A Dissertation

Trustworthy Clinical Decision Support Systems for Medical Trajectories

Presented to
the Faculty of the School of Engineering and Applied Sciences
University of Virginia

In Partial Fulfillment
of the requirements for the Degree
Doctor of Philosophy (Computer Science)

By

Josephine Lamp

March 2024

Abstract

The explosion of medical sensors and wearable devices has resulted in the collection of large amounts of medical trajectories. Medical trajectories are time series that provide a nuanced look into patient conditions and their changes over time, allowing for a more fine-grained understanding of patient health. It is difficult for clinicians and patients to effectively make use of such high dimensional data, especially given the fact that there may be years or even decades worth of data per patient. Clinical Decision Support Systems (CDSS) provide summarized, filtered, and timely information to patients or clinicians to help inform medical decision-making processes. Although CDSS have shown promise for data sources such as tabular and imaging data, e.g., in electronic health records, the opportunities of CDSS using *medical trajectories* have not yet been realized due to challenges surrounding data use, model trust and interpretability, and privacy and legal concerns.

This dissertation develops novel machine learning frameworks for trustworthy CDSS using medical trajectories. We define trustworthiness in terms of three desiderata: (1) **robust**—providing reliable outputs from the CDSS even when inputs are variable, irregular or missing; (2) **explainable**—providing understandable, actionable explanations for CDSS predictions to clinicians or patients; and (3) **privacy-preserving**—providing CDSS that use data without violating patients' privacy expectations. We develop interpretable machine learning frameworks that are robust to missing, irregular, variable and conflicting trajectories that directly address data and model challenges. Moreover, we develop privacy-preserving learning methodologies that allow for the safe sharing and aggregation of medical trajectories and directly address privacy challenges. We evaluate our frameworks across a wide selection of benchmarks and show that our techniques can learn valuable insights from trajectory data with high accuracy and strong privacy guarantees.

For my mom and grandma, Cindy and Lauretta.

Acknowledgements

It has been my dream to complete my Ph.D. for many years, and I have reached the end of this journey due to the kind support and help from many exceptional people. First and foremost, I would like to thank Lu Feng and Dave Evans, my amazing advisors, for their unwavering support, guidance, inspiration and insight throughout my graduate career. I am grateful to Lu for always having my back and for all of her mentoring, encouragement and support; I am constantly inspired by her passion for research and perseverance. I am grateful to Dave for always providing invaluably helpful guidance on a range of topics from research to life in general; I strive to emanate his innate curiosity, creativity and resourcefulness. I feel extremely lucky to have been mentored and guided by these two incredible researchers for many years.

I would like to thank my awesome committee members: I am incredibly grateful to Tianhao Wang for all of the brainstorming help, in-depth technical discussions and feedback he provided, and for always encouraging me, even when I asked (many, many) stupid questions. I am also especially appreciative of Sula Mazimba who always carved out time for me despite his incredibly busy clinician schedule, and thank him for believing in me and my crazy ideas. I have really cherished all of our collaboration and sincerely appreciate the poise, humbleness and kindness with which he has constantly supported me. Moreover, I thank Tinting Zhu, Miaomiao Zhang, and Rich Nguyen for their immeasurable time, feedback and support.

Many thanks to my outstanding collaborators, including Kenneth Bilchick, Prince Afriyie, and Nate Paul whose expertise and insights greatly helped the research in this dissertation. I also sincerely thank Mark Derdzinski, Joost van der Linden, Christopher Hannemann and all of the folks at Dexcom. They are some of the most intelligent, conscientious, innovative and kind-hearted people I have had the privilege to know, and they constantly inspire me; I strive to be as humble and well-rounded a human as they all are.

I would also like to thank those that blazed the trail before me and put me on the path towards my Ph.D. Specifically, to my high school computer science teacher Richard Guenther for sparking my interest in technology and CS, and to my undergraduate advisor, Carlos Rubio-Medrano for instilling in me a passion for research and helping me cultivate many essential research and writing skills. As a direct result of their efforts, I was ultimately inspired to pursue my research career.

A colossal thank you to my wonderful partner Stuart Graham for all his love and constant support and, most importantly, for always making me laugh and finding a way to recharge my creativity and inspiration, even when things were tough. Going through life with him is phenomenal and I am so grateful to have him by my side. Countless thanks to my best friend Ingy ElSayed-Aly whom I spent so many hours brainstorming, theorizing, diagramming, writing, laughing, crying and adventuring with. She is one of the kindest, most enduring and resilient people I know, and there is no one else I would rather have gone through every step of the Ph.D. together with. I would like to thank my brother Steven Lamp for his unwavering support and help in so many aspects of my life, and in particular for all his programming guidance. He is a genius and I am so lucky to have him as my little brother.

Last but not least, many thanks to numerous other family and friends who have supported me every step of the way including Erin Lanphier, Anna Spears, Megan Bottcher, my grandma Lauretta Olson, my brothers Levi and Shane, and my dad Rick Lamp. Thanks to my grandparents Bob and Jennie Lamp, who, although now gone from this world, will always be a source of inspiration in my heart. Finally, a special thanks to my mom, Cindy Gockley, who is the most hard-working, compassionate, loving, and selfless person I know. I would not be where I am today without all of her support, motivation and cheerleading, and I am so incredibly grateful for all her love and sacrifice.

Contents

List of Figures

List of Tables

List of Abbreviations

AE	**A**uto**e**ncoder
AUC	**A**rea **U**nder the **C**urve
CARNA	**C**haracterizing **A**dvanced heart failure **R**isk and hemody**NA**mic phenotypes
CDSS	**C**linical **D**ecision **S**upport **S**ystem
CGM	**C**ontinuous **G**lucose **M**onitor
DeLvTx	**De**ath, **LV**AD implementation or **h**eart **t**ransplantation
DP	**D**ifferential **P**rivacy
DP-RuL	**D**ifferentially **P**rivate **Ru**le **L**earning
DP-SGD	**D**ifferentially-**P**rivate **S**tochastic **G**radient **D**escent
GAN	**G**enerative **A**dversarial **N**etwork
GDPR	**G**eneral **D**ata **P**rotection **R**egulation
HF	**H**eart **F**ailure
HIPAA	**H**ealth **I**nsurance **P**ortability and **A**ccountability **A**ct
ICU	**I**ntensive **C**are **U**nit
LDP	**L**ocal **D**ifferential **P**rivacy
LVAD	**L**eft **V**entricular **A**ssist **D**evice
MCTS	**M**onte-**C**arlo **T**ree **S**earch
ML	**M**achine **L**earning
MSE	**M**ean **S**quared **E**rror
MVDD	**M**ulti-**V**alued **D**ecision **D**iagram
PCA	**P**rincipal **C**omponent **A**nalysis
ROC	**R**eceiver **O**perator **C**haracteristic Curve
STL	**S**ignal **T**emporal **L**ogic
T1D	**T**ype **I** **D**iabetes
XAI	E**x**plainable **A**rtificial **I**ntelligence

Chapter 1

Introduction

Recently, the explosion of medical sensors and wearable devices has enabled the collection of large amounts of medical trajectories (medical time series data), relatively quickly and unobtrusively. Medical trajectories can provide a more nuanced look into a patient's condition and how it changes over time, compared to single point of time (snapshot) measurements. As such, trajectories present an opportunity for the fine-grained tracking and understanding of disease progression and patient outcomes.

Although there have been calls to take advantage of this type of data in healthcare settings [1], it is difficult for already overburdened clinicians and patients to effectively make use of such high dimensional data, especially when there could be years or even decades of data *per patient*. In order to make optimal decisions, it is necessary to take into account a patient's entire longitudinal history, i.e., simultaneously consider all of their previous trajectories *together*, in order to obtain a holistic view of the patient state and outcomes [2, 3]. However, doing so is a nontrivial task. For example, people with diabetes who use Continuous Glucose Monitors (CGM) commonly have glucose data recorded every 5 minutes by the device, resulting in 288 longitudinal time points per day, 2,016 data points per week and 526,000 data points across 5 years for a single feature and a single patient [4]. Scaling this up to multiple features and patients quickly becomes infeasible to analyze and effectively use by hand.

Clinical Decision Support Systems (CDSS) aim to provide summarized, filtered information at appropriate times to patients or clinicians to help inform medical decision making processes (Figure 1.1) [5]. In particular, well-designed CDSS that employ machine learning (ML) technologies can learn from large, high-dimensional sets of data and succinctly return the most relevant, personalized information back to a patient or clinician, enabling them to make informed patient care decisions [6]. Example CDSS applications are shown in Figure 1.2.

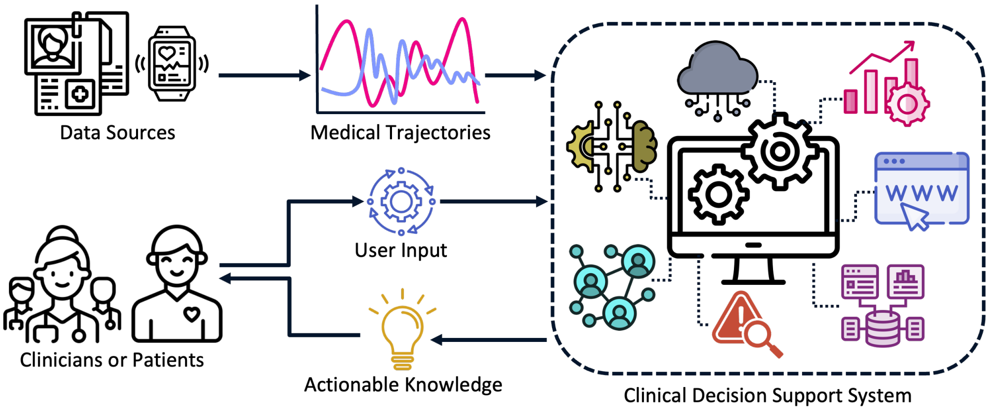


Figure 1.1: Architecture of CDSS using Medical Trajectories

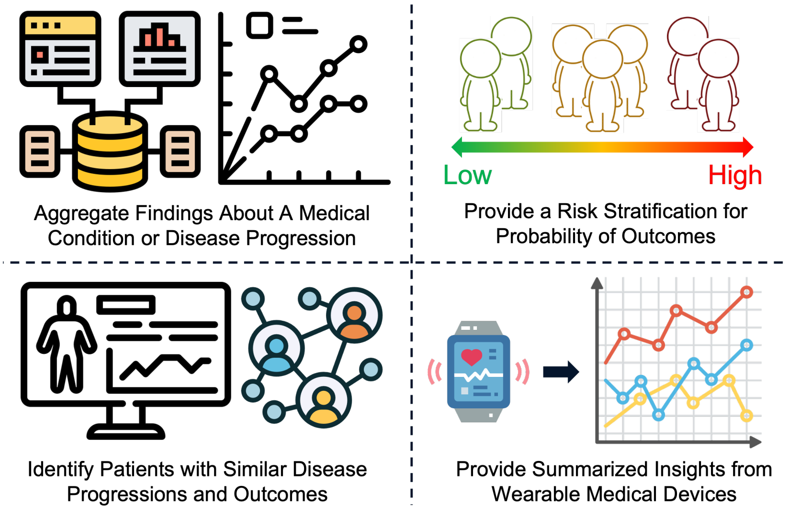

Aggregate Findings About A Medical Condition or Disease Progression
Low
High
Provide a Risk Stratification for Probability of Outcomes
Identify Patients with Similar Disease Progressions and Outcomes
Provide Summarized Insights from Wearable Medical Devices

FIGURE 1.2: Example CDSS Applications

1.1 Challenges with Trajectory Data

CDSS have shown great promise in clinical applications using tabular and imaging data, such as in electronic health records [6], but the opportunities for using medical trajectories in CDSS have not yet been realized [7]. This is due to several unaddressed challenges which we categorize into three key areas: *data challenges*, *model challenges* and *privacy challenges*. An overview of the challenges is shown in Figure 1.3.

Data Challenges. The effectiveness of the models within CDSS may be limited by issues with the input trajectory data used for training. Time series data have unique characteristics including their temporality, changes over time, and sheer volume of data. These characteristics make building models more difficult and must be carefully considered when constructing useful ML models [8]. As such, CDSS and machine learning models built for static data may not be simply transferred over and applied to trajectory data.

In addition, trajectories, and medical trajectories in particular, may be very noisy, highly variable, and contain missing values [9]. Entire features (covariates) may be missing; for instance, given a dataset with trajectories of heart rate, blood pressure and lab values, a specific patient may have no data recorded for heart rate. Alternatively, there may be missing recordings at particular timepoints; for example, for a trajectory recorded at 5 timepoints, the heart rate value may be missing for a patient at timepoint 2 and 5. Moreover, these trajectories may also be recorded at different intervals and for a different number of times based on a patient's disease progression, resulting in trajectories of varied length and periodicity (e.g., across a 3 year period, one patient may have 3 serial echocardiograms, while another may have 40 serial echos).

Furthermore, there may be conflicting inferences among feature trajectories [8]. For example, a patient may have a decreasing heart rate trajectory, associated with poorer outcomes, and an increasing blood pressure trajectory, indicative of better outcomes. CDSS models must be able to handle the innate patient variability in states and presentation of conditions, as well as any potentially conflicting trajectory inferences in order to provide a *holistic* view of the patient, enabling patients or clinicians to easily determine the best course of action or treatment decision.

Model Challenges. CDSS need to be explainable, e.g., clearly explain how the system came to a decision, in order to promote trust and reliable usage of CDSS by

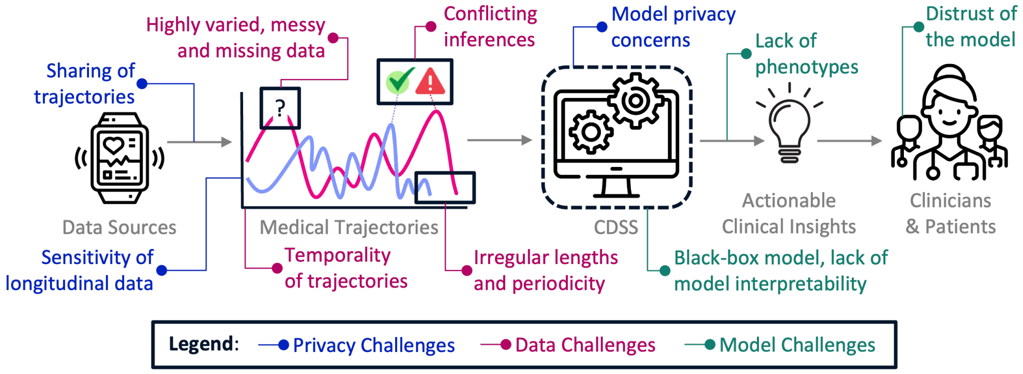

Highly varied, messy and missing data
Conflicting inferences
Model privacy concerns
Distrust of the model
Lack of phenotypes
Sharing of trajectories
?
Data Sources
Medical Trajectories
CDSS
Actionable Clinical Insights
Clinicians & Patients
Sensitivity of longitudinal data
Temporality of trajectories
Irregular lengths and periodicity
Black-box model, lack of model interpretability
Legend: Privacy Challenges Data Challenges Model Challenges

FIGURE 1.3: Summary of Challenges for CDSS using Trajectory Data

clinicians and patients [7]. However, developing ML-based CDSS that are explainable remains a large challenge [10], especially given the high dimensionality of trajectories which makes it even more difficult to explain model decisions or predictions [11, 12]. This is further exacerbated for trajectories because the best performing models for time series tend to be the least explainable (e.g., LSTM and RNN models that are black boxes) [3]. Providing *phenotypes*, sets of features and their thresholds that characterize a condition, is a helpful method to add explainability to models and succinctly describe the criteria used to make an output decision. However, providing temporal phenotypes (e.g., a peak as compared to a declining slope in one feature may indicate different things about an outcome) is an open challenge [13, 14].

Privacy Challenges. Unsurprisingly, there are well-documented privacy and legal concerns surrounding the use of sensitive medical trajectories [15]. For example, there are legal stipulations regarding the sharing of such data around the globe, including the Health Insurance Portability and Accountability Act (HIPAA) in the USA, the Personal Information Protection and Electronic Documents Act (PIPEDA) in Canada, and the Data Protection Directive and General Data Protection Regulation (GDPR) in Europe. Medical trajectories are particularly challenging to work with given their increased sensitivity due to the longitudinal nature of this type of data; i.e., a lot more information may be leaked about a specific patient because their patterns of behavior may be recorded at a very granular level across a long period of time [16, 17].

These privacy and legal stipulations may seriously limit the ability to train, deploy or share trajectory data and CDSS models [2, 5]. The sharing of medical trajectories is important to enable clinical and technological scientific advancements, but in practice time series are rarely shared due to these concerns. For example, there are no large, open source databases containing granular longitudinal traces from people with diabetes. This majorly limits researchers in the development of diabetes technology including artificial insulin delivery systems which can drastically improve a patient's quality of life [17]. As such, there is a need for mechanisms that allow for the safe sharing of medical trajectories.

Furthermore, sharing trained CDSS models enables their broader use and adoption, resulting in increased promise to improve patient outcomes on a population level. However, similar to traces, models are rarely shared due to privacy concerns, such as concerns about the models memorizing training data and disclosing sensitive patient information [5]. Given this, there is a need for the development of privacy-preserving CDSS models that are able to learn from sensitive trajectories and be shared safely, with reduced privacy concerns.

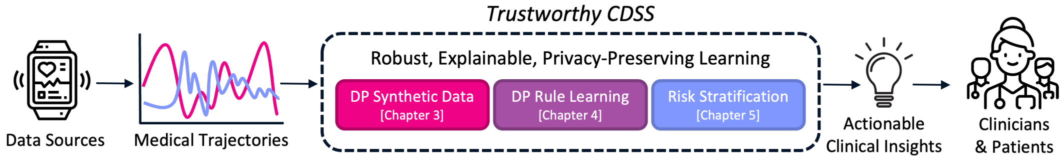

Trustworthy CDSS
Robust, Explainable, Privacy-Preserving Learning
DP Synthetic Data [Chapter 3]
DP Rule Learning [Chapter 4]
Risk Stratification [Chapter 5]
Data Sources
Medical Trajectories
Actionable Clinical Insights
Clinicians & Patients

FIGURE 1.4: Dissertation Overview

1.2 Dissertation Overview

In this dissertation, the objective is to develop trustworthy CDSS for medical trajectories that address the aforementioned data, model and privacy challenges. We define trustworthiness in terms of three desiderata: **robust**, **explainable** and **privacy-preserving**. *Robust* indicates the CDSS always provides reliable outputs, even when inputs are variable, irregular or missing. *Explainable* indicates the CDSS provides understandable, actionable explanations for model predictions, that are easy to understand by nontechnical clinicians or patients. Lastly, *privacy-preserving* indicates the CDSS use input trajectory data in ways that do not violate patients' privacy expectations.

Dissertation Summary. *This dissertation develops novel robust, explainable and privacy-preserving machine learning frameworks for trustworthy trajectory-based CDSS* (Figure 1.4). We develop interpretable machine learning frameworks for trajectories that are robust to missing, irregular, variable and potentially conflicting data, which specifically address the data and model challenges. In addition, we develop privacy-preserving learning methodologies for trajectory-based CDSS, that allow for the safe sharing and aggregation of medical trajectories and directly address privacy challenges.

1.2.1 Contributions

We develop three different robust, explainable and privacy-preserving machine learning frameworks. First, we introduce an approach for privately generating synthetic univariate trajectories, enabling their safe sharing. Next, we introduce a framework to privately learn population trajectories represented in rule-based structures, such as those commonly used in electronic health records, facilitating private model learning in distributed settings. Finally, we introduce an approach for risk stratifying patient outcomes and providing explainable patient phenotypes, enabling timely triage and allocation of life-saving therapies. Due to the interdisciplinary nature of this research, this work presents contributions to multiple disciplines including Computer Science, Biomedical Informatics and Clinical application areas, both in general and specifically for Diabetes and Heart Failure. These frameworks are summarized as follows.

