synthetic data that does well in model training might perform differently when used as evaluation data. This was the case with Privbayes and most of the GAN-based synthetic data generators. This observation is important as synthetic data techniques gain acceptance as a data publishing approach in domains such as healthcare, humanitarian action, education, and population studies.

6.3 SECURE MULTIPARTY COMPUTATION FOR SYNTHETIC DATA GENERATION FROM DISTRIBUTED DATA

In this thesis we introduced and started the study of a novel approach for generating differentially private synthetic data from distributed databases based on MPC. Our experiments show that utilizing MPC to emulate a central authority produces synthetic datasets with utility at par with data produced in a centralized fashion.

Many useful applications call for synthetic data generated based on original data that is held by multiple data owners. In this paper we proposed to replace the trusted curator that is used in current approaches by MPC protocols that generate synthetic data and privately perturb the data to satisfy DP requirements. We demonstrated this approach with MPC protocols for the MWEM algorithm, which is an MPC-friendly technique in the sense that the majority of the computations that need to be done over secret shares can be performed efficiently with state-of-the-art MPC schemes.

While the simplicity of MWEM makes it attractive to many applications and to adapting it to our framework, MWEM is a marginal-based synthetic data generator, and therefore, the proposed approach can be easily adapted to other marginal-based synthetic data generator such as MWEM-PGM, AIM and MST. From our assessment of different synthesizers in Chapter 4, we know that marginal-based synthesizers can present better performance than other kinds of synthesizers for some specific tasks.

6.4 FUTURE WORKS

Detection of medias with abusive content. This research presented a framework for robust deployment of CSAM detection models based on metadata. Future research should focus on expanding the framework to internet content such as URLs, web page metadata, social media users, social media pages and forums.

Assessment of synthetic data in machine learning pipelines. Although the datasets utilized in our analysis are commonly employed in fairness literature, extending the validity of our findings to larger-scale datasets would provide a more comprehensive understanding of the generalizability and robustness of marginal-based synthetic data approaches. Future research should focus on exploring the performance of these frameworks in real-world scenarios with diverse and extensive datasets, such experiments would clarify whether synthesizers behave differently in the presence of different types of dataset . This would contribute to the broader applicability and reliability of synthetic data methods in various domains and facilitate a more nuanced understanding of their limitations and capabilities. Finally, our work focuses solely on classification tasks. Extending our analysis to regression tasks, and evaluating fairness metrics (AGARWAL *et al.*, 2019b) in regression tasks when in presence of differentially private synthetic data hasn't been studied yet and would be an interesting sequel to this work.

Synthetic data generated from distributed sources. The work presented in this thesis proposes the first MPC-based differentially private synthetic data generation protocol. The proposed protocol focus on a marginal-based synthesizer, MWEM. Extending this protocol to other marginal-based synthesizers is a straightforward extension, however an interesting extension of this work would be modifying the algorithms of the marginal-based synthesizers to be more efficient when making the computations using MPC protocols.

REFERENCES

ABADI, M.; CHU, A.; GOODFELLOW, I.; MCMAHAN, H. B.; MIRONOV, I.; TALWAR, K.; ZHANG, L. Deep learning with differential privacy. In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. [S.l.: s.n.], 2016. p. 308–318.

ABAY, N. C.; ZHOU, Y.; KANTARCIOGLU, M.; THURAISINGHAM, B.; SWEENEY, L. Privacy preserving synthetic data release using deep learning. In: SPRINGER. *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018, Dublin, Ireland, September 10–14, 2018, Proceedings, Part I 18*. [S.l.], 2019. p. 510–526.

ABOWD, J. M. The us census bureau adopts differential privacy. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. [S.l.: s.n.], 2018. p. 2867–2867.

ADAMS, S.; CHOUDHARY, C.; COCK, M. D.; DOWSLEY, R.; MELANSON, D.; NASCIMENTO, A. C.; RAILSBACK, D.; SHEN, J. Privacy-preserving training of tree ensembles over continuous data. *Proceedings on Privacy Enhancing Technologies (PoPETS)*, p. 205–226, 2022.

AGARWAL, A.; DOWSLEY, R.; MCKINNEY, N. D.; WU, D.; LIN, C.-T.; De Cock, M.; NASCIMENTO, A. C. A. Protecting privacy of users in brain-computer interface applications. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, v. 27, n. 8, p. 1546–1555, 2019.

AGARWAL, A.; DUDÍK, M.; WU, Z. S. Fair regression: Quantitative definitions and reduction-based algorithms. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2019. p. 120–129.

AGARWAL, S.; GODBOLE, S.; PUNJANI, D.; ROY, S. How much noise is too much: A study in automatic text classification. In: IEEE. *Seventh IEEE International Conference on Data Mining (ICDM 2007)*. [S.l.], 2007. p. 3–12.

AKTAY, A.; BAVADEKAR, S.; COSSOUL, G.; DAVIS, J.; DESFONTAINES, D.; FABRIKANT, A.; GABRILOVICH, E.; GADEPALLI, K.; GIPSON, B.; GUEVARA, M. *et al.* Google covid-19 community mobility reports: Anonymization process description (version 1.0). *arXiv preprint arXiv:2004.04145*, 2020.

AL-NABKI, M. W.; FIDALGO, E.; ALEGRE, E.; ALAIZ-RODRÍGUEZ, R. Short text classification approach to identify child sexual exploitation material. *arXiv preprint arXiv:2011.01113*, 2020.

ARAKI, T.; FURUKAWA, J.; LINDELL, Y.; NOF, A.; OHARA, K. High-throughput semi-honest secure three-party computation with an honest majority. In: *ACM SIGSAC Conference on Computer and Communications Security*. [S.l.: s.n.], 2016. p. 805–817.

BAGDASARYAN, E.; POURSAEED, O.; SHMATIKOV, V. Differential privacy has disparate impact on model accuracy. *Advances in Neural Information Processing Systems*, v. 32, p. 15479–15488, 2019.

BARENSTEIN, M. Propublica's compas data revisited. *arXiv preprint arXiv:1906.04711*, 2019.

BAROCAS, S.; BRADLEY, E.; HONAVAR, V.; PROVOST, F. Big data, data science, and civil rights. *arXiv preprint arXiv:1706.03102*, 2017.

BAROCAS, S.; HARDT, M.; NARAYANAN, A. Fairness in machine learning. *Nips tutorial*, v. 1, p. 2, 2017.

BEHERA, M. R.; UPADHYAY, S.; SHETTY, S.; PRIYADARSHINI, S.; PATEL, P.; LEE, K. F. Fedsyn: Synthetic data generation using federated learning. *arXiv preprint arXiv:2203.05931*, 2022.

BELINKOV, Y.; BISK, Y. Synthetic and natural noise both break neural machine translation. *arXiv preprint arXiv:1711.02173*, 2017.

BELL, J.; GASCON, A.; GHAZI, B.; KUMAR, R.; MANURANGSI, P.; RAYKOVA, M.; SCHOPPMANN, P. *Distributed, private, sparse histograms in the two-server model*. 2022. Cryptology ePrint Archive, Paper 2022/920.

BOGDANOVA, D.; ROSSO, P.; SOLORIO, T. Exploring high-level features for detecting cyberpedophilia. *Computer speech & language*, Elsevier, v. 28, n. 1, p. 108–120, 2014.

BURSZTEIN, E.; CLARKE, E.; DELAUNE, M.; ELIFFF, D. M.; HSU, N.; OLSON, L.; SHEHAN, J.; THAKUR, M.; THOMAS, K.; BRIGHT, T. Rethinking the detection of child sexual abuse imagery on the internet. In: *The World Wide Web Conference*. New York, NY, USA: Association for Computing Machinery, 2019. (WWW '19), p. 2601–2607. ISBN 9781450366748. Disponível em: <https://doi.org/10.1145/3308558.3313482>.

CALMON, F. P.; WEI, D.; VINZAMURI, B.; RAMAMURTHY, K. N.; VARSHNEY, K. R. Optimized pre-processing for discrimination prevention. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. [S.l.: s.n.], 2017. p. 3995–4004.

CANETTI, R. Security and composition of multiparty cryptographic protocols. *Journal of CRYPTOLOGY*, Springer, v. 13, n. 1, p. 143–202, 2000.

CATRINA, O.; SAXENA, A. Secure computation with fixed-point numbers. In: *14th International Conference on Financial Cryptography and Data Security*. [S.l.]: Springer, 2010. (Lecture Notes in Computer Science, v. 6052), p. 35–50.