Differentially-Private Synthetic Glucose Traces (GlucoSynth, Chapter 3). Sharing medical time series can facilitate research and therapy development, but is hindered by serious privacy and legal concerns with sharing longitudinal time series data in medical contexts [17]. One solution to this problem is to generate a set of synthetic traces from the original traces such that the synthetic data may be shared publicly in place of the real ones with significantly reduced privacy and legal concerns. This project focuses on the problem of generating high-quality, privacy-preserving synthetic glucose traces, a task which generalizes to other time series sources and application domains, including activity sequences, inpatient events, hormone traces and cyber-physical systems. Previous methods for time series synthesis, e.g., [18, 19, 20], suffer from one or more of the following issues when applied to glucose traces: 1) they do not generate realistic synthetic glucose traces—in particular, they produce physiologically-impossible phenomenon in the traces; 2) they require additional information (features, metadata or labels) to guide the model learning which are often not available; 3) they do not provide any privacy guarantees, or, in order to uphold a strong formal privacy guarantee, severely degrade the utility (e.g., quality) of the synthetic data.

We develop GlucoSynth, a novel privacy-preserving Generative Adversarial Network (GAN) framework to generate high-quality, private synthetic glucose traces. The core intuition behind this approach is to conserve relationships amongst motifs (glucose events) within the traces, in addition to typical temporal dynamics. Moreover, our framework incorporates differential privacy mechanisms [21] to provide strong formal privacy guarantees. GlucoSynth presents the following contributions:

- We formalize the concept of motifs and define a notion of *motif causality*, inspired from Granger causality [22], which characterizes relationships amongst sequences of motifs within time series traces.
- We build a novel GAN framework that is trained to optimize motif causality within the traces in addition to temporal dynamics and distributional characteristics of the data.
- We integrate differential privacy into the framework, which provides an intuitive bound on how much information may be disclosed about any individual in the dataset, allowing the GlucoSynth model to be trained with privacy guarantees.
- We present a comprehensive evaluation using 1.2 million glucose traces from individuals with diabetes collected across 2022, showcasing the suitability of our model to outperform all previous models and generate high-quality synthetic glucose traces with strong privacy guarantees.

Differentially-Private Rule Learning (DP-RuL, Chapter 4). Many distributed CDSS rely on logic-based learning systems [23], in which structured *rules* are used to make decisions due to their increased expressivity (diverse representations of medical associations), dual understandability by humans and machines (e.g., using a rule grammar), and increased explainability which promotes user trust in the system [24, 25]. Serious privacy concerns arise with the use of patient data in CDSS, especially those deployed in third-party health applications since they are not covered by HIPAA [15, 26]. Given these concerns, the goal of this project is to learn a population ruleset representative of the local client rule structures, while preserving the privacy of individuals involved in the rule collection. Local Differential Privacy (LDP) is a paradigm well suited to the distributed framework deployed for many CDSS since individual users each perturb their own data before it is collected and aggregated [27]. Previous work has developed differentially-private methods for distributed learning in various settings including finding new frequent strings [28], discovering keystroke data [29, 30], text mining [31], frequent item mining [32, 33, 34] and data mining personal information [35]. However, no previous work has developed LDP methods for learning logic-based rule structures or for CDSS applications, and none of the methods developed for these other settings can be directly applied to provide an adequate solution to the privacy rule discovery problem.

We present DP-RuL (Differentially Private Rule Learning, the first locally differentially-private framework to learn population rulesets with high coverage and clinical utility for logic-based CDSS. Specifically, DP-RuL presents the following contributions:

- We develop a novel Rule Discovery Protocol, which uses a method based on Monte-Carlo Tree search (MCTS) to search a rule grammar in a structured way and find population rules contained by the clients. The protocol follows the traditional MCTS steps (Selection, Expansion, Querying, and Backpropagation). To provide LDP, we adapt the querying phase to use randomized response. To find clinically useful rules, we adapt the MCTS scoring function, which guides the search process about which subtrees to continue searching down, to use privacy-preserving estimates of the number of clients who have rules that match a template rule structure in the grammar. By guiding the searching based on client responses, and taking advantage of the rule grammar, we are able to efficiently learn population rulesets, including rules with complex structures.
- Each query in the Rule Discovery Protocol is allocated a privacy loss budget that determines the randomized response noise used in the response. We develop an adaptive budget allocation method which dynamically provisions the privacy loss budget. The intuition behind this method is to find the minimum budget per query to gain enough information to determine whether a node should be further explored.
- We evaluate our protocol on three clinical datasets from different medical domains and find that we are able to learn population rulesets with high coverage and clinical utility, even at low privacy loss budgets.
- Our framework is open source, available at: `https://github.com/jozieLamp/DP_Rule_Learning_for_CDSS`.

Interpretable Learning for Risk Stratification (CARNA, Chapter 5). Risk stratification is the process by which patients are grouped based on their disease condition in order to make timely treatment decisions [36]. As a specific example, identifying high risk advanced heart failure (HF) patients early on in the care continuum is critical for timely allocation of advanced life-saving therapies such as device implantation or transplant allocation. Due to high variability in patient conditions and disease complexity, determining patient risk involves a challenging, multi-faceted decision making process that places a high burden on clinicians [37]. Hemodynamic assessments provide measures of cardiovascular function and can enhance understanding of HF trajectories [38], but it is difficult to obtain a comprehensive picture of patient state from these as they may be variable, conflicting, and missing [39]. Previous methods for risk stratification [40, 41, 42, 43] use statistical or naive models which are difficult to optimize and prone to bias. Moreover, no previous models integrate invasive hemodynamics or contain mechanisms to handle missing trajectory data. Machine learning (ML) models present a promising opportunity to outperform traditional risk assessment methods, especially when dealing with large, high-dimensional data [44], but they remain unpopular in clinical use due to modest model performance and issues with model interpretability [45].

To address these limitations, we develop CARNA (Characterizing Advanced heart failure Risk and hemodyNAmic phenotypes),[1] an explainable ML framework that learns risk scores to predict the probability of patient outcomes (mortality and rehospitalization), and outputs descriptive patient phenotypes, i.e., sets of features and

[1] So named for the Roman healing goddess who presides over the heart.

their thresholds, that characterize each predicted risk score. CARNA has the following contributions:

- We develop a general purpose risk stratification and phenotyping framework that can handle variable, missing and conflicting trajectories. Although applied for HF risk stratification, this framework can be applied to other diseases and medical applications. To provide a risk categorization, we first stratify risk into categories using a hierarchical clustering algorithm. We harness the explainability and expressivity of machine learned Multi-Valued Decision Diagrams (MVDDs) to learn risk scores using the outputted risk categories from the clustering. MVDDs represent logical functions in directed, acyclic graphs where nodes represent features, edges represent logical operators ("and", "or") with parameter threshold values, and leaf nodes represent the final score classification [46]. Due to their use of logical operators, MVDDs can handle missing data, as multiple substitutable features may contribute to the same score prediction. Moreover, the "path" through the MVDD may be returned to provide a descriptive patient phenotype that characterizes the score.

- We provide robust validation of the CARNA framework using four independent HF cohorts, and compare them with six established HF risk scores and three traditional ML models. The CARNA models achieve high performance and outperform all benchmarks across metrics including Accuracy, Sensitivity, Specificity and AUC.

- We provide an extensible, open-source tool implementation that includes a deployed web server, which provisions live risk score prediction for ease of clinical use: https://github.com/jozieLamp/CARNA.

- This framework uses a novel integration of invasive and noninvasive hemodynamics. Direct findings from this framework about advanced HF (e.g., relations among hemodynamics and other features) may inform future clinical research studies.

- We introduce a new clinical paradigm for HF risk stratification, in which predicting risk categories is used over singular binary events (as is done in traditional risk stratification). This approach may facilitate complementary evidence-based modeling of "risk–benefit" trade-offs when it comes to the challenging shared decision discussions between clinicians and patients concerning HF prognostication and the timing of advanced heart failure therapies.

Chapter 2

Background

In this chapter, we briefly discuss relevant cross-cutting background topics, including differential privacy (Section 2.1), explainable machine learning in CDSS (Section 2.2) and risk stratification (Section 2.3). In addition, we introduce two CDSS application areas used as running examples throughout the dissertation (Section 2.4). Detailed discussions of background and related work tailored to each specific project is available in the succeeding chapters.

2.1 Differential Privacy

Differential Privacy (DP) [21, 47] is a formal notion of privacy that provides an intuitive bound on the amount of information that can be disclosed about any individual in a dataset. A randomized algorithm $\mathcal{M}$ satisfies (ϵ, δ)-differential privacy if, for all datasets D_1 and D_2 differing by at most a single element, and all S $\subseteq$ Range($\mathcal{M}$),

$$Pr[\mathcal{M}(D_1) \in S] \leq e^{\epsilon}\ Pr[\mathcal{M}(D_2) \in S]+\delta$$

ϵ and δ are important privacy parameters determining the *privacy loss budget*, which dictates the level of privacy guaranteed by the algorithm; smaller values indicate stronger privacy.

Local Differential Privacy. Local Differential Privacy (LDP) is a DP paradigm well suited to the distributed framework deployed for many CDSS systems, as, in this method, individual users each perturb their own data before it is collected and used for population-level aggregation [27]. In such a setting, a centralized, untrusted server, S, wishes to compute some summary statistic y from n individual clients' data records, $\{x_1, ..., x_n\}$. Each client locally perturbs the requested data x_i before sending it to the server. In this way, the server (or any other adversary who has access to a client's responses or the server's output) cannot determine any individual's data with high confidence.

DP in Medicine. There is a large body of work on differential privacy, including large-scale deployments of aggregate statistical collection using differential privacy by Apple [48, 49], Google Chrome [50, 28] and Mac OS [51]. There have also been many DP models for record and tabular medical data, including aggregating sensitive statistical features from electronic health records, collecting one dimensional data, sharing entire tabular health datasets, as well as the privatizing and sharing of genomic data, such as those from genome-wide association studies [52, 53]. Additionally, work has developed privacy-preserving machine learning methods for a range of models including decision trees, graph neural networks, boosting, convolutional neural networks (often applied to learn predictions from health data) [54, 35, 55]. In summary, there are many works applying DP to tabular *health data* for data publishing and data mining [52]. However, applications of DP for medical trajectories are much more limited and no previous work has developed DP models for CDSS-specific applications, or directly implemented DP mechanisms within deployed CDSS.

2.2 Explainable AI in Medicine

Explainable AI (XAI) encompasses a wide range of techniques to provide interpretable explanations about a model's choices, such as through the use of feature maps, textual annotations, local, feature or example explanations, and model simplification methods [56]. XAI methods have been used in a range of healthcare applications [57], including for predicting heart failure incidence [58]. Adding explainability to a model often comes with a trade-off: the XAI methods may be complicated to implement, inefficient (i.e., it takes much longer to train an explainable model), or have stability and reliability issues [59]. In addition, although these methods are good for understanding more about the model (e.g., visualizing at a high level how combinations of features contributed to a prediction), they are not always conducive to quick decision making in high stress environments. For example, interpreting feature maps or understanding textual explanations is nontrivial; one may need time to decipher the explanation and determine its use (e.g., figure out how a risk score was computed).

Rules and Logic-Based CDSS. One of the most straightforward XAI methods is to output model decisions using rules. Many CDSS rely on logic-based learning systems [23], in which structured *rules* are used to make decisions. One of the main motivations for logic-based learning in CDSS is because the easy to understand rules promote dual understandability by humans and machines (e.g., using a rule grammar), and increased explainability (no black box models,) which promotes user trust in the system [24, 25]. Moreover, these learners are quite expressive, allowing for diverse representations of medical associations. Even with the proliferation of deep learning and generative ML, rule-based learners are still extremely common in clinical applications; indeed, some deep learning frameworks actually use a rule-based output layer or use an ensemble that includes a rule-based learner to explain model predictions, providing some interpretability and increasing trust of the overall system [60, 61].

2.3 Risk Stratification

Many CDSS, especially those used in chronic disease, use *risk stratification*. Risk stratification is the process by which patients are grouped or stratified based on their disease condition and other health factors [36]. This process is commonly completed to understand patient risk of adverse outcomes, such as mortality or rehospitalization, in order to make timely treatment decisions. As part of the risk stratification process, a succinct patient phenotype may be provided to characterize the predicted outcome risk in an understandable way. A *patient phenotype* is a particular set of features and their thresholds or patterns that characterize a condition or outcome. For example, a patient with a high risk of mortality may be characterized by a phenotype with an age > 72, a decreasing systolic blood pressure trajectory, an increasing mean arterial pressure trajectory, and a cardiac output trajectory that stays below a threshold of 3.7 L/min.

Risk Stratification in Heart Failure. As a specific application area, we look at risk stratification for heart failure. Identifying patients at risk of heart failure (HF) is critical in order to provision correct treatment decisions and implement advanced, life-saving therapies at the correct time [62]. For high-risk heart failure patients, analyzing trajectories obtained from procedures such as hemodynamic assessments and imaging systems, e.g., echocardiograms, can allow clinicians to better understand the patient state thereby allowing them to make more informed triage decisions earlier in the care continuum. However, HF risk assessment using such trajectories is a complex, multi-faceted decision-making process [37]. Related to the challenges mentioned in Chapter 1, this is because it is hard to obtain a holistic, overall understanding of the patient state since trajectories may present in ways that are associated with conflicting outcomes, there may be many missing features, and the trajectories themselves may be noisy. Furthermore, these trajectories may be recorded at different intervals and for a different number of times based on a patient's disease progression (e.g., across a 3 year period, one patient may have 3 serial echocardiograms, while another may have 40 serial echos), resulting in trajectories of varied length and periodicity.

2.4 Clinical Case Studies

Although there are many types of CDSS use cases and application areas, in particular this dissertation focuses on two themes across all the developed frameworks: chronic disease management and facilitating large scale clinical research.

Chronic Disease Management. The first use case enables the analysis of data from patients with chronic diseases who are being monitored over a long time period in outpatient settings. Data being recorded at high temporal granularity (e.g., multiple times per day for periods of months and years) present obvious privacy concerns, as they can disclose intimate details about a patient's life. However, deriving patterns of patient behaviors that result in different outcome states (e.g., a better or worse patient state of being for a chronic disease) can support clinical research studies related to disease progression, improved treatment guidance, triage of patient status, etc. In particular, being able to aggregate personalized data patterns collected from individual patients to derive population level findings is especially relevant for the study of rare disease conditions. The ability to extrapolate common details or symptoms from the patients to improve diagnoses and treatment protocols (which are often lacking for rare diseases), while protecting the confidentiality of the patients suffering from such rare diseases, is an important opportunity for CDSS. In this dissertation, we use the specific CDSS example application areas of advanced heart failure (HF) and Type I Diabetes (T1D).

Facilitating Large Scale Clinical Research on Distributed Clinical Data. The second use case looks at enabling the analysis of large amounts of distributed clinical data. For example, this may include retrospective studies using inpatient hospital data after patients have been discharged, across multiple organizations and outside patient care needs, or for studies collected across individuals collecting personalized data from wearables and other personal sensors. Currently, conglomerating and then studying data across multiple (distributed) centers is very difficult due to HIPAA constraints and patient privacy concerns [2, 5, 15]. However, analyzing this data for large scale population studies is useful for medical researchers to learn more about various disease and conditions progressions, triage and diagnostic choices in the hospital.

In addition to the application areas of heart failure and T1D as mentioned previously, we also look at aggregating inpatient data research across multiple hospitals in two specific examples of Intensive Care Units (ICU) and Sepsis. A common goal of CDSS using ICU data is to understand predictors of clinical deterioration in the

ICU. Deterioration refers to a patient's quick onset of a declining physical state that may result in life-threatening outcomes such as death. Symptoms of deterioration are highly variable between patients, especially because the condition may occur with little to no warning. Although deterioration can be characterized by physical indicators (from hospital sensors such as SpO_2, ECG, heart rate etc.) these measures are highly variable between patients, and, because deterioration usually occurs extremely quickly with little to no warning, it is very hard to predict which patients may deteriorate. In this context, the goal of a CDSS is to aggregate patterns across patients from multiple health organizations in order to predict whether or not a patient may deteriorate in a timely manner such that clinicians can take an action to try and prevent the deterioration.

In the sepsis case, users of CDSS might wish to understand and predict the onset and progression of Sepsis. Sepsis is the body's extreme reaction to an infection it already had and is life-threatening as it can cause a cascade of biological damages such as tissue damage, shock, and organ failure. There is still debate about what actually causes the onset of sepsis. Similar to ICU deterioriation, sepsis occurs extremely quickly, and presents in highly variable ways between patients. Due to this variability, and wide range in symptom presentation based on patient underlying health conditions (e.g., why they were originally admitted to the hospital), age, sex etc., a CDSS would wish to learn aggregated patterns characterizing sepsis onset from similar patient cohorts. In this way, being able to better understanding and predict the timing and presentation of symptoms leading to sepsis based on individual patient characteristics can allow for timely, life-saving hospital interventions.

It is our intention that the frameworks developed in this dissertation support both of these use cases including the sharing of clinical trajectories collected from wearables in outpatient settings as well as the aggregation of population level patterns of behaviors that can inform disease understanding and clinical management and treatment decisions. In the next chapters, we describe each of these frameworks in detail.

Chapter 3

Differentially-Private Synthetic Glucose Traces (GlucoSynth)

The sharing of medical time series data can facilitate therapy development.[1] As a motivating example, sharing glucose traces can contribute to the understanding of diabetes disease mechanisms and the development of artificial insulin delivery systems that improve people with diabetes' quality of life. Unsurprisingly, there are serious legal and privacy concerns (e.g., HIPAA, GDPR) with the sharing of such granular, longitudinal time series data in a medical context [17]. One solution is to generate a set of synthetic traces from the original traces. In this way, the synthetic data may be shared publicly in place of the real ones with significantly reduced privacy and legal concerns.

This project focuses on the problem of generating high-quality, privacy-preserving synthetic glucose traces, a task which generalizes to other time series sources and application domains, including activity sequences, inpatient events, hormone traces and cyber-physical systems. Specifically, we focus on long (over 200 timesteps), bounded, univariate time series glucose traces. We assume that available data does not have any labels or extra information including features or metadata, which is quite common, especially in diabetes. Continuous Glucose Monitors (CGMs) easily and automatically send glucose measurements taken subcutaneously at fixed intervals (e.g., every 5 minutes) to data storage facilities, but tracking other sources of diabetes-related data is challenging [63]. We characterize the quality of the generated traces based on three criteria— synthetic traces should (1) conserve characteristics of the real data, i.e., glucose dynamics and control-related metrics (*fidelity*); (2) contain representation of diverse types of realistic traces, without the introduction of anomalous patterns that do not occur in real traces (*breadth*); and (3) be usable in place of the original data for real-world use cases (*utility*).

Generative Adversarial Networks (GANs) [64] have shown promise in the generation of time series data. However, previous methods for time series synthesis, e.g., [18, 19, 20], suffer from one or more of the following issues when applied to glucose traces: 1) surprisingly, they do not generate realistic synthetic glucose traces – in particular, they produce human physiologically impossible phenomenon in the traces; 2) they require additional information (features, metadata or labels) to guide the model learning which are not available for our traces; 3) they do not include any privacy guarantees, or, in order to uphold a strong formal privacy guarantee, severely degrade the utility of the synthetic data.

Generating high-quality synthetic glucose traces is a difficult task due to the innate characteristics of glucose data. Glucose traces can be best understood as sequences

[1] This chapter is based on: Lamp, Josephine, Mark Derdzinski, Christopher Hannemann, Joost Van der Linden, Lu Feng, Tianhao Wang, and David Evans. "GlucoSynth: Generating Differentially-Private Synthetic Glucose Traces." Advances in Neural Information Processing Systems 36 (2024).