CELIS, L. E.; HUANG, L.; KESWANI, V.; VISHNOI, N. K. Fair classification with noisy protected attributes: A framework with provable guarantees. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2021. p. 1349–1361.

CHENG, V.; SURIYAKUMAR, V. M.; DULLERUD, N.; JOSHI, S.; GHASSEMI, M. Can you fake it until you make it? impacts of differentially private synthetic data on downstream classification fairness. In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. [S.l.: s.n.], 2021. p. 149–160.

COHEN, J. P.; DAO, L.; ROTH, K.; MORRISON, P.; BENGIO, Y.; ABBASI, A. F.; SHEN, B.; MAHSA, H. K.; GHASSEMI, M.; LI, H. *et al.* Predicting covid-19 pneumonia severity on chest x-ray with deep learning. *Cureus*, Cureus Inc., v. 12, n. 7, 2020.

CRAMER, R.; DAMGÅRD, I.; ESCUDERO, D.; SCHOLL, P.; XING, C. SPD$\mathbb{Z}_{2^k}$: Efficient MPC mod 2^k for dishonest majority. In: SPRINGER. *Annual International Cryptology Conference*. [S.l.], 2018. p. 769–798.

CRAMER, R.; DAMGÅRD, I.; MAURER, U. General secure multi-party computation from any linear secret-sharing scheme. In: SPRINGER. *International Conference on the Theory and Applications of Cryptographic Techniques*. [S.l.], 2000. p. 316–334.

CRAMER, R.; DAMGÅRD, I.; NIELSEN, J. B. *Secure Multiparty Computation and Secret Sharing*. [S.l.]: Cambridge University Press, 2015.

DALSKOV, A.; ESCUDERO, D.; KELLER, M. Fantastic four: Honest-majority four-party secure computation with malicious security. In: *USENIX 2021*. [S.l.: s.n.], 2021. p. 2183–2200.

DAVIDSON, J.; GOTTSCHALK, P. *Internet Child Abuse: Current Research and Policy*. [S.l.]: Routledge-Cavendish, 2010. ISBN 9780415559805.

DAWSON, R. Homoglyhps. *Codebox GitHub Repository*, 2022. https://github.com/codebox/homoglyph, Last accessed on 2023-01-30.

De Cock, M.; DOWSLEY, R.; HORST, C.; KATTI, R.; NASCIMENTO, A.; POON, W.-S.; TRUEX, S. Efficient and private scoring of decision trees, support vector machines and logistic regression models based on pre-computation. *IEEE Transactions on Dependable and Secure Computing*, v. 16, n. 2, p. 217–230, 2019.

De Cock, M.; DOWSLEY, R.; NASCIMENTO, A. C. A.; RAILSBACK, D.; SHEN, J.; TODOKI, A. High performance logistic regression for privacy-preserving genome analysis. *BMC Medical Genomics*, v. 14(23), 2021.

DENG, P.; LINSKY, C.; WRIGHT, M. Weaponizing unicodes with deep learning-identifying homoglyphs with weakly labeled data. In: IEEE. *2020 IEEE International Conference on Intelligence and Security Informatics (ISI)*. [S.l.], 2020. p. 1–6.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

DHAMI, D. S.; DAS, M.; NATARAJAN, S. Beyond simple images: human knowledge-guided gans for clinical data generation. In: *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*. [S.l.: s.n.], 2021. v. 18, n. 1, p. 247–257.

DU, X.; SCANLON, M. Methodology for the automated metadata-based classification of incriminating digital forensic artefacts. In: *Proceedings of the 14th International Conference on Availability, Reliability and Security*. [S.l.: s.n.], 2019. p. 1–8.

DUA, D.; GRAFF, C. *UCI Machine Learning Repository*. 2017. Disponível em: <http://archive.ics.uci.edu/ml>.

DWORK, C.; MCSHERRY, F.; NISSIM, K.; SMITH, A. Calibrating noise to sensitivity in private data analysis. In: SPRINGER. *Theory of cryptography conference*. [S.l.], 2006. p. 265–284.

DWORK, C.; MCSHERRY, F.; NISSIM, K.; SMITH, A. Calibrating noise to sensitivity in private data analysis. In: SPRINGER. *Theory of cryptography conference*. [S.l.], 2006. p. 265–284.

DWORK, C.; ROTH, A. *et al.* The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, v. 9, n. 3-4, p. 211–407, 2014.

FOURNIER, R.; CHOLEZ, T.; LATAPY, M.; CHRISMENT, I.; MAGNIEN, C.; FESTOR, O.; DANILOFF, I. Comparing pedophile activity in different p2p systems. *Social Sciences*, MDPI, v. 3, n. 3, p. 314–325, 2014.

FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, v. 55, n. 1, p. 119–139, 1997.

FRITCHMAN, K.; SAMINATHAN, K.; DOWSLEY, R.; HUGHES, T.; COCK, M. D.; NASCIMENTO, A.; TEREDESAI, A. Privacy-preserving scoring of tree ensembles: A novel framework for AI in healthcare. In: *Proc. of 2018 IEEE BigData*. [S.l.: s.n.], 2018. p. 2412–2421.

GANEV, G. Dp-sgd vs pate: Which has less disparate impact on gans? *arXiv preprint arXiv:2111.13617*, 2021.

GANEV, G.; OPRISANU, B.; CRISTOFARO, E. D. Robin hood and matthew effects: Differential privacy has disparate impact on synthetic data. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2022. p. 6944–6959.

GANGWAR, A.; FIDALGO, E.; ALEGRE, E.; GONZÁLEZ-CASTRO, V. Pornography and child sexual abuse detection in image and video: A comparative evaluation. In: IET. *8th International Conference on Imaging for Crime Detection and Prevention (ICDP 2017)*. [S.l.], 2017. p. 37–42.

GILES, O.; HOSSEINI, K.; MINGAS, G.; STRICKSON, O.; BOWLER, L.; SMITH, C. R.; WILDE, H.; LIM, J. N.; MATEEN, B.; AMARASINGHE, K. *et al.* Faking feature importance: A cautionary tale on the use of differentially-private synthetic data. *arXiv preprint arXiv:2203.01363*, 2022.

GOODFELLOW, I. J.; SHLENS, J.; SZEGEDY, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

GUO, C.; HANNUN, A.; KNOTT, B.; MAATEN, L. van der; TYGERT, M.; ZHU, R. Secure multiparty computations in floating-point arithmetic. *Information and Inference: A Journal of the IMA*, Oxford University Press, v. 11, n. 1, p. 103–135, 2022.

HAMIDA, S. B.; MRABET, H.; CHAIEB, F.; JEMAI, A. Assessment of data augmentation, dropout with l2 regularization and differential privacy against membership inference attacks. *Multimedia Tools and Applications*, Springer, p. 1–30, 2023.

HAMIDA, S. B.; MRABET, H.; JEMAI, A. How differential privacy reinforces privacy of machine learning models? In: SPRINGER. *International Conference on Computational Collective Intelligence*. [S.l.], 2022. p. 661–673.

HARDT, M.; LIGETT, K.; MCSHERRY, F. A simple and practical algorithm for differentially private data release. *Advances in Neural Information Processing Systems*, v. 25, 2012.

HARDT, M.; PRICE, E.; SREBRO, N. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, v. 29, 2016.

HEIDARI, H.; LOI, M.; GUMMADI, K. P.; KRAUSE, A. A moral framework for understanding fair ml through economic models of equality of opportunity. In: *Proceedings of the Conference on Fairness, Accountability, and Transparency*. [S.l.: s.n.], 2019. p. 181–190.

HEIGOLD, G.; NEUMANN, G.; GENABITH, J. van. How robust are character-based word embeddings in tagging and mt against wrod scramlbing or randdm nouse? *arXiv preprint arXiv:1704.04441*, 2017.

HERNADEZ, M.; EPELDE, G.; ALBERDI, A.; CILLA, R.; RANKIN, D. Synthetic tabular data evaluation in the health domain covering resemblance, utility, and privacy dimensions. *Methods of Information in Medicine*, 2023.

HOLOHAN, N.; BRAGHIN, S.; AONGHUSA, P. M.; LEVACHER, K. Diffprivlib: The IBM differential privacy library. *CoRR*, abs/1907.02444, 2019. Disponível em: <http://arxiv.org/abs/1907.02444>.