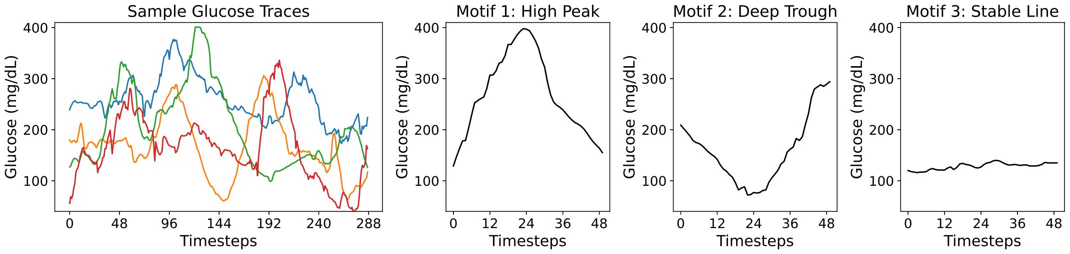
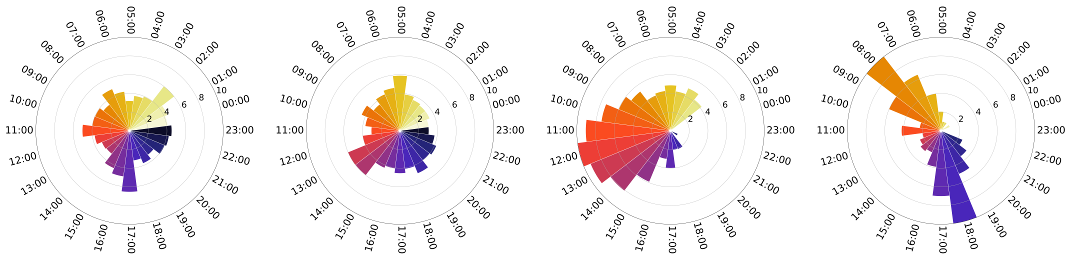


Figure 3.1: Example Real Glucose Traces and Glucose Motifs from our Dataset.

of events, which we call *motifs*, shown in Figure 3.1, and they are more event-driven than many other types of time series. As such, a current glucose value may be more influenced by an event that occurred in the far past compared to values from immediate previous timesteps. For example, a large meal eaten earlier in the day (30-90 minutes ago) may influence a patient's glucose more than the glucose values from the past 15 minutes. As a result, although there is some degree of temporal dependence within the traces, *only* conserving the immediate temporal relationships amongst values at previous timesteps does not adequately capture the dynamics of this type of data. In particular, we find that the main reason previous methods fail is because they may not sufficiently learn event-related characteristics of glucose traces.

Contributions. We present *GlucoSynth*, a privacy-preserving GAN framework to generate synthetic glucose traces. The core intuition behind our approach is to conserve relationships amongst motifs (events) within the traces, in addition to the typical temporal dynamics contained within time series. We formalize the concept of motifs and define a notion of *motif causality*, inspired from Granger causality [22], which characterizes relationships amongst sequences of motifs within time series traces (Section 3.2). We define a local motif loss to first train a motif causality block that learns the motif causal relationships amongst the sequences of motifs in the real traces. The block outputs a motif causality matrix, that quantifies the causal value of seeing one particular motif after some other motif. Unrealistic motif sequences (such as a peak to an immediate drop in glucose values) will have causal relationships close to 0 in the causality matrix. We build a novel GAN framework that is trained to optimize motif causality within the traces in addition to temporal dynamics and distributional characteristics of the data (Section 3.3). Explicitly, the generator computes a motif causality matrix from each batch of synthetic data it generates, and compares it with the real causality matrix. As such, as the generator learns to generate synthetic data that yields a realistic causal matrix (thereby identifying appropriate causal relationships from the motifs), it implicitly learns not to generate unrealistic motif sequences. We also integrate differential privacy (DP) [21] into the framework (Section 3.4), which provides an intuitive bound on how much information may be disclosed about any individual in the dataset, allowing the GlucoSynth model to be trained with privacy guarantees. Finally, in Section 3.5, we present a comprehensive evaluation using 1.2 million glucose traces from individuals with diabetes collected across 2022, showcasing the suitability of our model to outperform all previous models and generate high-quality synthetic glucose traces with strong privacy guarantees.



(a) Glucose Motif 1 (b) Glucose Motif 2 (c) Temporal Motif 1 (d) Temporal Motif 2

FIGURE 3.2: Temporal Distributions of Sample Motifs. Each radial graph displays the temporal distribution of a motif; there are 24 radial bars from 00:00 to 23:00, and each segment displays the percentage of motif occurrences by each hour. Glucose motifs 1 and 2 are from Fig. 3.1; they are not temporally-dependent and show up across the day. Temporal motifs 1 and 2 are from a cardiology dataset [65].

3.1 Preliminaries

3.1.1 Motifs

Glucose (and many other) traces can be best understood as sequences of events or *motifs*. Motifs characterize phenomenon in the traces, such as peaks or troughs. We define a *motif*, μ, as a short, ordered sequence of values (v) of specified length τ, $\mu = [v_i, v_{i+1}, \ldots, v_{i+\tau}]$ and σ is a tolerance value to allow approximate matching (within σ for each value). Some examples of glucose traces and motifs are shown in Figure 3.1. We denote a set of n time series traces as $X = [x_1, ..., x_n]$. Each time series may be represented as a sequence of motifs: $x_i = [\mu_{i_1}, \mu_{i_2} ...]$ where each i_j gives the index of the motif in the set that matches $x_{i_{j\cdot\tau}}, ...x_{i_{(j+1)\cdot\tau-1}}$. Given the motif length τ, the motif set is the union of all size-τ chunks in the traces. This definition is chosen for a straightforward implementation but motifs can be generated in other ways, such as through the use of rolling windows or signal processing techniques [66, 67]. Motifs are pulled from the data such that there is always a match from a trace motif to a motif from the set (if multiple matches, the closest one is chosen).

3.1.2 Glucose Dynamics (Why Standard Approaches Fail)

We first present a study of the characteristics of glucose data in order to motivate the development of our framework. Although there are general patterns in sequences of glucose motifs (e.g., motif patterns corresponding to patients that eat 2x vs. 3x a day), individual glucose motifs are typically not time-dependent, as illustrated in Figure 3.2. The radial graphs display the temporal distribution of the first two glucose motifs from Figure 3.1 and two temporally-dependent motifs from a cardiology dataset [65]. There are 24 radial bars from 00:00 to 23:00 for each hour of the day, and the bar value is the percentage of total motif occurrences at that hour across the entire dataset (i.e., value of 10 would indicate that 10% of the time that motif occurs during that hour). Note that the glucose motifs show up fairly evenly *across* all hours of the day whereas the motifs from the cardiology dataset have shifts in their distribution and show up frequently at *specific* hours of the day. The lack of temporal dependence in glucose motifs is likely due to the diverse patient behaviors within a patient population. Glucose in particular is highly variable and influenced by many factors including eating, exercise, stress levels, and sleep patterns. Moreover, due to innate variability within human physiology, motif occurrences can differ even for the *same* patient across weeks or months. These findings indicate that only conserving the temporal relationships within glucose traces (as many previous methods do) may not be sufficient to properly learn glucose dynamics and output realistic synthetic traces.

3.1.3 Granger Causality

Granger causality [22] is commonly used to quantify relationships amongst time series without limiting the degree to which temporal relationships may be understood as done in other time series models, e.g., pure autoregressive ones. In this framework, an entire system (set of traces) is studied *together*, allowing for a broader characterization of their relationships, which may be advantageous, especially for long time series. We define $x_t \in \mathbb{R}^n$ as an n-dimensional vector of time series observed across n traces and T timesteps. To study causality, a vector autoregressive model (VAR) [68] may be used. A set of traces at time t is represented as a linear combination of the previous K lags in the series: $x_t = \sum_{k=1}^{K} A^{(k)} x_{t-k} + e_t$ where each $A^{(k)}$ is a $n \times n$ dimensional matrix that describes how lag k affects the future timepoints in the series' and e_t is a zero mean noise. Given this framework, we state that time series q does not *Granger-cause* time series p, if and only if for all k, $A^{(k)}_{p,q} = 0$. To better represent nonlinear dynamics amongst traces, a nonlinear autoregressive model (NAR) [69], g, may be defined, in which $x_t = g\left(x_{1_{<t}}, ..., x_{n_{<t}}\right) + e_t$ where $x_{p_{<t}} = \left(x_{p_1} ..., x_{p_{t-1}}, x_{p_t}\right)$ describes the past of series p. The NAR nonlinear functions are commonly modeled jointly using neural networks.

3.2 Motif Causality

Using Granger causality as defined would overwhelm the generator with too much information, resulting in convergence issues for the GAN. Instead of looking at traces comprehensively, we need a way to *scope* how the generator understands relationships between time series. To this end, we aim to use the same intuition developed from Granger causality, namely developing an understanding of relationships comprehensively using less stringent temporal constraints, but scope these relationships specifically in terms of *motifs*. Therefore, we develop a concept of *motif causality* which, by learning causal relationships amongst sequences of motifs, allows the generator to learn realistic motif sequences and produce high quality synthetic traces as a result.

3.2.1 Extending Granger Causality to Motifs

In order to quantify the relationships amongst sequences of motifs to best capture glucose dynamics, we extend the idea of Granger causality to work with motifs. Given a motif set with m motifs, we build a separate (component) model, called a *motif network* in our method, for each motif, resulting in m motif networks. For a single motif μ_i at time t, μ_{i_t}, we define a function g_i specifying how motifs in previous timesteps are mapped to that motif: $\mu_{i_t} = g_i\left(\mu_{1_{<t}}, ..., \mu_{m_{<t}}\right) + e_{i_t}$ where $\mu_{j_{<t}} = \left(\mu_{j_1} ..., \mu_{j_{t-1}}, \mu_{j_t}\right)$ describes the past of motif μ_j. The output of g_i is a vector, which is added to the noise vector e_{i_t}. Essentially, we define motif μ_i in terms of its relationship to past motifs. The g_i function takes in some *mapping* that describes how motifs in previous timesteps are mapped to the current motif μ_{i_t}. The mapping is not specified in this notation, and could be defined in many different ways. In our case, we instantiate g_i using a single-layer LSTM, described next.

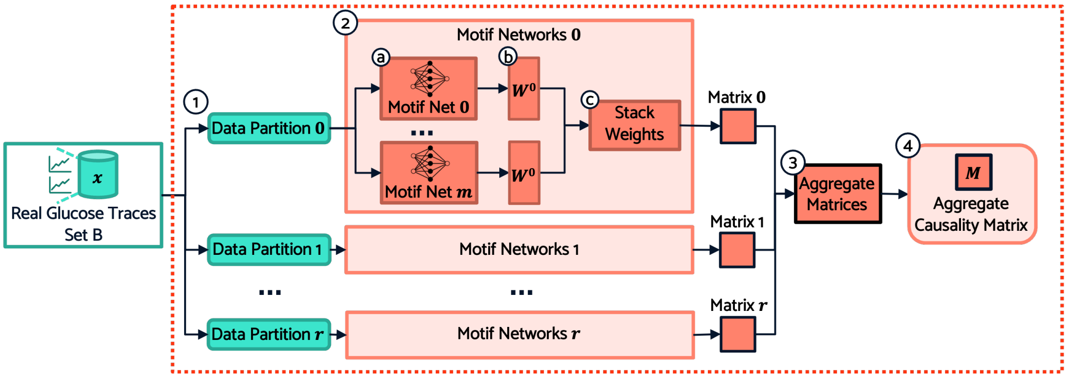


FIGURE 3.3: Motif Causality Block.

A g_i function for each motif μ_i in the motif set is modeled using a motif network with a single-layer RNN architecture. For a RNN predicting a single component motif, let $h_t \in \mathbb{R}^m$ represent the m-dimensional hidden state at time t. This represents the historical context of the motifs in the series for predicting a component motif at time t, μ_{i_t}. At time t, the hidden state is updated: $h_t = g_i(h_{t-1}) + e_{i_t}$. g_i here is the function describing how motifs in previous timesteps are mapped to the current motif, and is modeled (instantiated) as a single-layer LSTM as they are good at modeling long, nonlinear dependencies amongst traces [70]. The output for a motif μ_i at time t, μ_{i_t} can be obtained by a linear decoding of the hidden state, $\mu_{i_t} = W^o h_t + e_{i_t}$, where W^o is a matrix of the output weights. These weights control the update of the hidden state and thereby control the influence of past motifs on this component motif. Essentially, this function learns a weighting that quantifies how helpful motifs in previous timesteps are for predicting the specified motif μ_i at time t. We note that we define causality in this way based on how Granger causality models such relationships, which is different from traditional causality models.

If all elements in the jth column of W^o are zero ($W^o_{:j} = 0$), this is a sufficient condition for an input motif μ_j being motif non-causal on an output μ_i. Therefore, we can find the motifs that are motif-causal for motif μ_i using a group lasso penalty optimization across the columns of W^o:

$$\min_W \sum_{t=2}^{T} (\mu_{i_t} - g_i(\mu_{0_{<t}}, ..., \mu_{m_{<t}}))^2 + \sum_{j=1}^{m} ||W^o_{:j}||_2$$

We define this as the *local motif loss*, $\mathcal{L}_{ml}$, which is optimized in each motif network using proximal gradient descent.

3.2.2 Training the Motif Causality Block

We next describe how the motif causality block is trained to learn motif causal relationships amongst traces, displayed in Figure 3.3. The block is structured in this way to accommodate the privacy integration (Section 3.4.2); here, we present its implementation without any privacy noise.

Partition data. First, the data is partitioned into r partitions (Step 1, Figure 3.3) such that no models are trained on overlapping data. The number of partitions, r, is a user-specified hyperparameter.

Build motif network for each motif. Next, within each data partition a set of motif networks is trained. As a pre-processing step, we assume each trace has been

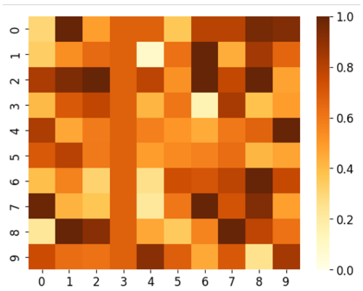


FIGURE 3.4: Example motif causality matrix for a small motif set ($m = 10$). Each value in the grid is between 0 and 1. 0 indicates no motif-causal relationship, and 1 indicates the strongest motif causal relationship.

chunked into a sequence of motifs of size τ (Section 3.1.1). τ is a hyperparameter, which we suggest chosen based on the longest effect time of a trace event. We use $\tau = 48$, corresponding to 4 hours of time, because large glucose events (from behaviors like eating) are encompassed within that time frame. We assume a tolerance of $\sigma = 2$ mg/dL, chosen to allow for reasonable variations in glucose. To model motif causality for an entire set of data, a g_i function is implemented for each motif via a separate RNN motif net following the description provided previously, resulting in m total networks (Step 2a, Figure 3.3). If all the motifs were trained together using a single motif network, it would not be possible to quantify the exact causal effects between each individual motif as we would not know which exact motifs contributed to a prediction (only that there is some combination of unknown motifs that contribute to an accurate prediction for a particular motif). By training each motif network separately, we are able to quantify the exact effect each motif has on each other, without any confounding effects from other motifs.

Combine outputs of individual motif networks. Each motif network outputs a vector of weights W^o of dimensionality $1 \times m$, corresponding to the learned causal relationships (Step 2b, Figure 3.3). Values in the vector are between 0 (no causal relationship) and 1 (strongest causal relationship) and give the degree to which every other motif is motif causal of the particular motif μ_i the RNN was specialized for. To return a complete matrix that summarizes causal relationships amongst *all* motifs, we stack the weights (Step 2c). The output of each data partition is a complete motif causality matrix, resulting in r total matrices, each of dimensionality $m \times m$. An example matrix is in Fig. 3.4.

Aggregate matrices and integrate with GAN. After motif causality matrices have been outputted from each data partition, the weights in the matrices are aggregated (Step 3, Figure 3.3) to return the final aggregate causality matrix, M (Step 4). In the nonprivate version, the weights are averaged. Finally, M is sent to the generator to help it learn how to conserve motif relationships within sequences of motifs in the synthetically generated data. Details are described next in the subsequent section.

3.3 GlucoSynth

The complete GlucoSynth framework, shown in Figure 3.5, comprises four key blocks: the motif causality block (explained previously in Section 3.2), an autoencoder, a generator and a discriminator. We walk through the remaining components of the framework surrounding the GAN next.

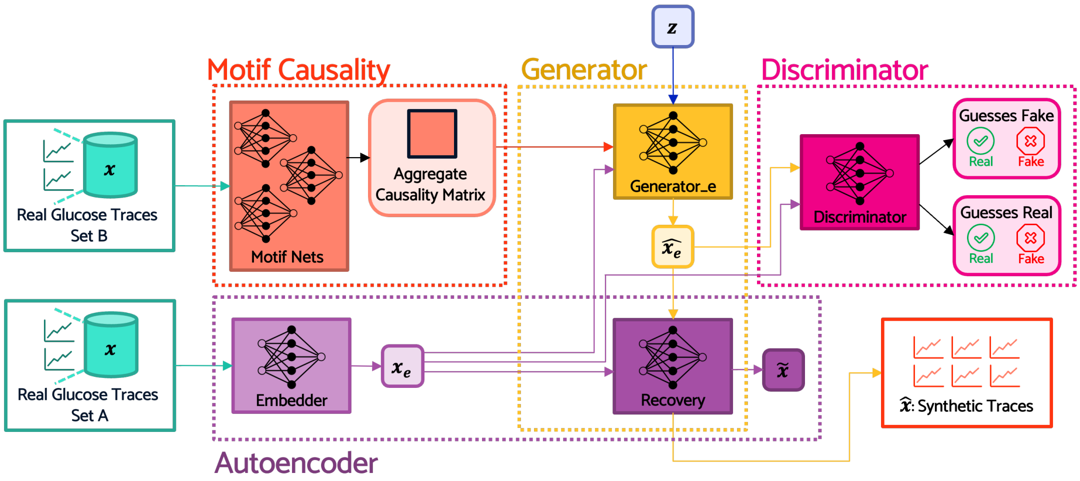


Figure 3.5: Overview of GlucoSynth Architecture.

3.3.1 GAN Architecture Components

Autencoder. We use an autoencoder (AE) with an RNN architecture to learn a lower dimensional representation of the traces, allowing the generator to better preserve underlying temporal dynamics of the traces. The autoencoder consists of two networks: an *embedder* and a *recovery network*. The embedder uses an encoding function to map the real data into a lower dimensional space: $Enc(x) : x \in \mathbb{R}^n \rightarrow x_e \in \mathbb{R}^e$ while the recovery network reverses this process, mapping the embedded data back to the original dimensional space: $Dec(x_e) : x_e \in \mathbb{R}^e \rightarrow \tilde{x} \in \mathbb{R}^n$. A foolproof autoencoder perfectly reconstructs the original input data, such that $x = \tilde{x} \equiv Dec(Enc(x))$. This process yields the Reconstruction Loss, $\mathcal{L}_R$, the Mean Square Error (MSE) between the original data x and the recovered data, $\tilde{x}$: $\mathsf{MSE}(x, \tilde{x})$.