JÁÑEZ-MARTINO, F.; ALAIZ-RODRÍGUEZ, R.; GONZÁLEZ-CASTRO, V.; FIDALGO, E.; ALEGRE, E. A review of spam email detection: analysis of spammer strategies and the dataset shift problem. *Artificial Intelligence Review*, Springer, p. 1–29, 2022.

JIN, D.; JIN, Z.; ZHOU, J. T.; SZOLOVITS, P. *TextFool: Fool your Model with Natural Adversarial Text*. 2019. Http://groups.csail.mit.edu/medg/ftp/psz-papers/2019%20Di%20Jin.pdf.

JONES, K. S. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, MCB UP Ltd, 1972.

JORDON, J.; SZPRUCH, L.; HOUSSIAU, F.; BOTTARELLI, M.; CHERUBIN, G.; MAPLE, C.; COHEN, S. N.; WELLER, A. Synthetic data–what, why and how? *arXiv preprint arXiv:2205.03257*, 2022.

JORDON, J.; YOON, J.; SCHAAR, M. V. D. Pate-gan: Generating synthetic data with differential privacy guarantees. In: *International Conference on Learning Representations*. [S.l.: s.n.], 2018.

KAMIRAN, F.; CALDERS, T. Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, Springer, v. 33, n. 1, p. 1–33, 2012.

KAUFMAN, S.; ROSSET, S.; PERLICH, C.; STITELMAN, O. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, ACM New York, NY, USA, v. 6, n. 4, p. 1–21, 2012.

KELLER, M. MP-SPDZ: A versatile framework for multi-party computation. In: *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. [S.l.: s.n.], 2020. p. 1575–1590.

KUCHIPUDI, B.; NANNAPANENI, R. T.; LIAO, Q. Adversarial machine learning for spam filters. In: *Proceedings of the 15th International Conference on Availability, Reliability and Security*. [S.l.: s.n.], 2020. p. 1–6.

LATAPY, M.; MAGNIEN, C.; FOURNIER, R. Quantifying paedophile activity in a large p2p system. *Information Processing & Management*, Elsevier, v. 49, n. 1, p. 248–263, 2013.

LIU, J.; JUUTI, M.; LU, Y.; ASOKAN, N. Oblivious neural network predictions via miniONN transformations. In: *ACM SIGSAC Conference on Computer and Communications Security*. [S.l.: s.n.], 2017. p. 619–631.

MACEDO, J.; COSTA, F.; SANTOS, J. A. dos. A benchmark methodology for child pornography detection. In: IEEE. *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. [S.l.], 2018. p. 455–462.

MCKENNA, R.; MIKLAU, G.; SHELDON, D. Winning the NIST contest: A scalable and general approach to differentially private synthetic data. *arXiv preprint arXiv:2108.04978*, 2021.

MCKENNA, R.; MULLINS, B.; SHELDON, D.; MIKLAU, G. Aim: An adaptive and iterative mechanism for differentially private synthetic data. *arXiv preprint arXiv:2201.12677*, 2022.

MCKENNA, R.; SHELDON, D.; MIKLAU, G. Graphical-model based estimation and inference for differential privacy. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2019. p. 4435–4444.

MCSHERRY, F.; TALWAR, K. Mechanism design via differential privacy. In: *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*. [S.l.: s.n.], 2007. p. 94–103.

MCSHERRY, F. D. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In: *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. [S.l.: s.n.], 2009. p. 19–30.

MIRONOV, I. On significance of the least significant bits for differential privacy. In: *Proceedings of the 2012 ACM conference on Computer and communications security*. [S.l.: s.n.], 2012. p. 650–661.

MISHRA, P.; LEHMKUHL, R.; SRINIVASAN, A.; ZHENG, W.; POPA, R. A. Delphi: A cryptographic inference service for neural networks. In: *29th USENIX Security Symposium*. [S.l.: s.n.], 2020. p. 2505–2522.

MLADENIĆ, D. Feature subset selection in text-learning. In: SPRINGER. *European conference on machine learning*. [S.l.], 1998. p. 95–100.

MOHASSEL, P.; ZHANG, Y. SecureML: A system for scalable privacy-preserving machine learning. In: *2017 IEEE Symposium on Security and Privacy (SP)*. [S.l.: s.n.], 2017. p. 19–38.