Generator. We implement the generator via an RNN or LSTM. Importantly, the generator works in the embedded space, by receiving the input traces passed through the embedder (x_e). To generate synthetic data, a random vector of noise, z is passed through the generator and then the recovery network to return the synthetic traces in the original dimensional space. To learn how to produce high-quality synthetic data, the generator receives three key pieces of information:

1 – Stepwise. The generator receives batches of real data to guide the generation of realistic next step vectors. To do this, a Stepwise Loss, $\mathcal{L}_S$, is computed at time t using the MSE between the batch of embedded real data, x_{et}, and the batch of embedded synthetic data, $\hat{x}_{et}$: $\mathsf{MSE}(x_{et}, \hat{x}_{et})$. This allows the generator to compare (and learn to correct) the discrepancies in stepwise data distributions.

2 – Motif Causality. The generator needs to preserve sequences of motifs in addition to temporal dynamics. Using the aggregate causality matrix M returned from the Motif Causality Block, the generator computes a motif causality matrix, $M_{\hat{x}}$, on the set of synthetic data $\hat{x}$. Because the original causality matrix was not trained on data in the embedded space, we first run the set of embedded synthetic data through the recovery network $\hat{x}_e \rightarrow \hat{x}$. From there, the Motif Causality Loss, $\mathcal{L}_M$, is computed as the MSE error between the two matrices: $\mathsf{MSE}(M, M_{\hat{x}})$. These matrices give a causal value of seeing a motif μ_i in the future after some motif μ_j— unrealistic motif sequences will have causal values close to 0. As the generator learns to generate synthetic data that yields a realistic causal matrix (thereby identifying appropriate causal relationships from the motifs), it implicitly learns to not generate unrealistic motif sequences.

3 – Distributional. To guide the generator to produce a diverse set of traces, the generator computes a Distributional Loss, $\mathcal{L}_D$, the moments loss (MML), between the overall distribution of the real data x_e and the distribution of the synthetic data $\hat{x}_e$: $\mathsf{MML}(x_e, \hat{x}_e)$. The MML is the difference in the mean and variance of two matrices.

Discriminator. The discriminator is a traditional discriminator model using an RNN, the only change being it also works in the embedded space. The discriminator yields the Adversarial Loss Real, $\mathcal{L}_{Ar}$, the Binary Cross Entropy (BCE) between the discriminator guesses on the real data y_{x_e} and the ground truth y, a vector of 0's, $\mathsf{BCE}(y_{x_e}, y)$ and the Adversarial Loss Fake, $\mathcal{L}_{Af}$, the BCE between the discriminator guesses on the fake data $y_{\hat{x}_e}$ and the ground truth y, a vector of 1's, $\mathsf{BCE}(y_{\hat{x}_e}, y)$.

3.3.2 Training Procedure

First, the motif causality block is trained following the procedure described in Section 3.2.2, and then the rest of the GAN is trained. The autoencoder is optimized to minimize $\mathcal{L}_R + \alpha\mathcal{L}_S$, where α is a hyperparameter that balances the two loss functions. If the AE only receives $\mathcal{L}_R$ (as is typically done), it becomes overspecialized, i.e., it becomes too good at learning the best lower dimensional representation of the data such that the embedded data are no longer helpful to the generator. For this reason, the AE also receives $\mathcal{L}_S$, enabling the dual training of the generator and embedder. The generator is optimized using $\min(1 - \mathcal{L}_{Af}) + \eta(\mathcal{L}_S + \mathcal{L}_D) + \mathcal{L}_M$, where η is a hyperparameter that balances the effect of the stepwise and distributional loss. Finally the discriminator is optimized using the traditional adversarial feedback $\min \mathcal{L}_{Af} + \mathcal{L}_{Ar}$. The networks are trained in sequence (within each epoch) in the following order: autoencoder, generator, then discriminator. In our experiments we set $\alpha = 0.1$ and $\eta = 10$ as they enable GlucoSynth to converge fastest, i.e., in the fewest epochs.

3.4 Providing Differential Privacy

There are two components to our privacy architecture, described in the following two subsections: (1) each network in the GAN (Embedder, Recovery, Generator and Discriminator networks) is trained in a differentially private manner using the Differentially-Private Stochastic Gradient Descent (DP-SGD) algorithm from Abadi et al. [71]; and (2) the motif causality block is trained using the PATE framework from Papernot et al. [72]. Importantly, two completely separate datasets are used for the training of the motif causality block (dataset B in Figure 3.5) and the GAN (dataset A in Figure 3.5). We structure the privacy integration in this way to allow for better privacy-utility trade-offs. Our design satisfies the formal differential privacy notion introduced by Dwork et al. [73]. Differential Privacy (DP) provides an intuitive bound on the amount of information that can be learned about any individual in a dataset. A randomized algorithm $\mathcal{M}$ satisfies (ϵ, δ)-differential privacy if, for all datasets D_1 and D_2 differing by at most a single unit, and all $S \subseteq \text{Range}(\mathcal{M})$, $Pr[\mathcal{M}(D_1) \in S] \leq e^{\epsilon} Pr[\mathcal{M}(D_2) \in S] + \delta$. The parameters ϵ and δ determine the *privacy loss budget*, which provide a way to tradeoff privacy and utility; smaller values have stronger privacy. Importantly, privacy is provisioned at the *trace* level, and we assume each individual has only one trace in the dataset.

3.4.1 Training the GAN Networks with DP

To add privacy to the GAN components, each of the networks (Embedder, Recovery, Generator and Discriminator) is trained in a differentially private manner using DP-SGD [71]. Although the overall GAN framework is complicated, the individual networks all use simple RNN or LSTM architectures with Adam optimizers. As such, adding DP noise to their network weights is straightforward. We employ the following procedure using Tensorflow Privacy functions [74]. Since there are four networks being trained with DP, we divide the privacy loss budget evenly to get the budget per network, $\epsilon_{net} = \epsilon/4$. Then, we use Tensorflow's built-in DP accountant to determine how much noise must be added to the weights of each network based on the number of epochs, batch size, number of traces and ϵ_{net}. This function returns a noise multiplier, which we use when we instantiate a Tensorflow DP Keras Adam Optimizer for each network. Finally, we train each of the networks using their respective DP Keras Adam Optimizer, which automatically trains the network using DP-SGD.

3.4.2 Training the Motif Causality Block with DP

We train the motif causality block using the PATE framework [72]. PATE provides a way to return aggregated votes about the class a data point belongs to. First, the data is partitioned into r partitions, where r is determined based on the size of the dataset and the privacy loss budget. Then, a class membership model is trained independently for each partition. The class membership votes from each partition are aggregated by adding noise to the vote matrix and the noisiest votes are returned using the max-of-Laplacian mechanism (LNMax), tuned based on the privacy budget and r.

We use PATE to train the motif causality block: instead of predicting the degree of class membership we predict *causal* membership, e.g., does motif μ_i have a causal relationship to μ_j. The motif causality block is trained in the same procedure described in Section 3.2.2 with two changes: (1) the number of data partitions, r, is determined based on the privacy budget, instead of a user-specified value; (2) the final causality matrix M is aggregated using DP across the partitions. In normal PATE, carefully calibrated noise is added to a matrix of votes for each class, such that the classes with the noisiest votes are outputted. In our use, each value in a motif causality matrix may be likened to a class (i.e., causal “class" prediction between motif μ_i and μ_j). Thus, we use the LNMax mechanism (from predefined Tensorflow Privacy functions [74]) to aggregate the matrices weights and return M.

We use PATE instead of training each motif network using DP-SGD for better privacy-utility trade-offs. With DP-SGD, we would need to add noise to *every* motif net, eating up our privacy budget quickly and severely impacting the quality of the returned casuality matrices. PATE allows us to train each of the motif networks without any noise on the gradients, but then aggregates their returned causality matrices in a privacy-preserving manner, resulting in a better privacy-utility trade-off.

3.5 Evaluation

Evaluating synthetic data is notoriously difficult [75], so we provide an extensive evaluation across three criteria. Synthetic data should: 1) conserve characteristics of the real data (*fidelity*, Section 3.5.1); 2) contain diverse patterns from the real data without the introduction of anomalous patterns (*breadth*, Section 3.5.2); and 3) be usable in place of the original for real-world use cases (*utility*, Section 3.5.3).

Data. We use 100,000 single-day glucose traces randomly sampled across each month from January to December 2022, for a total of 1.2 million traces, collected from Dexcom's G6 Continuous Glucose Monitors (CGMs) [4]. Data was recorded every 5 minutes ($T = 288$) and each trace was aligned temporally from 00:00 to 23:59. Specifically, 100,000 single day patient traces were randomly sampled across each month (from January to December 2022). More details about this specific CGM technology are available in [4]. As explained in the approach (Section 3.3), our model uses two *separate* datasets for the training of the motif causality block and the rest of the GAN. As such, we used two different samples of glucose traces with no overlap between patients for the training of each section (meaning we actually used a total of 2.4 million traces across the entire model). We also note that we have received the proper ethical and legal consent from the individuals to use their data in this way (and for this purpose).

Hyperparameters. Our experiments were completed in the Google Cloud platform on an Intel Skylake 96-core cpu with 360 GB of memory. We use a separate validation dataset (not the set of original training traces) for all experimental results. Throughout all our experiments we use GlucoSynth model parameters of $\alpha = 0.1$ and $\eta = 10$ and a motif tolerance of $\sigma = 2$ mg/dL and motif length $\tau = 48$. Motif length of 48 timesteps is equivalent to 4 hours of time and represents a clinically significant threshold. This threshold was chosen because the effect of any behaviors on glucose occur within 4 hours of the event (e.g., the effect from eating a meal – a rise in glucose – will occur within 4 hours after eating.) We note that other choices for τ could be used, based on what types of phenomenon the users wish to replicate; for example, to capture day/night glucose rhythm effects, we suggest a τ of 144, corresponding to 12 hours of time.

We vary ϵ in our privacy experiments, but keep δ the same at $5e{-}4$. Importantly, in order to meet our privacy guarantees, we assume that privacy is provisioned at the trace level and each individual has only one trace in the dataset. The motif set is derived separately from the training data (either from a public dataset or generated based on knowledge about the underlying data, e.g., the possible glucose motif combinations), so as not to effect the differential privacy guarantees or use up any privacy budget. In our case, we assume the motif set is all-encompassing and generated from the universe of possible motifs, resulting in $m = 5,977,610$ total motifs in the motif set.

Benchmark Details. We restrict our comparison to the five most closely related state-of-the-art models for generating synthetic univariate time series with no labels or auxiliary data: Three nonprivate—TimeGAN [18], Fourier Flows (FF) [76], non-volume preserving transformations (NVP) [77]; and two private—RGAN [78] and dpGAN [79]. TimeGAN [18] is implemented from `www.github.com/jsyoon0823/TimeGAN`; Fourier Flows (FF) [76] are implemented from `www.github.com/ahmedmalaa/Fourier-flows`; RGAN [78] is implemented from `www.github.com/ratschlab/RGAN`; and DPGAN [79] is adapted from `www.github.com/SAP-samples/security-research-differentially-private-generative-models`. All the benchmarks were trained according to their suggested parameters, with most models trained for 10,000 epochs. We note that we trained for more than the suggested epochs (50,000 instead of 10,000) and tried many additional hyperparameter settings for RGAN to attempt to improve its performance and provide the fairest comparison possible.

3.5.1 Fidelity

Visualization. We provide visualizations of sample real and synthetic glucose traces from all models. Although this is not a comprehensive way to evaluate trace quality, it does give a snapshot view about what synthetic traces may look like, e.g., how realistic the synthetic traces may look. We provide heatmap visualizations, where each heatmap contains 100 randomly sampled glucose traces. Each row is a single trace from timestep 0 to 288. The values (coloring) in each row indicate the glucose value (between 40 mg/dL and 400 mg/dL). Figure 3.6 shows the nonprivate models, and Figures 3.7, 3.8, 3.9 show the private models with different privacy budgets. Upon examining the heatmaps, we notice that GlucoSynth consistently generates realistic looking glucose traces, even at very small privacy budgets.

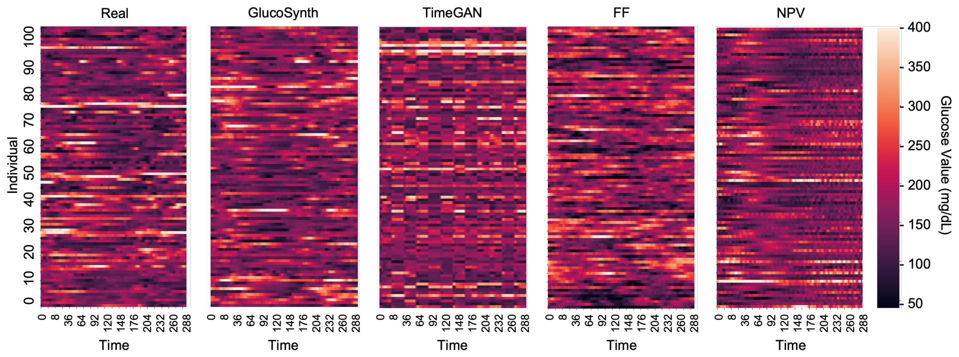


FIGURE 3.6: Heatmaps for Nonprivate Models

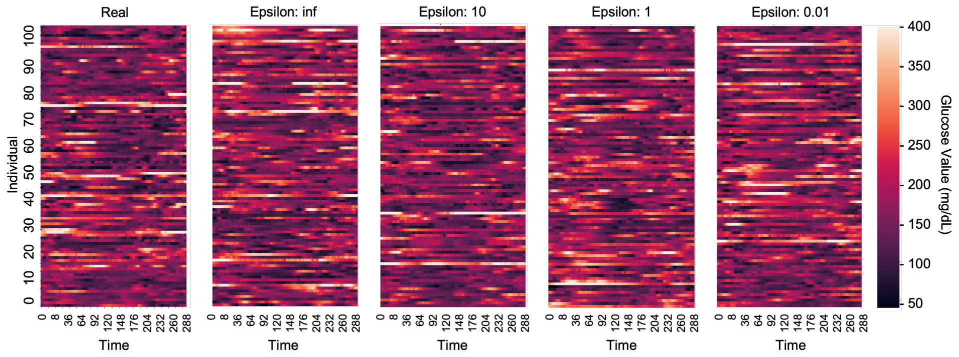


FIGURE 3.7: Heatmaps for GlucoSynth Across Different Privacy Budgets

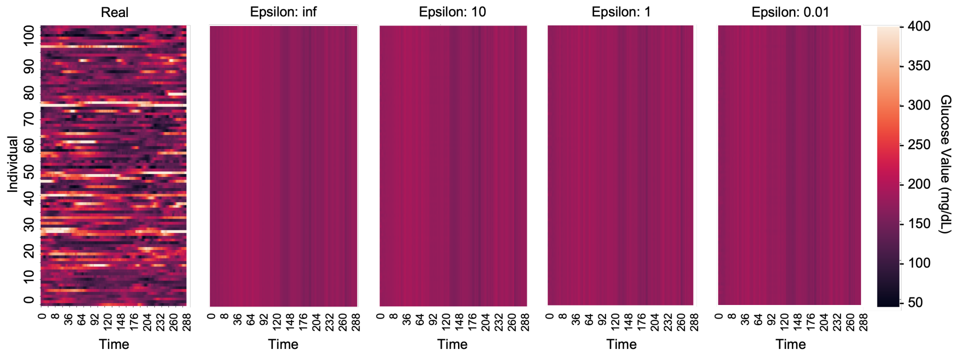


FIGURE 3.8: Heatmaps for RGAN Across Different Privacy Budgets

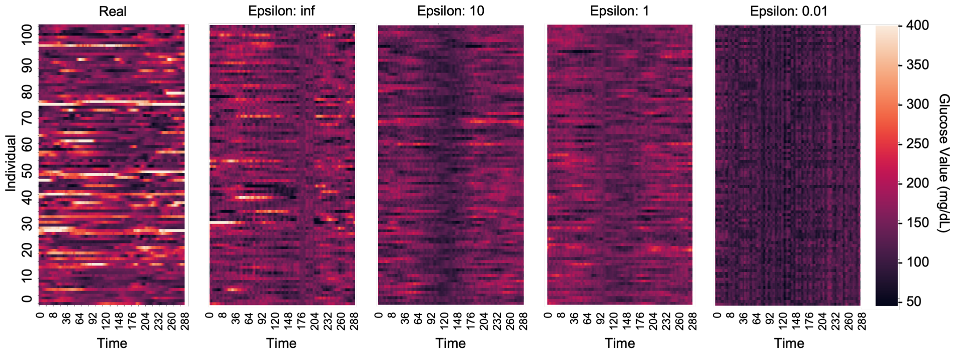


FIGURE 3.9: Heatmaps for dpGAN Across Different Privacy Budgets

TABLE 3.1: Glycemic Metric Explanations

Metric	Name	Explanation
VAR	Signal Variance	average trace variability
TIR	Time in Range	% of time glucose ≥ 70 & ≤ 180
Hypo	Time Hypoglycemic	% of time glucose < 70
Hyper	Time Hyperglycemic	% of time glucose > 180
GVI	Glycaemic Variability Index	more detailed measure of glucose variability
PGS	Patient Glycaemic Status	metric combining GVI and TIR

TABLE 3.2: Population Data Statistics. Each cell value for the synthetic data shows the (metric, p-value) using a 0.05 testing threshold. Bolded values do not have a statistically significant difference from the real data (what we want).

Model	ϵ	VAR	TIR	Hypo	Hyper	GVI	PGS
Real Data	N/A	2832.76	60.31	1.58	38.11	4.03	349.23
GlucoSynth	0.01	2575.501, 0.0	61.759, $2.0e{-5}$	1.331, 0.0	36.91, $5.66e{-4}$	**4.002, 0.085**	323.056, 0.0
	0.1	**2803.513, 0.356**	**60.088, 0.532**	1.264, 0.0	**38.648, 0.137**	3.969, $2.74e{-4}$	**347.562, 0.712**
	1	2760.853, 0.022	**60.597, 0.41**	**1.512, 0.163**	**37.892, 0.537**	**4.019, 0.577**	**345.159, 0.368**
	10	**2800.805, 0.316**	**60.24, 0.845**	**1.538, 0.395**	**38.222, 0.76**	3.963, $6.7e{-5}$	**344.376, 0.28**
	100	**2796.424, 0.244**	**60.138, 0.625**	**1.567, 0.808**	**38.295, 0.609**	**4.044, 0.32**	**352.679, 0.449**
	∞	**2811.622, 0.503**	**60.165, 0.682**	**1.54, 0.416**	**38.295, 0.61**	**4.056, 0.083**	**353.584, 0.339**
TimeGAN	∞	2234.576, $8.08e{-3}$	**62.315, 0.42**	0.657, $8.233e{-3}$	**37.028, 0.669**	5.482, 0.0	503.148, $0.2e{-5}$
FF	∞	**2836.067, 0.902**	46.578, 0.0	5.627, 0.0	47.795, 0.0	4.931, 0.0	528.773, 0.0
NVP	∞	1789.430, 0.0	65.499, 0.0	**1.507, 0.154**	32.994, 0.0	6.607, 0.0	589.473, 0.0
RGAN	0.01	56.96, 0.0	78.756, 0.0	0.0, $1.78e{-4}$	21.244, 0.0	2.52, 0.0	93.409, 0.0
	0.1	52.553, 0.0	71.617, $3.7e{-5}$	0.0, $1.78e{-4}$	25.715, 0.0	2.208, 0.0	98.944, 0.0
	1	67.346, 0.0	78.154, 0.0	0.0, $1.78e{-4}$	21.846, 0.0	2.251, 0.0	85.417, 0.0
	10	76.632, 0.0	83.681, 0.0	0.0, $1.78e{-4}$	16.319, 0.0	2.23, 0.0	64.562, 0.0
	100	84.918, 0.0	74.285, 0.0	0.0, $1.78e{-4}$	25.715, $0.6e{-5}$	2.208, 0.0	98.944, 0.0
	∞	89.702, 0.0	78.044, 0.0	0.0, $1.78e{-4}$	21.956, 0.0	2.184, 0.0	82.923, 0.0
dpGAN	0.01	451.098, 0.0	95.275, 0.0	4.60, 0.0	0.124, 0.0	7.718, 0.0	41.549, 0.0
	0.1	1057.205, 0.0	86.43, 0.0	0.837, 0.0	12.732, 0.0	6.349, 0.0	148.412, 0.0
	1	874.663, 0.0	86.631, 0.0	1.135, 0.0	12.234, 0.0	4.794, 0.0	118.286, 0.0
	10	1029.971, 0.0	88.122, 0.0	2.002, 0.0	9.876, 0.0	4.759, 0.0	93.632, 0.0
	100	821.636, 0.0	89.354, 0.0	0.664, 0.0	9.982, 0.0	4.613, 0.0	82.561, 0.0
	∞	1120.553, 0.0	81.773, 0.0	1.359, $0.3e{-5}$	16.868, 0.0	6.248, 0.0	188.991, 0.0

Population Statistics. In order to evaluate fidelity on a population scale, we compute a common set of glucose metrics used to evaluate patient glycemic control on the real and synthetic data, including average trace variability (VAR), Time in Range (TIR), the percentage of time glucose is within the clinical guided range of 70-180 mg/dL; and time hypo- and hyper- glycemic (time below and above range, respectively) in Table 3.2. More details on each of the metrics are included in Table 3.1. We test if the difference in metrics between the synthetic and real data is statistically significant, using a p-value of 0.05. A p-value < 0.05 indicates the difference is statistically significant. We want synthetic data that has similar population statistics to the real data: p-values > 0.05 such that the differences in statistics between real and synthetic data are not significant. GlucoSynth outperforms all other models, with no statistically significant difference in all metrics for privacy budgets of $\epsilon \geq 100$ and only one metric with a statistically significant difference for budgets $\epsilon = 1 - 10$.

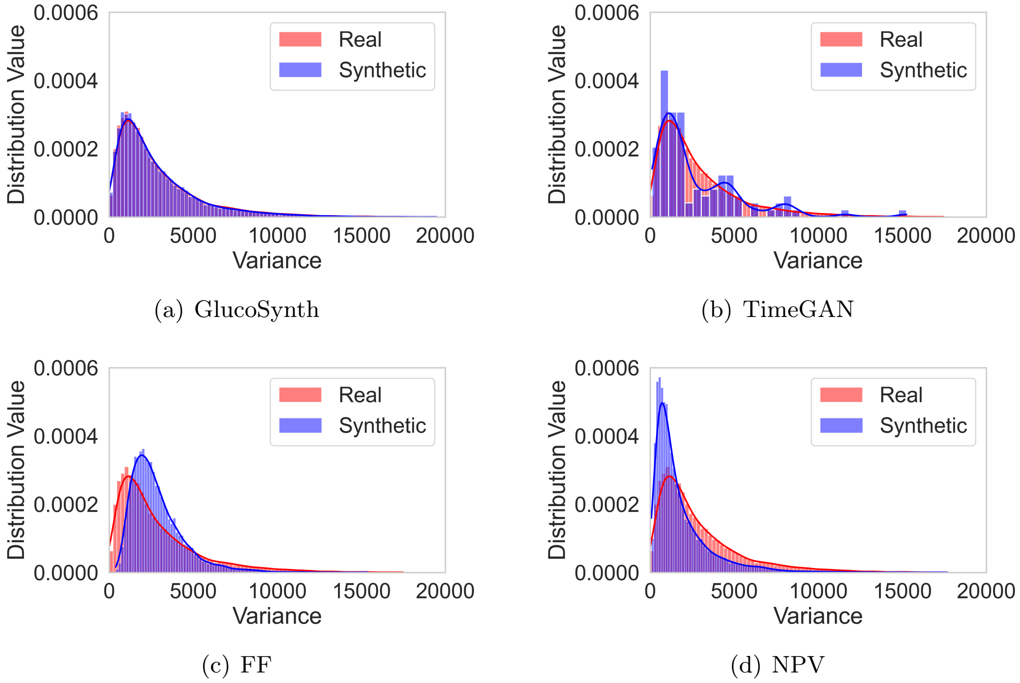
Distributional Comparisons. We visualize differences in distributions between the real and synthetic data by plotting the distribution of variances and using PCA [80]. Figure 3.10 and Figure 3.11 show the variance distribution and PCA plots, respectively for the nonprivate models. We also compare distributional changes across privacy budgets: Figures 3.12 and 3.13 show GlucoSynth, Figures 3.14 and 3.15 show RGAN and Figures 3.16 and 3.17 show dpGAN.

Looking at the figures, GlucoSynth better captures the distribution of the real data compared to all of the nonprivate models. As evidenced in the PCA plot, (Fig. 3.11), FF comes the closest to capturing the real distribution in its synthetic data, but ours does a better job of representing the more rare types of traces. GlucoSynth also outperforms all of the private models across all privacy budgets. Even at small budgets ($\epsilon < 1$), the general shape of the overall distribution is conserved (e.g., see Figure 3.12).

3.5.2 Breadth

Breadth refers to the representation of different types of traces within the synthetic data, and artifacts are patterns in the synthetic data not contained in the original data. This criteria hits on the fact that we would like a diverse representation of phenomenon (e.g., trace types) from the real data in the synthetic data, without the introduction of too many fake patterns. This is particularly notable in a clinical scenarios where the introduction of anomalous phenomenon in data used for training and diagnostic purposes can have real world safety consequences to patients. We quantify breadth in terms of glucose motifs. For each model's synthetic traces, we build a motif set (see Section 3.1.1). Given a real motif set from the validation traces S_x, for each synthetic motif set $S_{\hat{x}}$, we compute "Validation Motifs", (VM), the fraction of motifs found in the validation motif set that are present in the synthetic motif set, $\mathrm{VM}/|S_{\hat{x}}|$. This metric quantifies how good our synthetic motif set is (e.g., are its motifs mostly similar to motifs found in real traces). We also compute metrics related to *coverage*, the fraction of motifs in the validation motif set that are found in our synthetic data, defined as $\mathrm{VM}/|S_x|$. This gives a sense of the breadth in a more traditional manner. To compare actual *distributions* of motifs (not just counts), we compute the MSE between the distribution of real motifs S_x and the distribution of synthetic motifs $S_{\hat{x}}$. This gives a measure about how close the synthetic motif distribution is to the real one. We want high VM and coverage, and low MSE. Results are in Table 3.3.

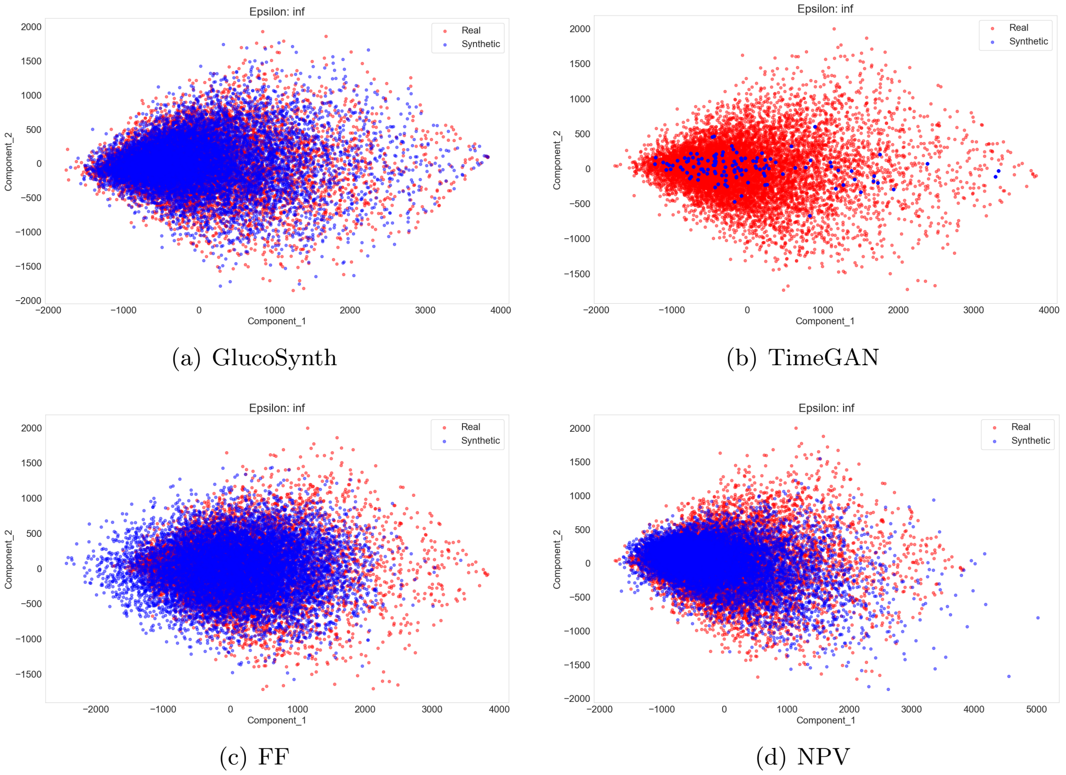
Compared to all other models across all privacy budgets, our model has the best ratio of found validation motifs, with close to 1.0 for VM and the lowest MSEs. It also has the best coverage for nonprivate settings and an ϵ of 100. Interestingly, dpGAN has the best coverage compared to all other models for privacy budgets $\epsilon \leq 10$ but worse MSEs across all budgets than GlucoSynth. This means that although it finds a broader *number* of motifs contained in the real data, the overall distributions of motifs it creates in the synthetic data have much higher error rates. We argue that the tradeoff found by our model is better because although it does miss some of the *types* of motifs from the real data (misses some breadth), from the ones it does find it constructs realistic distributions of the motifs and generates very few anomalous ones.



(a) GlucoSynth (b) TimeGAN

(c) FF (d) NPV

FIGURE 3.10: Distributional Variance for Nonprivate Models



(a) GlucoSynth (b) TimeGAN

(c) FF (d) NPV

FIGURE 3.11: PCA Comparison for Nonprivate Models

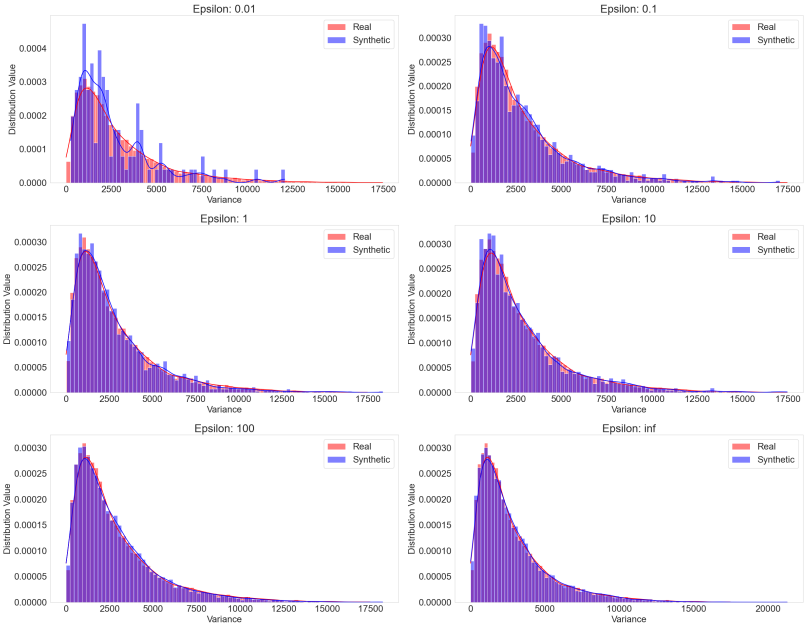


FIGURE 3.12: GlucoSynth Distributional Variance Comparison Across Privacy Budgets

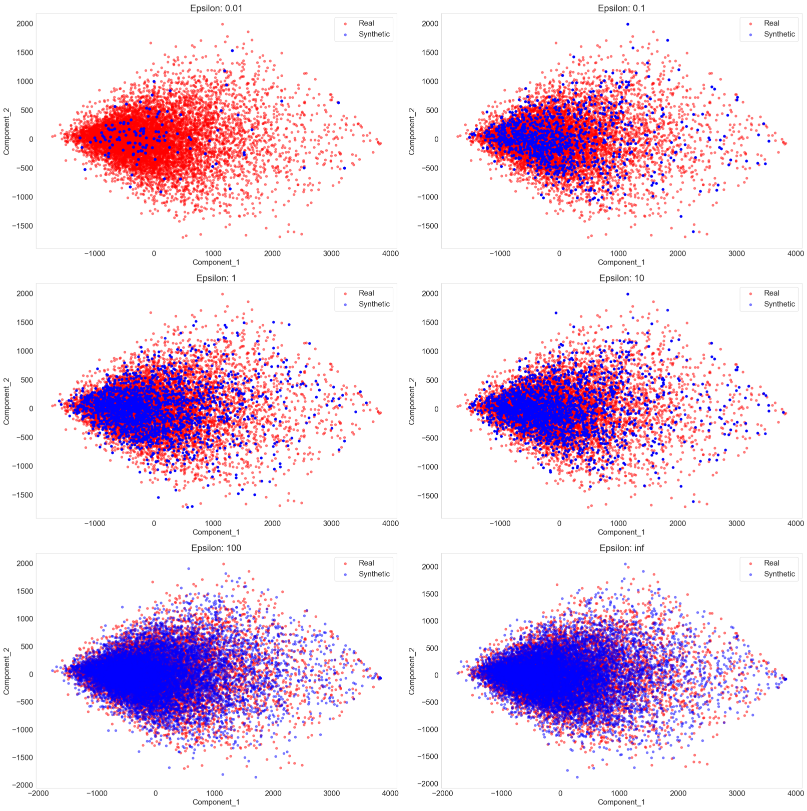


FIGURE 3.13: GlucoSynth PCA Comparison Across Privacy Budgets

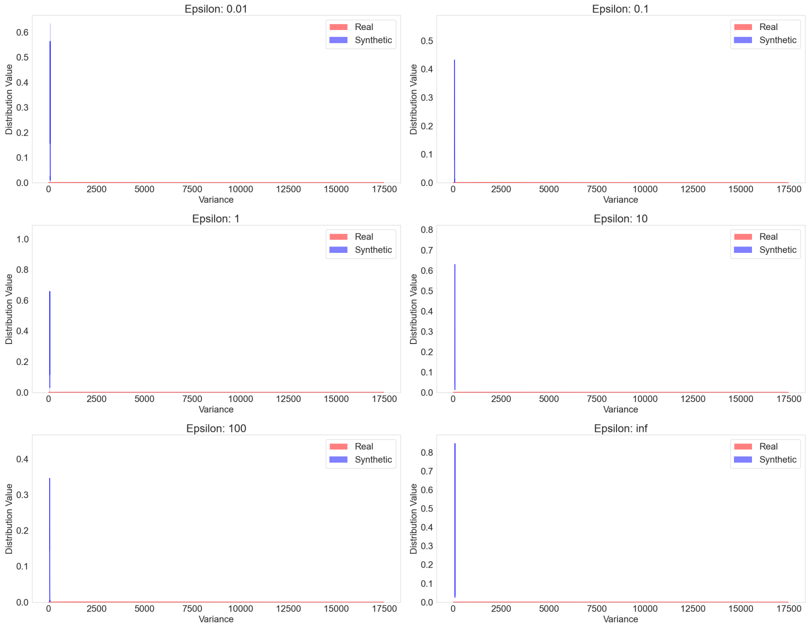


FIGURE 3.14: RGAN distributional Variance Comparison Across Privacy Budgets

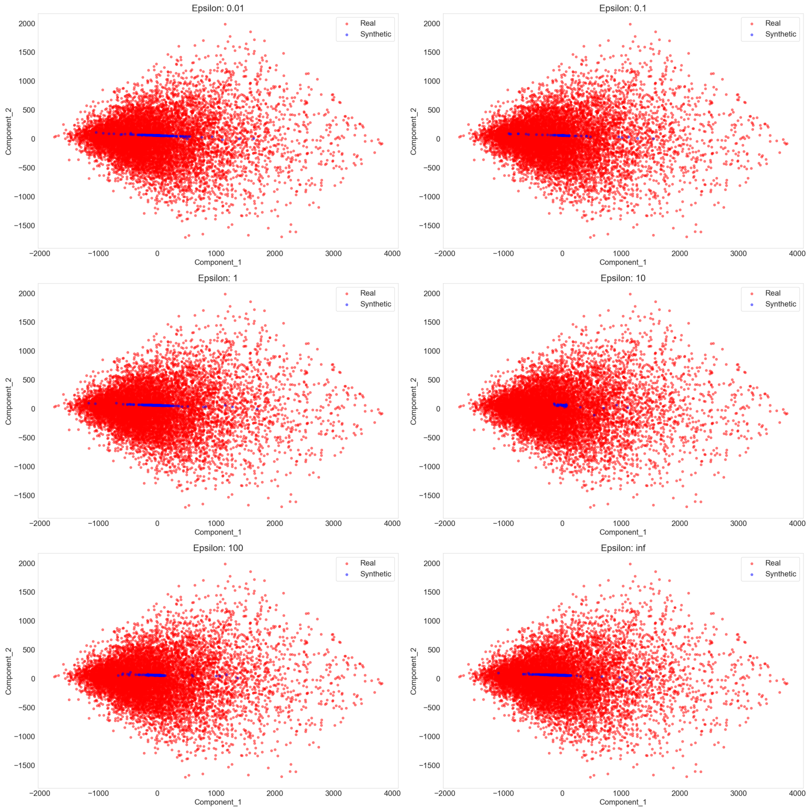


FIGURE 3.15: RGAN PCA Comparison Across Privacy Budgets

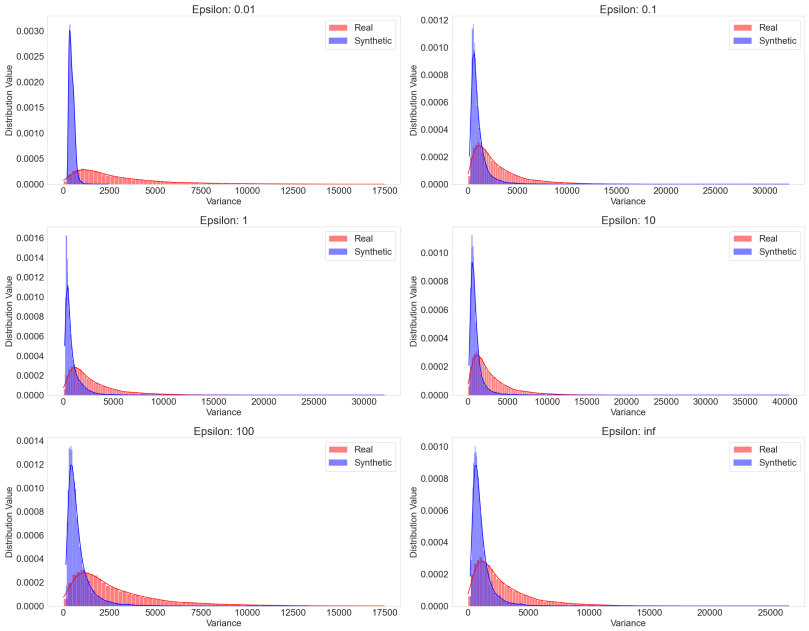


FIGURE 3.16: dpGAN distributional Variance Comparison Across Privacy Budgets

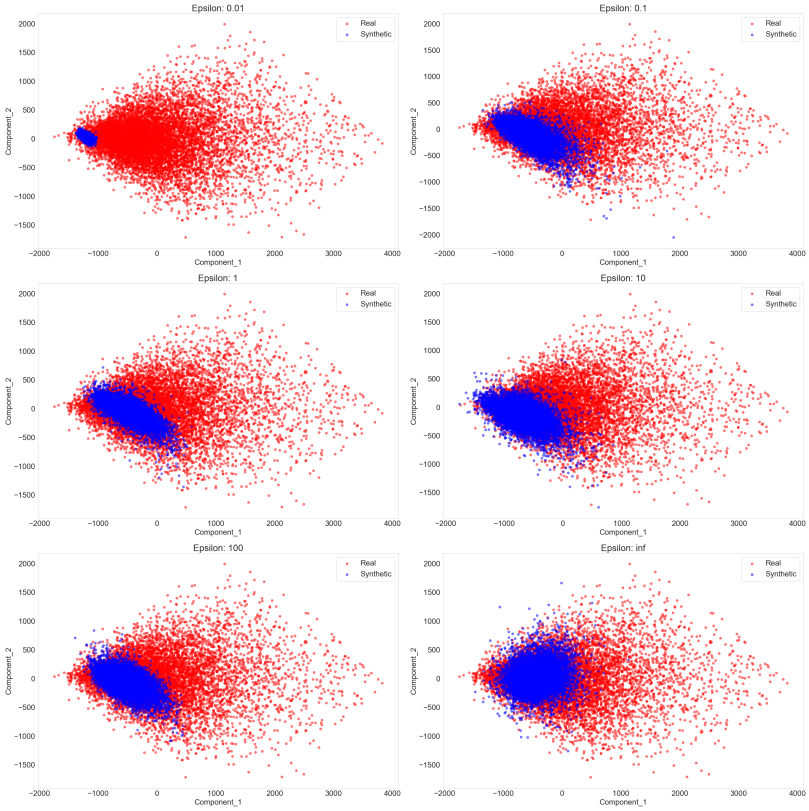


FIGURE 3.17: dpGAN PCA Comparison Across Privacy Budgets

Table 3.3: Breadth and Utility Evaluation. VM = fraction of motifs found in validation motif set; We want high VM, Coverage and low MSE, RMSE; Bolded values indicate the best ones at each privacy budget (nonprivate compared with private models when $\epsilon = \infty$).