MOVAHEDI, P.; NIEMINEN, V.; PEREZ, I. M.; PAHIKKALA, T.; AIROLA, A. Evaluating classifiers trained on differentially private synthetic health data. In: IEEE. *2023 IEEE 36th International Symposium on Computer-Based Medical Systems (CBMS)*. [S.l.], 2023. p. 748–753.

NABKI, M. W. A.; FIDALGO, E.; ALEGRE, E.; ALAÍZ-RODRÍGUEZ, R. File name classification approach to identify child sexual abuse. In: *ICPRAM*. [S.l.: s.n.], 2020. p. 228–234.

NG, A. Y.; JORDAN, M. I. On discriminative vs. denerative classifiers: A comparison of logistic regression and naive bayes. In: *Advances in neural information processing systems 14, Proceedings of the 2001 NIPS conference*. [S.l.]: MIT Press, 2001. p. 841–848.

NGO, V. M.; THORPE, C.; DANG, C. N.; MCKEEVER, S. Investigation, detection and prevention of online child sexual abuse materials: A comprehensive survey. In: IEEE. *2022 RIVF International Conference on Computing and Communication Technologies (RIVF)*. [S.l.], 2022. p. 707–713.

NGONG, I. C.; MAUGHAN, K.; NEAR, J. P. Towards auditability for fairness in deep learning. *arXiv preprint arXiv:2012.00106*, 2020.

NIAN, F.; LI, T.; WANG, Y.; XU, M.; WU, J. Pornographic image detection utilizing deep convolutional neural networks. *Neurocomputing*, Elsevier, v. 210, p. 283–293, 2016.

NIKOLENKO, S. I. *Synthetic data for deep learning*. [S.l.]: Springer, 2021. v. 174. (Springer Optimization and its Applications, v. 174).

PANCHENKO, A.; BEAUFORT, R.; NAETS, H.; FAIRON, C. Towards detection of child sexual abuse media: categorization of the associated filenames. In: SPRINGER. *Advances in Information Retrieval: 35th European Conference on IR Research, ECIR 2013, Moscow, Russia, March 24-27, 2013. Proceedings 35*. [S.l.], 2013. p. 776–779.

PAPERNOT, N.; ABADI, M.; ERLINGSSON, U.; GOODFELLOW, I.; TALWAR, K. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755*, 2016.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

PEERSMAN, C. *Detecting deceptive behaviour in the wild: text mining for online child protection in the presence of noisy and adversarial social media communications*. Tese (Doutorado) — Lancaster University, 2018.

PEERSMAN, C.; SCHULZE, C.; RASHID, A.; BRENNAN, M.; FISCHER, C. icop: Automatically identifying new child abuse media in p2p networks. In: IEEE. *2014 IEEE Security and Privacy Workshops*. [S.l.], 2014. p. 124–131.

PEERSMAN, C.; SCHULZE, C.; RASHID, A.; BRENNAN, M.; FISCHER, C. icop: Live forensics to reveal previously unknown criminal media on p2p networks. *Digital Investigation*, Elsevier, v. 18, p. 50–64, 2016.

PENTYALA, S.; DOWSLEY, R.; COCK, M. D. Privacy-preserving video classification with convolutional neural networks. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2021. p. 8487–8499.

PEREIRA, M.; DODHIA, R.; ANDERSON, H.; BROWN, R. Metadata-based detection of child sexual abuse material. *IEEE Transactions on Dependable and Secure Computing*, IEEE, 2023.

PEREIRA, M.; KIM, A.; ALLEN, J.; WHITE, K.; FERRES, J. L.; DODHIA, R. Us broadband coverage data set: A differentially private data release. *arXiv preprint arXiv:2103.14035*, 2021.

PEREIRA, M.; KSHIRSAGAR, M.; MUKHERJEE, S.; DODHIA, R.; FERRES, J. L. An analysis of the deployment of models trained on private tabular synthetic data: Unexpected surprises. *arXiv preprint arXiv:2106.10241*, 2021.

PEREIRA, M.; KSHIRSAGAR, M.; MUKHERJEE, S.; DODHIA, R.; FERRES, J. L.; SOUSA, R. de. Assessment of differentially private synthetic data for utility and fairness in end-to-end machine learning pipelines for tabular data. *Plos one*, Public Library of Science San Francisco, CA USA, v. 19, n. 2, p. e0297271, 2024.

PEREIRA, M.; PENTYALA, S.; COCK, M. D.; NASCIMENTO, A.; SOUSA, R. de. Secure multiparty computation for synthetic data generation from distributed data. In: *NeurIPS 2022 Workshop on Synthetic Data for Empowering ML Research*. [S.l.: s.n.], 2022.

PERRONE, V.; DONINI, M.; ZAFAR, M. B.; SCHMUCKER, R.; KENTHAPADI, K.; ARCHAMBEAU, C. Fair bayesian optimization. *arXiv preprint arXiv:2006.05109*, 2020.

QIAN, Z.; CALLENDER, T.; CEBERE, B.; JANES, S. M.; NAVANI, N.; SCHAAR, M. van der. Synthetic data for privacy-preserving clinical risk prediction. *medRxiv*, Cold Spring Harbor Laboratory Press, p. 2023–05, 2023.

RAJOTTE, J.-F.; MUKHERJEE, S.; ROBINSON, C.; ORTIZ, A.; WEST, C.; FERRES, J. L.; NG, R. T. Reducing bias and increasing utility by federated generative modeling of medical images using a centralized adversary. *arXiv preprint arXiv:2101.07235*, 2021.

RESEARCH, M. *The global victim-perpetrator synthetic dataset*. 2022. Disponível em: <https://www.ctdatacollaborative.org/global-victim-perpetrator-synthetic-dataset>.

RIMOL, M. *Gartner Identifies Top Five Trends in Privacy Through 2024*. 2022. Gartner Press Release. Disponível em: <https://www.gartner.com/en/newsroom/press-releases/2022-05-31-gartner-identifies-top-five-trends-in-privacy-through-2024>.

ROSENBLATT, L.; LIU, X.; POUYANFAR, S.; LEON, E. de; DESAI, A.; ALLEN, J. Differentially private synthetic data: Applied evaluations and enhancements. *arXiv preprint arXiv:2011.05537*, 2020.

ROSENBLATT, L.; LIU, X.; POUYANFAR, S.; LEON, E. de; DESAI, A.; ALLEN, J. Differentially private synthetic data: Applied evaluations and enhancements. *arXiv preprint arXiv:2011.05537*, 2020.

Science and Technology Policy Office. Request for information on advancing privacy-enhancing technologies. *The Daily Journal of the United States Government*, p. 35250–35252, 2022.

SILVA, C. Laranjeira da; MACEDO, J.; AVILA, S.; SANTOS, J. dos. Seeing without looking: Analysis pipeline for child sexual abuse datasets. In: *2022 ACM Conference on Fairness, Accountability, and Transparency*. [S.l.: s.n.], 2022. p. 2189–2205.

SOLON, O. Child sexual abuse images and online exploitation surge during pandemic. *NBC News*, 2020. Disponível em: <https://www.nbcnews.com/tech/tech-news/child-sexual-abuse-images-online-exploitation-surge-during-pandemic-n1190506>.

TANG, J.; KOROLOVA, A.; BAI, X.; WANG, X.; WANG, X. Privacy loss in apple's implementation of differential privacy on macos 10.12. *arXiv preprint arXiv:1709.02753*, 2017.

TAO, Y.; MCKENNA, R.; HAY, M.; MACHANAVAJJHALA, A.; MIKLAU, G. Benchmarking differentially private synthetic data generation algorithms. *arXiv preprint arXiv:2112.09238*, 2021.

THORN. *Meet the new anti-grooming tool from Microsoft, Thorn, and our partners.* https://www.thorn.org/blog/what-is-project-artemis-thorn-microsoft-grooming, Last accessed on 2020-05-08.

TORKZADEHMAHANI, R.; KAIROUZ, P.; PATEN, B. DP-CGAN: Differentially private synthetic data and label generation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. [S.l.: s.n.], 2019. p. 98–104.

VADHAN, S. P.; CROSAS, M.; HONAKER, J. Opendp : An open-source suite of differential privacy tools. In: . [s.n.], 2019. Disponível em: <https://api.semanticscholar.org/CorpusID:198976455>.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. *Advances in neural information processing systems*, v. 30, 2017.

VITORINO, P.; AVILA, S.; ROCHA, A. A two-tier image representation approach to detecting child pornography. In: *XII Workshop de Visão Computational*. [S.l.: s.n.], 2016. p. 129–134.