		Breadth			**Utility**
Model	ϵ	VM	Coverage	MSE	RMSE
GlucoSynth	0.01	**1.000**	0.010	**99.0**	**0.038 ± 3e−4**
	0.1	**1.000**	0.083	**11.2**	**0.036 ± 3e−4**
	1	**0.992**	0.145	**6.7**	**0.030 ± 1e−4**
	10	**1.000**	0.167	**5.0**	**0.029 ± 1e−4**
	∞	**0.987**	**0.534**	**1.6**	**7e−3 ± 2e−4**
TimeGAN	∞	0.625	6e−3	107.7	0.061 ± 3e−4
FF	∞	0.642	0.405	2.0	0.038 ± 3e−4
NVP	∞	0.482	0.328	1.9	0.029 ± 3e−5
RGAN	0.01	0.013	1e−3	108.6	0.819 ± 0.010
	0.1	0.015	0.031	107.3	0.688 ± 6e−3
	1	0.015	0.033	103.3	0.651 ± 0.018
	10	0.017	0.053	100.3	0.619 ± 0.016
	∞	0.026	0.091	79.6	0.460 ± 0.013
dpGAN	0.01	0.094	**0.054**	180.1	0.205 ± 5e−3
	0.1	0.390	**0.195**	28.9	0.045 ± 2e−4
	1	0.480	**0.239**	23.2	0.030 ± 2e−5
	10	0.743	**0.251**	16.1	0.035 ± 8e−5
	∞	0.855	0.293	10.9	0.028 ± 5e−5

3.5.3 Utility

We evaluate our synthetic glucose traces for use in a glucose forecasting task using the common paradigm TSTR (Train on Synthetic, Test on Real), in which the synthetic data is used to train the model and then tested on the real validation data. This use case was chosen as it is a frequent real-world problem in both academic and industry scenarios (e.g., used in the current development of artificial insulin delivery). We train an LSTM network optimized for glucose forecasting tasks [81] and report the Root Mean Square Error (RMSE) in Table 3.3. We run the experiment 10 times and train the LSTM for 10,000 epochs. We have also tested with other models including RNNs, attention-based models and other LSTM architectures (such as bidirectional LSTMs) but show the results for the best performing model, the LSTM optimized for glucose forecasting. That being said, even when using the other models we still find that the RMSE decreases as our privacy budget increases, and that GlucoSynth performs the best compared to previous methods.

Clarke Error Grids. Since RMSE may provide a limited view about the predictions from the glucose forecasting model, we also plot the Clarke Error Grid [82], which visualizes the differences between a predictive measurement and a reference measurement, and is the basis used for evaluation of the safety of diabetes-related medical devices (for example, used for evaluating glucose outputs from predictive models integrated into artificial insulin delivery systems). The Clarke Error Grid is implemented using `www.github.com/suetAndTie/ClarkeErrorGrid`. The grids are shown in Figure 3.18.

In the figures, the x-axis is the reference value and the y-axis is the prediction. A diagonal line means the predicted value is exactly the same as the reference value

TABLE 3.4: Clarke Error Grid Zones. Value is the percentage of predicted datapoints. Categories go from A to E, best to worst. Bolded rows indicate the best results on the synthetic data at each privacy budget (nonprivate models compared with private models when $\epsilon = \infty$)

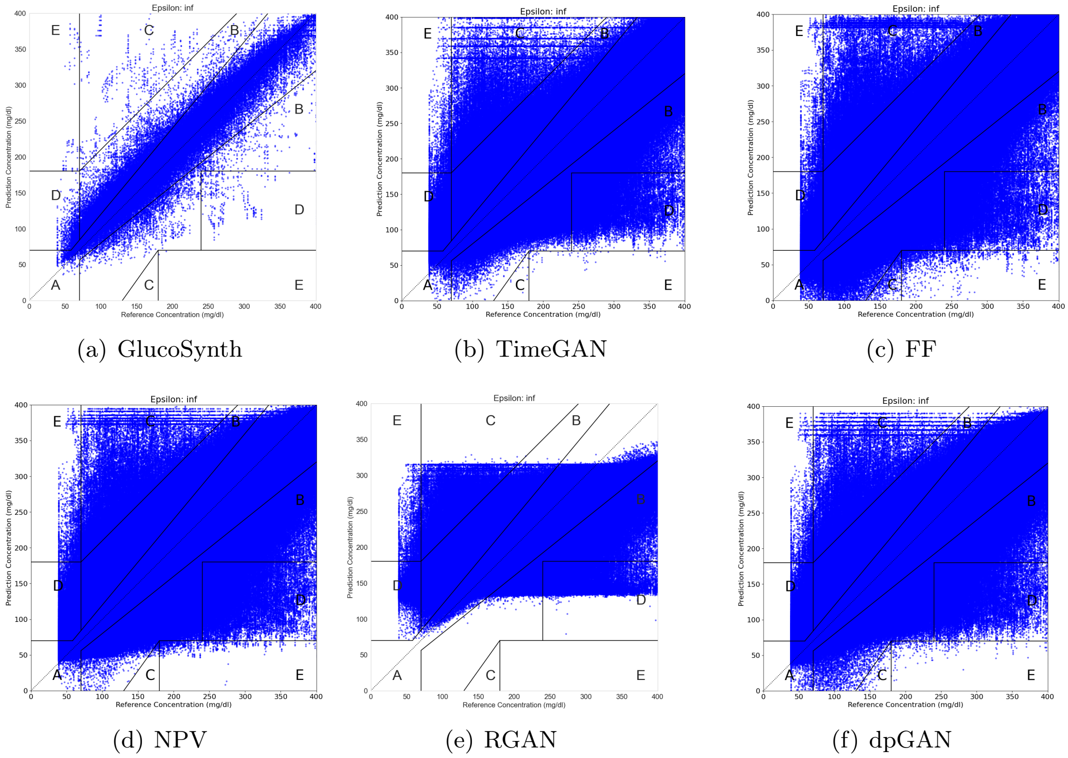
Model	ϵ	**A: Accurate**	**B: Acceptable**	**C: Overcorrection**	**D: Failure to Detect**	**E: Error**
GlucoSynth	0.01	$\mathbf{0.858 \pm 1.057e{-}3}$	$\mathbf{0.131 \pm 1.172e{-}3}$	$\mathbf{3.271e{-}3 \pm 0.0}$	$\mathbf{0.017 \pm 1.158e{-}4}$	$\mathbf{5.79e{-}6 \pm 1.2e{-}6}$
	0.1	$\mathbf{0.863 \pm 6.947e{-}3}$	$\mathbf{0.126 \pm 7.526e{-}4}$	$\mathbf{3.054e{-}3 \pm 1.45e{-}5}$	$\mathbf{0.018 \pm 4.34e{-}5}$	$\mathbf{5.79e{-}6 \pm 0.0}$
	1	$\mathbf{0.862 \pm 1.578e{-}3}$	$\mathbf{0.128 \pm 1.259e{-}3}$	$\mathbf{3.343e{-}3 \pm 1.45e{-}5}$	$\mathbf{0.016 \pm 3.329e{-}4}$	$\mathbf{5.79e{-}6 \pm 0.0}$
	10	$\mathbf{0.864 \pm 6.947e{-}3}$	$\mathbf{0.125 \pm 6.513e{-}4}$	$\mathbf{3.039e{-}3 \pm 5.79e{-}5}$	$\mathbf{0.017 \pm 4.34e{-}5}$	$\mathbf{8.68e{-}6 \pm 2.89e{-}5}$
	100	$\mathbf{0.864 \pm 1.74e{-}3}$	$\mathbf{0.126 \pm 1.447e{-}3}$	$\mathbf{3.387e{-}3 \pm 0.0}$	$\mathbf{0.017 \pm 2.895e{-}4}$	$\mathbf{5.79e{-}6 \pm 0.0}$
	∞	$\mathbf{0.964 \pm 1.201e{-}3}$	$\mathbf{0.035 \pm 1.158e{-}3}$	$\mathbf{3.039e{-}4 \pm 2.89e{-}5}$	$\mathbf{1.732e{-}4 \pm 1.158e{-}4}$	$\mathbf{8.68e{-}6 \pm 1.45e{-}5}$
TimeGAN	∞	0.741 ± 0.012	0.233 ± 0.012	$2.240e{-}3 \pm 9.8e{-}5$	$0.024 \pm 8.44e{-}4$	$2.19e{-}4 \pm 1.9e{-}5$
FF	∞	$0.824 \pm 6.624e{-}3$	$0.156 \pm 6.148e{-}3$	$3.547e{-}3 \pm 9.0e{-}5$	$0.017 \pm 3.940e{-}4$	$3.57e{-}4 \pm 8.0e{-}6$
NVP	∞	$0.79 \pm 3.03e{-}4$	$0.186 \pm 3.87e{-}4$	$3.49e{-}3 \pm 1.5e{-}5$	$0.02 \pm 1.04e{-}4$	$3.58e{-}4 \pm 5.0e{-}6$
RGAN	0.01	0.54 ± 0.014	0.435 ± 0.014	$3.389e{-}4 \pm 1.197e{-}4$	$0.024 \pm 2.71e{-}4$	$2.429e{-}4 \pm 3.43e{-}5$
	0.1	$0.594 \pm 1.998e{-}3$	$0.38 \pm 1.74e{-}3$	$1.326e{-}3 \pm 1.429e{-}4$	$0.025 \pm 1.069e{-}4$	$2.873e{-}4 \pm 8.68e{-}6$
	1	$0.637 \pm 6.785e{-}3$	$0.336 \pm 6.128e{-}3$	$2.661e{-}3 \pm 1.87e{-}5$	$0.024 \pm 6.464e{-}4$	$2.792e{-}4 \pm 2.95e{-}5$
	10	$0.634 \pm 3.452e{-}3$	$0.338 \pm 3.247e{-}3$	$2.253e{-}3 \pm 1.004e{-}4$	$0.025 \pm 2.894e{-}4$	$3.027e{-}4 \pm 1.71e{-}5$
	100	$0.638 \pm 4.709e{-}3$	$0.335 \pm 4.219e{-}3$	$1.991e{-}3 \pm 2.17e{-}5$	$0.025 \pm 4.884e{-}4$	$2.949e{-}4 \pm 2.26e{-}5$
	∞	$0.646 \pm 6.89e{-}4$	$0.326 \pm 7.19e{-}4$	$2.613e{-}3 \pm 2.852e{-}4$	$0.024 \pm 3.006e{-}4$	$2.859e{-}4 \pm 1.5e{-}5$
dpGAN	0.01	$0.308 \pm 3.482e{-}3$	$0.509 \pm 3.71e{-}3$	$2.894e{-}7 \pm 0.0$	$0.183 \pm 2.33e{-}4$	$1.114e{-}5 \pm 4.196e{-}6$
	0.1	$0.781 \pm 6.35e{-}4$	$0.191 \pm 5.37e{-}4$	$3.226e{-}3 \pm 5.715e{-}5$	$0.024 \pm 3.8e{-}5$	$2.533e{-}4 \pm 1.881e{-}6$
	1	$0.786 \pm 5.44e{-}4$	$0.187 \pm 5.81e{-}4$	$2.409e{-}3 \pm 2.894e{-}7$	$0.024 \pm 3.6e{-}5$	$2.078e{-}4 \pm 5.787e{-}7$
	10	$0.806 \pm 7.34e{-}4$	$0.169 \pm 6.09e{-}4$	$2.386e{-}3 \pm 1.476e{-}5$	$0.023 \pm 1.113e{-}4$	$2.146e{-}4 \pm 2.749e{-}6$
	100	$0.813 \pm 3.18e{-}4$	$0.161 \pm 2.86e{-}4$	$2.266e{-}3 \pm 2.083e{-}5$	$0.023 \pm 5.4e{-}5$	$1.889e{-}4 \pm 1.013e{-}6$
	∞	$0.819 \pm 1.487e{-}3$	$0.16 \pm 1.306e{-}3$	$3.193e{-}3 \pm 2.677e{-}5$	$0.018 \pm 1.60e{-}4$	$3.166e{-}4 \pm 5.208e{-}6$

(the best case). There are 5 total zones that make up the grid, listed in order from best to worst:

- Zone A – Clinically Accurate: Predictions differ from actual values by no more than 20% and lead to clinically correct treatment decisions.
- Zone B – Clinically Acceptable: Predictions differ from actual values by more than 20% but would not lead to any treatment decisions.
- Zone C – Overcorrections: Acceptable glucose levels would be corrected (overcorrection).
- Zone D – Failure to Detect: Predictions lie within the acceptable range but the actual values are outside the acceptable range, resulting in a failure to detect and treat errors in glucose.
- Zone E – Erroneous Treatment: Predictions are opposite the actual values, resulting in erroneous treatment, opposite of what is clinically recommended.

We show Clarke Error grids for all models (and the private models with no privacy included, $\epsilon = \infty$). This is because comparing the models at different privacy budgets is not very informative – it can be hard to tell exactly where changes between different budgets may occur. We also present a table with the percentages of predicted datapoints in each category in Table 3.4. This table includes a comparison among different privacy budgets for the private models (much more effective than the figures by themselves.)

Looking at the grids, we can see that GlucoSynth performs the best, with most of the values along the diagonal axis (Zone A and B) and less around the other zones (Zones C-E) as compared to the other models. This means that most of the predicted glucose values from the model trained on our synthetic data are in the Clinically Accurate and Acceptable ranges, with less in the erroneous zones. Moreover, by examining the table we see that GlucoSynth outperforms all other models across all privacy budgets as well.



(a) GlucoSynth

(b) TimeGAN

(c) FF

(d) NPV

(e) RGAN

(f) dpGAN

FIGURE 3.18: Clarke Error Zone Figures for All Models

Approach Limitations. In order to train on a huge set of glucose traces, we used a private dataset, not publicly available (one of the motivations for this project was actually to share a synthetic version of these traces). That being said, smaller samples of glucose traces with similar patient populations are available at OpenHumans [83] and T1D Exchange Registry [84]. In addition, one of the reasons our privacy results perform well is because we use two *separate* datasets for the training of the motif causality block and the GAN. However, this may be a limiting factor for others that do not have a large enough set of traces available to be able to train adequately on partitioned data.

3.6 Related Work

We overview related work in three lines of research: time series, conditional time series, and time series methods that employ differential privacy. Table 3.5 summarizes the characteristics of previous time series synthesis methods. We note that there have been exciting developments in new approaches for adjacent research tasks (data augmentation, forecasting) such as diffusion models [85], but there are not yet any publicly available models specifically for the generation of complete synthetic time series datasets. As such, we focus the scope of our comparison on the current state-of-the-art methods for synthetic time series which all build upon Generative Adversarial Networks (GANs) [64] and transformation-based approaches [77]. In particular TimeGAN [18], RGAN [78] and dpGAN [79] are most similar to ours and used as benchmarks in the evaluation in Section 3.5.

Time Series. There have been promising models to generate synthetic time series across a variety of domains such as financial data [86], cyber-physical systems (e.g., smart homes [87]), and medical signals [88]. Brophy et al. [89] provides a

Table 3.5: Summary of Previous Methods for Time Series Synthesis. *CI = conditional information or extra features

Name	Private?	No Labels Required?	No CI*?	Length
TimeGAN [18]	x	✓	✓	24 - 58
TTS-GAN [19]	x	x	✓	24 - 150
SigCWGAN [90]	x	✓	x	80,000
RGAN [78]	✓	✓	✓	16 - 30
RCGAN [78]	✓	✓	x	16 - 30
dpGAN [79]	✓	✓	✓	96
RDP-CGAN [20]	✓	✓	x	2 - 4097
DoppelGANger [93]	✓	✓	x	50 - 600
GlucoSynth (Ours)	✓	✓	✓	288

survey of GANs for time series synthesis. TimeGan [18] is a popular benchmark that jointly learns an embedding space using supervised and adversarial objectives in order to capture the temporal dynamics amongst traces. TTS-GAN [19], trains a GAN model that uses a transformer encoding architecture in order to best preserve temporal dynamics. Transformation-based approaches have also had success for time series data. Real-valued non-volume preserving transformations (NVP) [77] model the underlying distribution of the real data using generative probabilistic modeling and use this model to output a set of synthetic data. Similarly, Fourier Flows (FF) [76] transform input traces into the frequency domain and output a set of synthetic data from the learned spectral representation of the original data. Methods that only focus on learning the temporal or distributional dynamics in time series are not sufficient for generating *realistic* synthetic glucose traces due to the lack of temporal dependence within sequences of glucose motifs.

Conditional Time Series. Many works have developed time series models that supplement their training using extra features or conditional data. Esteban, Hyland, and Rätsch [78] develops two GAN models (RGAN/RCGAN) with RNN architectures, conditioned on auxiliary information provided at each timestep during training. SigCWGAN [90] uses a mathematical conditional metric ($Sig - W_1$) characterizing the signature of a path to capture temporal dependence of joint probability distributions in long time series data. However, our glucose traces do not have any additional information available so these methods cannot be used[2].

Differentially-Private GANs. To protect sensitive data, several GAN architectures (DP GANs) have been designed to incorporate privacy-preserving noise needed to satisfy differential privacy guarantees [91]. Although DP GANs such as Pate-GAN [92] have had great success for other data types and learning tasks (e.g., tabular data, supervised classification tasks), results have been less satisfactory in DP GANs developed for time series.

RGAN/RCGAN [78] also includes a DP implementation, but the authors find large gaps in performance between the nonprivate and private models. Frigerio et al. [79] extends a simple DP GAN architecture (denoted dpGAN) to to time-series data. The synthetic data from their private model conserves the distribution of the real data but loses some of the variability (diversity) from the original samples. RDP-CGAN [20] develops a convolutional GAN architecture that uses Rényi differential

[2] There is a caveat here that RGAN does not use auxillary information, hence why we compare with it in our benchmarks.

privacy specifically for medical data. Across different datasets, they find that reasonable privacy budgets result in major drops in the performance of the synthetic data. Finally, DoppelGANger [93] develops a temporal GAN framework for time series with metadata and perform an in-depth privacy evaluation. Notably, they find that providing strong theoretical DP guarantees results in destroying the fidelity of the synthetic data, beyond anything feasible for use in real-world scenarios. Each of these methods touches on the innate challenge of generating DP synthetic time series due to very high tradeoffs between utility and privacy. Our DP framework uses two different methods to integrate privacy into our GAN architecture, resulting in a better utility-privacy trade-off than previous methods.

3.7 Summary

In this chapter we have presented GlucoSynth, a novel GAN framework with integrated differential privacy to generate synthetic glucose traces. GlucoSynth conserves motif relationships within the traces, in addition to the typical temporal dynamics contained within time series. We presented a comprehensive evaluation using 1.2 million glucose traces wherein our model outperformed all previous models across three criteria of fidelity, breadth and utility. GlucoSynth facilitates the sharing of medical trajectories with reduced privacy and legal concerns, directly addressing the privacy challenges elucidated in Chapter 1. In the next chapter (Chapter 4) we introduce a differential privacy-based framework to learn aggregate population rule structures from local client rulesets produced by rule-based learning mechanisms in CDSS.

Chapter 4

Differentially-Private Rule Learning (DP-RuL)

With the availability of mobile sensors and devices, CDSS are being integrated into third-party health applications for a myriad of health contexts, including chronic disease management, remote patient monitoring, and medical triage [94].[1] Many CDSSs rely on logic-based learning systems [23], in which structured *rules* are used to make decisions due to their increased expressivity (diverse representations of medical associations), dual understandability by humans and machines (e.g., using a rule grammar), and increased explainability which promotes user trust in the system [24, 25]. Even with the proliferation of deep learning and generative ML, rule-based learners are still extremely common in clinical applications; indeed, some deep learning frameworks actually use a rule-based output layer or ensemble learner to better explain model predictions, increasing trust and interpretability of the overall system [60, 61]. In a typical distributed CDSS setting, mobile apps using data from wearables learn and characterize patient behaviors using a rule-based learner, such as Signal Temporal Logic (STL) Learning (described in Section 4.1.1). From there, the apps send the rules to a centralized server which aggregates patterns across patients to learn about clinical conditions that may generalize to broader populations or subcohorts.

Serious privacy concerns, such as data compromise and unsanctioned use of user data, arise with the use of patient data in CDSSs, especially those deployed in third-party health applications [26]. Since these third-party health applications are not covered by HIPAA, they are not subject to the same protective privacy requirements that govern data in health organizations [15]. Breaches of patient data from third party health apps, however, can have significant consequences, including job and insurance discrimination based on exposed sensitive health details (e.g., a patient's past drug, mental health or serious disease history) [95].

Project Goal. Given these concerns, the goal of this project is to learn a population ruleset representative of the local client rule structures, while preserving the privacy of individuals involved in the rule collection. We consider an untrusted server S that wishes to generate a population ruleset R_S from the local rulesets of n individual clients, $\{R_1, ..., R_n\}$. Participating clients are expected to behave honestly but want to protect the sensitive information contained in their rulesets from the server and other protocol participants. We wish to learn population rulesets with two key qualities: (1) *coverage* — the learned population ruleset captures well the breadth of behavior of the client population; and (2) *clinical utility* — the learned rules are useful in a medical context.

[1] This chapter is based on: Josephine Lamp, Lu Feng, and David Evans. "Differentially-Private Rule Learning for Clinical Decision Support Systems". Arxiv (2024).

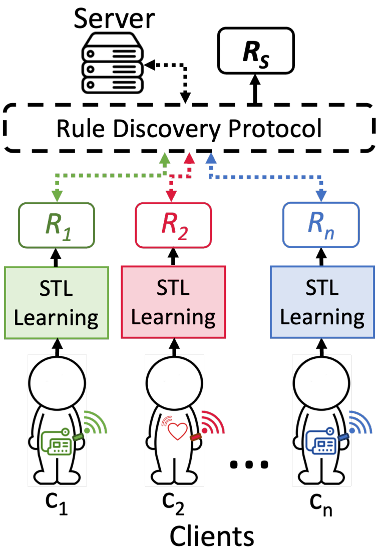


FIGURE 4.1: Our privacy-preserving CDSS framework. Clients locally collect data from sensors and wearables, which are used to learn personalized rule sets $(R_1, \dots, R_n)$ using STL Learning describing individual conditions. A *Rule Discovery Protocol* sends a series of structured queries to the clients who respond using randomized response, to produce an aggregate population ruleset R_S to discover generalizable clinical rules.

Learning Rules with Privacy. To provide local differential privacy (LDP), individual users each perturb their own data before it is collected and used for population-level aggregation [27]. Previous work has developed differentially-private methods for distributed learning in various settings including finding new frequent strings [28], discovering keystroke data [29, 30], text mining [31], frequent item mining [32, 33, 34] and data mining personal information [35]. However, as we discuss more in Section 4.4, no previous work has developed LDP methods for learning logic-based rule structures or for CDSS applications, and none of the methods developed for these other settings can be directly applied to provide an adequate solution to the privacy rule discovery problem.

Contributions. We present and evaluate the first locally differentially-private framework to learn population rulesets with high coverage and clinical utility for logic-based CDSSs, depicted in Figure 4.1. We develop a novel Rule Discovery Protocol (Section 4.2.1), which uses a method based on Monte-Carlo Tree search (MCTS) to search a rule grammar in a structured way and find population rules contained by the clients. The protocol follows the traditional MCTS steps (Selection, Expansion, Querying, and Backpropagation). To provide LDP, we adapt the querying phase to use randomized response. To find clinically useful rules, we adapt the MCTS scoring function, which guides the search process about which subtrees to continue searching down, to use privacy-preserving estimates of the number of clients who have rules that match a template rule structure in the grammar. By guiding the searching based on client responses, and taking advantage of the rule grammar, we are able to efficiently learn population rulesets including rules with complex structures.

Each query in the Rule Discovery Protocol is allocated a privacy loss budget that determines the randomized response noise used in the response. We develop an adaptive budget allocation method, which dynamically provisions the privacy loss budget (Section 4.2.6). The intuition is to find the minimum budget per query to gain

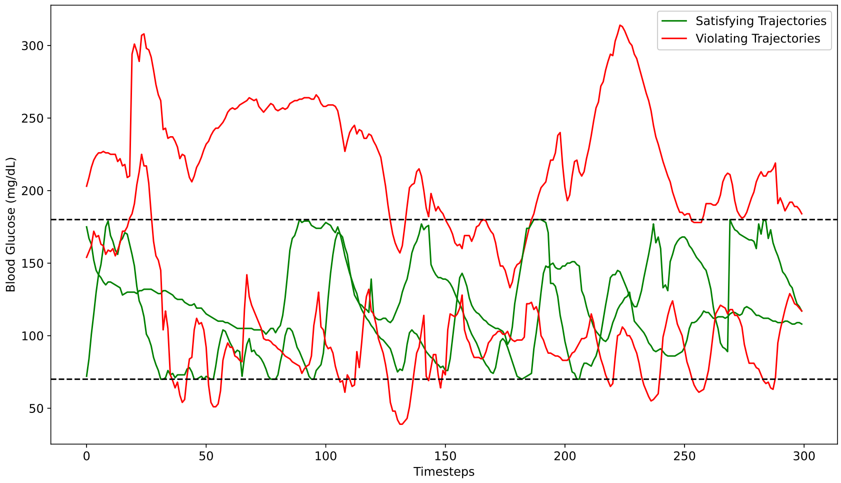


FIGURE 4.2: Visual of the STL-learned rule $\square_{[0,300]}(\mathsf{BG} \geq 70 \wedge \mathsf{BG} \leq 180)$ from glucose trajectories. The green trajectories satisfy the rule (glucose in range), and the red violate it.

enough information to determine whether a node should be further explored.

We evaluate our protocol on three clinical datasets from different medical domains: Intensive Care Unit (8000 patients, 2,418,776 total timepoints), Sepsis (40,336 patients, 1,552,210 total timepoints), and Diabetes (34,013 patients, 140,461 total timepoints), and find that we are able to learn population rulesets with high coverage and clinical utility, even at low privacy loss budgets (Section 4.3).

4.1 Preliminaries

In this section we provide an overview of the relevant background on Signal Temporal Logic and STL Learning (Section 4.1.1), Monte-Carlo Tree Search (Section 4.1.2), and Local Differential Privacy (Section 4.1.3).

4.1.1 Signal Temporal Logic

Signal Temporal Logic (STL) is a formal specification language used to express temporal properties over real-valued trajectories, commonly used to reason about behaviors of real-world systems, such as cyber-physical systems [96]. We denote Z and P as finite sets of real and propositional variables. We let $w : \mathbb{T} \longrightarrow \mathbb{R}^m \times \mathbb{B}^u$ be a multi-dimensional signal, where $\mathbb{T} = [0, d) \subseteq \mathbb{R}$, $m = |Z|$ and $u = |P|$. The syntax of an STL formula φ over $Z \cup P$ is defined by the grammar:

$$\varphi ::= p \mid z \sim l \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \square_I \varphi \mid \lozenge_I \varphi \mid \varphi_1 \mathbf{U}_I \varphi_2$$

where $p \in P$, $z \in Z$, $\sim \ \in \{<, \leq\}$, $l \in \mathbb{Q}$, $I \subseteq \mathbb{R}^+$ is an interval and $\square$, $\lozenge$, and $\mathbf{U}$ denote temporal operators "always", "eventually", and "until". STL can be interpreted over a signal to describe the satisfaction of a formula (an example is in Figure 4.2). Bartocci et al. [96] provide a comprehensive survey on STL and its use in cyber-physical systems.

STL Learning. Although there are many rule-based machine learning methods, we focus on STL Learning due to its ability to expressively represent temporal properties of real-valued signal trajectories and because it is used in real clinical use cases (e.g., [97]). STL learning takes advantage of the expressivity of the STL language

Table 4.1: Example STL Rules Learned from Our Datasets

Dataset	Rule
ICU	$\square_{[0,0]}(\text{HR} \geq 80.369 \wedge \text{Pulse} \geq 74.034)$
	$((\text{MET} \geq 0.007)\ \mathbf{U}_{[0,1]}\ (\text{DeathProb} = 0.032))$
	$\lozenge_{[0,1]}(\text{Blood\_Urea\_Nitrogen} \leq 12.889 \wedge \text{Creatinine} \geq 0.723)$
Sepsis	$((\text{Temp} \geq 9.059)\ \mathbf{U}_{[1,1]}\ (\text{BaseExcess} \geq 0.048))$
	$\square_{[0,1]}(((\text{HGB} \geq 0.385 \wedge \text{MAP} \leq 110.015) \vee \text{Bilirubin\_Direct} \leq 107.835) \wedge \text{AST} \geq 47.955)$
	$\lozenge_{[1,2]}(((\text{PaCO\_2} \geq 0.171 \vee \text{Chloride} \leq 8.029) \implies \text{Potassium} \geq 0.014) \wedge \text{SepsisProb} \geq 0.85)$
T1D	$((\text{HbA1c} \geq 7.571)\ \mathbf{U}_{[1,2]}\ (\text{Hypoglycemia} = 1.0))$
	$\square_{[0,1]}((\text{TotalDailyInsPerKg} \leq 0.305 \implies \text{PtA1cGoal} \geq 0.518\%) \wedge \text{GFR} \leq 89\%)$
	$\lozenge_{[1,1]}(((\text{BMI} \geq 27.066 \vee \text{HeightCm} \geq 180.022) \implies \text{HbA1c} \leq 6\%) \wedge \text{Bolus} \geq 57.424)$

The ICU rules characterize relationships between labs, physiological values and mortality (MET, DeathProb). The Sepsis rules characterize relationships between lab values and sepsis outcomes (SepsisProb). The T1D rules characterize relationships between insulin, blood glucose levels, glomular filtration rate (GFR), body mass index (BMI) and glycemic outcomes (HbA1c levels, goals and hypoglcemia).

and provides ML techniques to infer STL formulae and parameters from continuous trajectories [98]. There are many STL Learning algorithms, and our Rule Discovery Protocol does not depend on how the local client STL rules are learned. For our experiments, we use the Nenzi et al. [99] genetic algorithm-based methodology as it is well suited to supervised classification tasks and is able to learn *both* the parameters and the structure of STL formulae from real data. The algorithm requires positive and negative trajectories (e.g., regular and anomalous) and its goal is to learn rules that best characterize and separate the positive and negative samples. We use our own implementation of such an algorithm, developed in Python3 and available here: https://github.com/jozieLamp/STLlearning. Some examples of learned STL rules are shown in Table 4.1.

4.1.2 Monte-Carlo Tree Search (MCTS)

Monte-Carlo Tree Search (MCTS) [100, 101] is a well-known algorithmic search method used to solve sequential decision problems and search large combinatorial spaces. MCTS works by building a search tree that balances *exploration*, finding new options in the search space, and *exploitation*, focusing on the parts of the space that are most likely to return good rewards. There are four key phases of MCTS: Selection, Expansion, Querying (traditionally called Simulation), and Backpropagation.

A common MCTS algorithm is the Upper Confidence Bounds for Trees (UCT) algorithm [101]. This method asymmetrically searches the tree, focusing on the pathways that are most promising. The UCT scoring function, which we adapt for our method, is:

$$score = rw + C_p \times \sqrt{\frac{v_{parent}}{v_b}}$$

where rw is the current reward, v_{parent} is the visit count of the parent node, and v_b is the visit count of the current node. The hyperparameter C_p balances the exploration vs. exploitation trade-off in the MCTS search, and is typically set to $\frac{1}{\sqrt{2}}$.

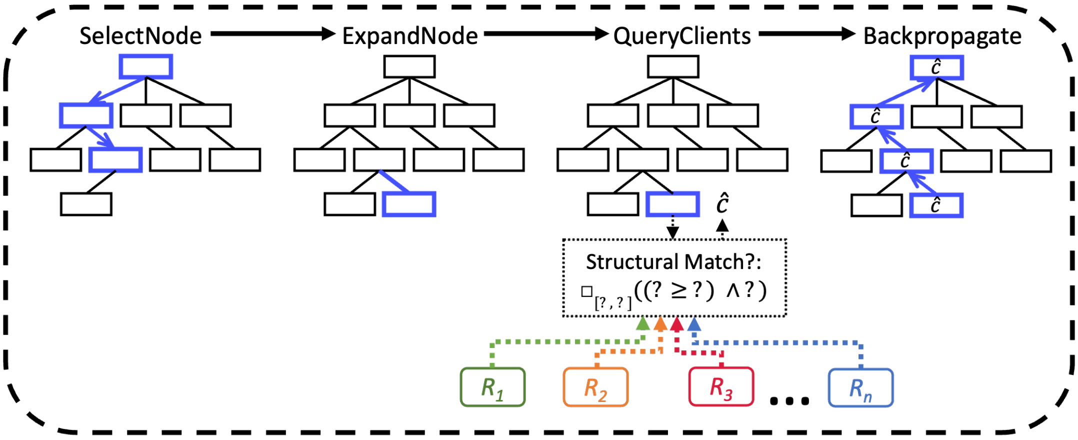


FIGURE 4.3: Rule Discovery Protocol. The protocol iterates through each MCTS phase (SelectNode, ExpandNode, QueryClients, Backpropagate) to send a series of structured queries to the clients, who respond using randomized response, to generate R_S.

4.1.3 Local Differential Privacy

Local Differential Privacy (LDP) is a paradigm well suited to the distributed framework deployed for many CDSS systems. It provides privacy assurances to clients without relying on any external server since individual users each perturb their own data before it is collected and aggregated [27]. In this setting, a centralized, untrusted server S, wishes to aggregate some summary statistic s from n individual clients' data records, $\{x_1, ..., x_n\}$, that contain private information. Each client locally perturbs the requested data x_i before sending it to the server. An algorithm $\mathcal{A}$ satisfies ϵ-local differential privacy where $\epsilon > 0$ if, for any possible pairs of inputs x and x':

$$\forall s \in Range(\mathcal{A}) : \frac{Pr[\mathcal{A}(x) = s]}{Pr[\mathcal{A}(x') = s]} \leq e^{\epsilon}$$

where $Range(\mathcal{A})$ denotes every possible output of $\mathcal{A}$.

4.2 Rule Discovery Protocol

We introduce a rule discovery protocol (Figure 4.3) which integrates MCTS with LDP to search a rule grammar and find population rules. We walk through each protocol phase in Section 4.2.1–Section 4.2.5. Then, Section 4.2.6 describes an adaptive privacy loss budget allocation method which determines the privacy loss budget to use for each query.

4.2.1 Overview

The rule discovery protocol uses an *exploration tree*, T, to search over an STL grammar G. The protocol follows the traditional MCTS steps (Selection, Expansion, Querying, and Backpropagation), adapted to support LDP by using randomized response when clients respond to queries. We use an MCTS-based approach for our protocol due to its ability to efficiently search a tree structure while balancing the trade-off between exploration (i.e., finding new nodes and pathways through the tree), and exploitation (i.e., focusing on the known nodes in the tree that maximize the score function). This

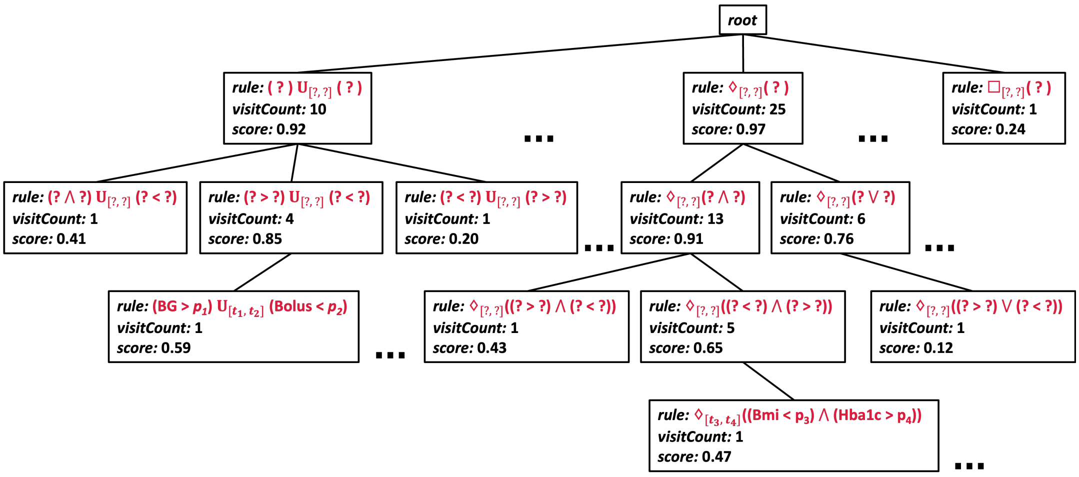

root
rule: (?) $U_{[?,?]}$ (?) visitCount: 10 score: 0.92
rule: $\Diamond_{[?,?]}$(?) visitCount: 25 score: 0.97
rule: $\Box_{[?,?]}$(?) visitCount: 1 score: 0.24
rule: (? ∧ ?) $U_{[?,?]}$ (? < ?) visitCount: 1 score: 0.41
rule: (? > ?) $U_{[?,?]}$ (? < ?) visitCount: 4 score: 0.85
rule: (? < ?) $U_{[?,?]}$ (? > ?) visitCount: 1 score: 0.20
rule: $\Diamond_{[?,?]}$(? ∧ ?) visitCount: 13 score: 0.91
rule: $\Diamond_{[?,?]}$(? ∨ ?) visitCount: 6 score: 0.76
rule: (BG > p_1) $U_{[t_1, t_2]}$ (Bolus < p_2) visitCount: 1 score: 0.59
rule: $\Diamond_{[?,?]}$((? > ?) ∧ (? < ?)) visitCount: 1 score: 0.43
rule: $\Diamond_{[?,?]}$((? < ?) ∧ (? > ?)) visitCount: 5 score: 0.65
rule: $\Diamond_{[?,?]}$((? > ?) ∨ (? < ?)) visitCount: 1 score: 0.12
rule: $\Diamond_{[t_3, t_4]}$((Bmi < p_3) ∧ (Hba1c > p_4)) visitCount: 1 score: 0.47
...