WAGH, S.; GUPTA, D.; CHANDRAN, N. SecureNN: 3-party secure computation for neural network training. *Proceedings on Privacy Enhancing Technologies*, v. 3, p. 26–49, 2019.

WALONOSKI, J.; KRAMER, M.; NICHOLS, J.; QUINA, A.; MOESEL, C.; HALL, D.; DUFFETT, C.; DUBE, K.; GALLAGHER, T.; MCLACHLAN, S. Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record. *Journal of the American Medical Informatics Association*, Oxford University Press, v. 25, n. 3, p. 230–238, 2018.

Westlake, B.; Bouchard, M.; Frank, R. Comparing methods for detecting child exploitation content online. In: *2012 European Intelligence and Security Informatics Conference*. [S.l.: s.n.], 2012. p. 156–163.

WIENS, J.; SARIA, S.; SENDAK, M.; GHASSEMI, M.; LIU, V. X.; DOSHI-VELEZ, F.; JUNG, K.; HELLER, K.; KALE, D.; SAEED, M. *et al.* Do no harm: a roadmap for responsible machine learning for health care. *Nature medicine*, Nature Publishing Group, v. 25, n. 9, p. 1337–1340, 2019.

WOODBRIDGE, J.; ANDERSON, H. S.; AHUJA, A.; GRANT, D. Predicting domain generation algorithms with long short-term memory networks. *arXiv preprint arXiv:1611.00791*, 2016.

WOODBRIDGE, J.; ANDERSON, H. S.; AHUJA, A.; GRANT, D. Detecting homoglyph attacks with a siamese neural network. In: IEEE. *2018 IEEE Security and Privacy Workshops (SPW)*. [S.l.], 2018. p. 22–28.

WOODHAMS, J.; KLOESS, J. A.; JOSE, B.; HAMILTON-GIACHRITSIS, C. E. Characteristics and behaviors of anonymous users of dark web platforms suspected of child sexual offenses. *Frontiers in Psychology*, v. 12, 2021. ISSN 1664-1078. Disponível em: <https://www.frontiersin.org/articles/10.3389/fpsyg.2021.623668>.

XIE, L.; LIN, K.; WANG, S.; WANG, F.; ZHOU, J. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

XIN, B.; GENG, Y.; HU, T.; CHEN, S.; YANG, W.; WANG, S.; HUANG, L. Federated synthetic data generation with differential privacy. *Neurocomputing*, Elsevier, v. 468, p. 1–10, 2022.

XIN, B.; YANG, W.; GENG, Y.; CHEN, S.; WANG, S.; HUANG, L. Private fl-gan: Differential privacy synthetic data generation based on federated learning. In: IEEE. *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.], 2020. p. 2927–2931.

XU, L.; SKOULARIDOU, M.; CUESTA-INFANTE, A.; VEERAMACHANENI, K. Modeling tabular data using conditional gan. *arXiv preprint arXiv:1907.00503*, 2019.

XU, W.; ZHAO, J.; IANNACCI, F.; WANG, B. Ffpdg: Fast, fair and private data generation. *online preprint*, 2021.

YAN, C.; YAN, Y.; WAN, Z.; ZHANG, Z.; OMBERG, L.; GUINNEY, J.; MOONEY, S. D.; MALIN, B. A. A multifaceted benchmarking of synthetic electronic health record generation models. *Nature communications*, Nature Publishing Group UK London, v. 13, n. 1, p. 7609, 2022.

Yiallourou, E.; Demetriou, R.; Lanitis, A. On the detection of images containing child-pornographic material. In: *2017 24th International Conference on Telecommunications (ICT)*. [S.l.: s.n.], 2017. p. 1–5.

YOON, J.; DRUMRIGHT, L. N.; SCHAAR, M. V. D. Anonymization through data synthesis using generative adversarial networks (ads-gan). *IEEE journal of biomedical and health informatics*, IEEE, v. 24, n. 8, p. 2378–2388, 2020.

ZHANG, J.; CORMODE, G.; PROCOPIUC, C. M.; SRIVASTAVA, D.; XIAO, X. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)*, ACM New York, NY, USA, v. 42, n. 4, p. 1–41, 2017.

ZHANG, X.; ZHAO, J.; LECUN, Y. Character-level convolutional networks for text classification. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2015. p. 649–657.