FIGURE 4.4: Example Partial Exploration Tree. Tree nodes contain the rule and MCTS components *visitCount* and *score*.

design is advantageous in an LDP setting where the number and accuracy of the queries is limited by the privacy budget.

Exploration Tree. An example partial exploration tree is shown in Figure 4.4. Each node records MCTS properties including node visit count and current score, which indicates its priority for exploration, explained in Section 4.2.2, as well as the rule structure. In a completed exploration tree, internal nodes contain incomplete rule templates where unfinished parts of the rules are represented using "?". Leaves contain either incomplete rule templates (indicating that a tree path was not fully explored and that node's children have not been visited), or completed rules. The completed rules constitute the learned population ruleset R_S. The rule discovery protocol searches G to iteratively build the exploration tree T and return the population rule set R_S.

Threat Model. We assume the network traffic is not observable to an adversary; our focus is on providing client privacy from the central server. In a setting where network traffic is exposed, it would be necessary to modify the protocol to ensure the communication pattern, timing, and packet sizes do not leak information about a client's rules. For our differential privacy notion, we quantify the unit of privacy as one rule, and we assume all rule structures are independent. In practice, a client may learn multiple rules that convey the same privacy information, so a privacy guarantee at the level of individual rules as the unit of privacy would be insufficient.

We assume all participating clients are honest—they follow the protocol as prescribed, keeping track of their own privacy loss budget, implementing randomized response as intended, and refusing to respond to any more queries once their privacy loss budget is expended. To keep things simple in our design and analysis, we assume all clients have the same privacy loss budget and each query uses the same per-query budget for all clients. An adversarial client could respond to queries in ways that would compromise the results, but we assume that in relevant clinical settings all participants would be motivated for the aggregate model to be as useful as possible.

Protocol Algorithm. The protocol, described in Algorithm 1, takes as input a rule grammar G; the valid rule threshold $\mathcal{V}$, the fraction of clients who must have a match to the rule structure for it to be considered viable in the protocol; the privacy loss budget ϵ; the number of clients n; and an exploration threshold θ. Details about θ are

Algorithm 1: Rule Discovery Protocol

```
1  protocol DiscoverRules($G$ (rule grammar), $\mathcal{V}$ (valid rule threshold),
     $\epsilon$ (privacy loss budget), $n$ (number of clients), $\theta$ (exploration threshold)):
2      $R_S \longleftarrow \emptyset$
3      $T \longleftarrow$ empty exploration tree
4      $b \longleftarrow T.root$
5      $plb \longleftarrow \epsilon$
6      while $plb > 0$ do
7          $b_{selected} \longleftarrow$ SelectNode($b$, $T$, $G$)
8          $b \longleftarrow$ ExpandNode($b_{selected}$, $T$, $G$)
9          $\hat{c}$, $plb \longleftarrow$ Query($b$, $plb$, $\epsilon$, $\mathcal{V}$, $n$, $\theta$, $R_S$)
10         Backpropagate($b$, $\hat{c}$, $T$)
11     end
12     return population ruleset $R_S$
13 end
```

explained in Section 4.2.6. In the start of the algorithm, the population ruleset R_S and exploration tree T are initialized, the current node b is set to the root of the tree, and a variable tracking the amount of privacy budget used, plb is initialized to the global privacy loss budget ϵ. The protocol loops iteratively through the four MCTS phases until the privacy loss budget has been used and the aggregated population ruleset R_S is returned.

To simplify the protocol design, we assume all clients have the same privacy budget, and every query is sent to every client. We also assume query executions are done completely; there are additional opportunities to save privacy loss budget by cutting off a query once enough responses have been received. If these simplifying constraints were removed, there are many opportunities to use the privacy loss budget more efficiently, such as querying subsets of clients or adjusting the privacy loss budget for a query as more information is learned from clients. We describe each phase of the protocol next.

4.2.2 Selection

In the first MCTS phase, the protocol selects a node to explore (Alg. 1, line 7). This function follows the traditional Selection implementation in MCTS, and pseudocode is available in Algorithm 2. The next node is recursively selected by either choosing the child node of the current node b that is unvisited, or choosing the child node that returns the maximum score according to the scoring function, discussed next. The selection function returns when a terminal node is reached.

Scoring. Scoring uses the classical UCT score [101], with the reward adapted to be the percent of clients who have a match to the rule structure (received from the clients' randomized responses and explained below in QueryClients). For node b,

$$score = \begin{cases} 0, & \mathit{If}\ \frac{\hat{c}}{n} < \mathcal{V} \\ \frac{\hat{c}}{n} + C_p \times \sqrt{\frac{v_{parent}}{v_b}}, & otherwise \end{cases} \tag{4.1}$$

where $\hat{c}$ is the client match count, C_p is a hyperparameter balancing the exploration and exploitation tradeoff in the MCTS search, v_{parent} is the visit count of the parent node, and v_b is the visit count of the current node.

Algorithm 2: SelectNode

```
Function SelectNode(node $b$, exploration tree $T$, grammar $G$):
    if $b$ is terminal then
        return $b$
    else if any child node of $b$ unvisited then
        SelectNode(unvisited child node)
    else
        for all child nodes of $b$ that are not completely explored do
            $b_{best} \longleftarrow$ child node with the maximum score according to
              Equation 4.1
        end
        SelectNode($b_{best}$)
    end
end
```

4.2.3 Expansion

Next, in the second phase the protocol expands reachable nodes and adds them to the exploration tree (Alg. 1, line 8). This function follows the traditional MCTS Expansion implementation in MCTS, and pseudocode is available in Algorithm 3. It either just returns $b_{selected}$, or chooses from among the child nodes according to a *selection policy*. This policy may select a node to expand *randomly* or based on the node that has the highest score (Equation 4.1).

Algorithm 3: ExpandNode

```
Function ExpandNode(node $b_{selected}$, exploration tree $T$, grammar $G$):
    if $b_{selected}$ is terminal or unvisited then
        return $b_{selected}$
    else
        if $b_{selected}$ has no children then
            Get all child nodes possible to visit using $G$ and add them to
              $b_{selected}$
        end
        return selectionPolicy($b_{selected}$.getChildren())
    end
end
```

4.2.4 Querying

In the next phase, clients are queried using randomized response (Alg. 1, line 9). This step is classically known as Simulation; our adaptation is illustrated in Algorithm 4.

Allocate Privacy Budget. The local privacy loss budget, β, to be used by each client for the query, is determined (Alg. 4, line 2). In the baseline method, a uniform budget is used for every query by just dividing the total budget by a pre-specified number of queries: $\beta = \epsilon/Q$, where Q is the number of queries. For the adaptive method, β is determined using the method described in Section 4.2.6.

Querying. β is used to send a query (in the form of a rule template) to all the clients and obtain an estimate of how many clients have a match to the rule structure

Algorithm 4: Query Clients for Matching Rules

```
1  protocol Query($b$, $plb$, $\epsilon$, $\mathcal{V}$, $n$, $\theta$, $R_S$):
2      $\beta \longleftarrow$ allocatePrivacyBudget($plb$, $\epsilon$, $\mathcal{V}, n, \theta$)
3      $t \longleftarrow b.rule$ // Get current rule structure
4      for $cl$ in $clients$ do
           // Query each client for structural rule match to $t$
5          $y$ += $cl$.QueryRuleMatch($t, \beta$)
6      end
7      $plb \longleftarrow plb - \beta$ // Update used budget
8      $\hat{c} \longleftarrow \frac{y-nq}{p-q}$ // Unbiased estimate of count
9      if $t$ is complete then
           // Determine privacy budget for learning parameters
10         $\beta_{param} \longleftarrow$ ParamPrivacyBudget ($t$, $plb$)
11         $plb \longleftarrow plb - \beta_{param}$ // Update used budget
12         $b.rule \longleftarrow$ QueryParameters($b.rule$,$\beta_{param}$)
13         $R_S.insert(b.rule)$ // Add completed rule to $R_S$
14     end
15     return $\hat{c}$, $plb$
16 end
```

contained at the selected node. The function first gets the partial rule template t from the current node b (Alg. 4, line 3). Next, it queries each client to get the number of yes responses, y, who have a match to t (line 4–line 6). Each client gives their binary (yes/no) response following a Direct Encoding Randomized Response method [102]. Additional details about the rule matching process are explained below. Then, the used privacy loss budget is updated (line 7) and the unbiased estimate of the count $\hat{c}$ is computed following traditional randomized response [102] (line 8). $\hat{c}$ is used to inform the score function (Equation 4.1) to guide the protocol in determining whether or not it should continue searching down a particular pathway. If a complete rule is found (one without any "?"), the parameters of the rule are queried (described below) and the rule is added to R_S (line 9 – line 14). Finally, $\hat{c}$ and plb are returned (line 15).

Client Rule Matching. In the query, the server sends out the rule template t to all the clients. The clients each check to see if they have any rules that contain a syntactic match to the template. A syntactic match is a structural rule match, in which the specified parts of t have matches to the client rule, and all other parts of the client rule (i.e., the "?" in t) are ignored. To account for possible semantic matches (equivalence relations following the defined STL logic [96], in which the client rule semantically has the same meaning as the template even though the syntactic structure of the rules may differ), we assume that all rule structures have been converted to a canonical set in the rule learning. Figure 4.5 shows an example of two client rules matching a template.

Parameter Querying. If a leaf node in the exploration tree has been reached (representing a completed rule structure, one without any "?" marks), the parameters of the discovered rule structure are queried and aggregated as well. We allocate the parameter budget β_{param} by using a small fixed constant multiplied by the number of parameters there are in the rule structure to fill in (Alg. 4, line 10). After updating the used budget plb (line 11), the clients are queried for their parameters using β_{param} (line 12). If a client does not have a rule match to t, they respond with

Template = $\square_{[?,?]}$(? > ? $\wedge$? < ?)

Client$_1$ Rule = $\square_{[0,1]}$(BG > 200 $\wedge$ basal < 0.02)

Client$_{94}$ Rule =$\square_{[0,2]}$(BG > 180 $\wedge$ A1CGoal < 0.06)

Figure 4.5: Example Rule Matching. Colors indicate the part of the rule to be matched. In the template, the variables have not yet been specified (part of the ?s), so the template matches client rules with different variables.

random (noised) parameter values. We aggregate parameters using a standard mean value estimation process using LaPlacian noise [102]. To aggregate the parameters, a percentile threshold τ is given, and a parameter value is selected at or below which (inclusive) τ% of the scores in the distribution may be found.

4.2.5 Backpropagation

In the last MCTS phase, (Alg. 1, line 10) scores are propagated up the exploration tree. This follows traditional Backpropagation in MCTS (pseudocode is available in Algorithm 5). Starting at the current node and continuing up through the node's parents (until the root node is reached), each node updates the following: the match count $\hat{c}$ is added to $b.responses$, a list tracking the previous yes responses, the number of visits $b.visitCount$ is incremented and the score of the current node, $b.score$ is updated using the scoring method (Equation 4.1).

Algorithm 5: Backpropagate

```
Function Backpropagate(node $b$, match count $\hat{c}$, exploration tree $T$):
    while $b.parent \neq None$ do
        Add $\hat{c}$ to $b.responses$
        $b.visitCount$ += 1
        Update $b.score$ according to Equation 4.1
        if $b$ is terminal or all children of $b$ completely explored then
            $b.completelyExplored \longleftarrow$ True
        end
        $b \longleftarrow b.parent$
    end
end
```

4.2.6 Adaptive Budget Allocation

We detail next how the privacy loss budget is dynamically allocated in the adaptive method. We define c as the true (unknown) count of how many clients have a match to the rule structure (t) at the current node (b) and $\hat{c}$ as the estimated count (obtained from noised client responses in the protocol). Based on the score function, any nodes that have client match counts $\hat{c}$ below the valid rule threshold $\mathcal{V}$ are ignored (not explored) in the searching, since they are unlikely to have clients with rule matches. As a result of noise being added to the client responses, there are two types of error that can occur in the searching: (1) Wasting queries searching down pathways which few clients have matches to ($\frac{\hat{c}}{n} \geq \mathcal{V}$ but there is no valid rule in the subtree) and

(2) failing to explore parts of the grammar that contain valid rules ($\frac{\hat{c}}{n} < \mathcal{V}$ but the subtree contains a rule where $\frac{c}{n} \geq \mathcal{V}$). We prioritize avoiding the second type of error, as sending a few more queries than necessary is better than missing entire subtrees of the grammar that may contain important and relevant rules.

To this end, we adaptively allocate our budget by finding the minimum budget per query, β, that ensures the probability of failing to explore a subtree that is likely to have valid rules is bounded by a user-specified exploration trade-off threshold, θ. Following typical LDP randomized response [102], a user outputs a response equal to the true response with probability p and a random value with probability q:

$$p = \frac{e^{\beta}}{1 + e^{\beta}} \tag{4.2}$$

$$q = 1 - p = \frac{1}{1 + e^{\beta}} \tag{4.3}$$

Given p and q, we can compute the estimated match count to a query, $\hat{c}$, as:

$$\hat{c} = y \times p + (n - y) \times q \tag{4.4}$$

where y is the number of yes responses returned from the random response mechanism. To find β, we formulate an optimization problem as follows:

$$\min_{\beta} \left(\int_{y=0}^{n} (P(\frac{\hat{c}}{n} < \mathcal{V} \mid \frac{c}{n} = \mathcal{V})) \right) \leq \theta \tag{4.5}$$

Since we have not actually sent a query yet, we do not have any responses from the clients and do not know the value of y. Therefore, we iterate over all possible values of y from 0 to the population size n. We assume the worst case scenario where the true percent $\frac{c}{n}$ is directly at the valid rule threshold $\mathcal{V}$. In summary, this equation seeks to find the minimum β, where, for all possible values of y, the probability that we falsely ignore this branch of the grammar is bounded by θ.

4.3 Experimental Evaluation

This section reports on the empirical evaluation of our framework. We first introduce the experimental setup (Section 4.3.1), and then evaluate our framework based on two criteria: coverage (Section 4.3.2) and clinical utility (Section 4.3.3).

4.3.1 Experimental Setup

Data. To evaluate our method, three different open source datasets were chosen to evaluate the applicability of our approach for different clinical use cases (e.g., across different domains and patient populations). Open data is necessary for reproducibility and means there are no actual privacy concerns with these data, but they are still representative of many sensitive and private datasets in the clinical setting.

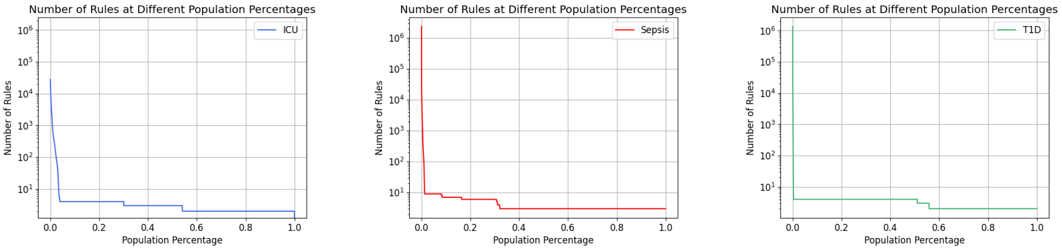
An overview of the datasets' characteristics are shown in Table 4.2. The Intensive Care Unit (ICU) dataset is from a study by Moss et al. [103] predicting inpatient deterioration. The Sepsis dataset [104] is from the PhysioNet/Computing in Cardiology Challenge 2019, in which they were trying to develop better algorithms for early detection of sepsis using physiological trace data. The Type I Diabetes dataset (T1D) [105], comes from the T1D Exchange Registry and collects longitudinal information of patients with T1D at each routine annual clinic exam between July 2007

Table 4.2: Clinical Dataset Details

Dataset	# Patients	# Features	Temporal Recording	# Timepoints	Ave. # Timepoints/Patient	Label	Negative Outcome % Patients	 % Timepoints
ICU	8000	57	Every 15 minutes	2,437,318	304.70	Deterioration	1.59	5.21×10^{-5}
Sepsis	40,336	35	Hourly	1,552,210	38.48	Sepsis	1.06	2.74×10^{-4}
T1D	34,013	40	Yearly	140,461	4.13	Hypoglycemia	5.26	1.27

Table 4.3: STL-Learned Ruleset Characteristics

Dataset	# Client Rules	Ruleset Size	Rules/Patient	Operators/Rule
ICU	598,699	34,208	74.85	2.51
Sepsis	4,420,910	2,344,179	109.60	2.97
T1D	2,105,755	1,353,598	61.91	4.35



(a) ICU

(b) Sepsis

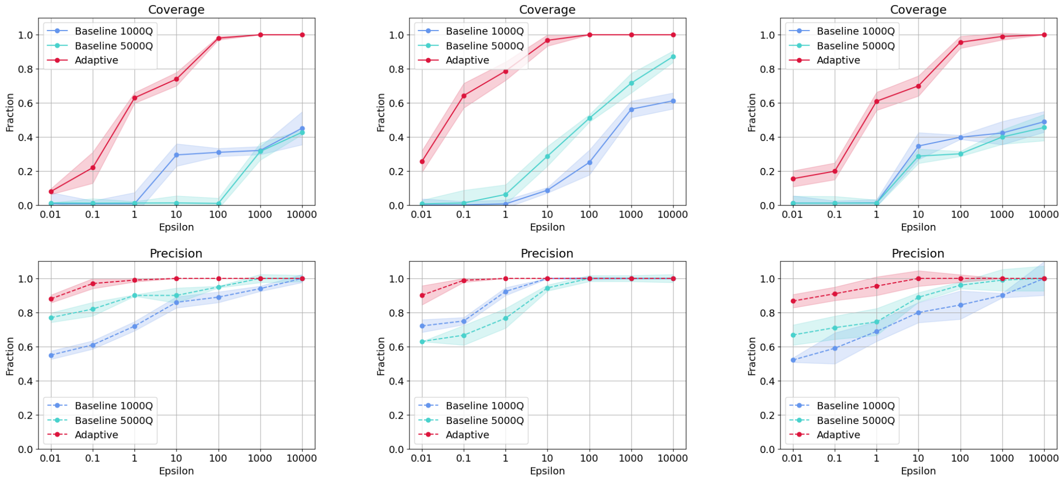
(c) T1D

Figure 4.6: Rule Breakdown by Population Percentage

to April 2018. We used the ICU dataset in developing our methods and for both validation and testing purposes, but only used the Sepsis and T1D datasets for final testing and evaluation to simulate a realistic scenario where the method cannot be tuned to particular data but must be determined without access to the intended data.

Rules. A set of rules was learned locally for each patient in each dataset using STL Learning. The STL Learners were trained for 1000 epochs to predict each dataset's outcome (Deterioriation, Sepsis, and Hypoglycemia for the ICU, Sepsis and T1D datasets respectively). No limits were set on the number of rules outputted from each learner, so clients have different numbers of rules in their local rulesets. Example learned rules were shown in Table 4.1. Table 4.3 reports ruleset characteristics. In the table, # Client Rules indicates the sum of the lengths of all individual client rulesets and Ruleset Size indicates the length of the total set of client rules (which has no duplicate client rules). Figure 4.6 shows the breakdown of how many rules there are at different population percentages for each ruleset.

Experimental Details. For all experiments we set C_p, the MCTS parameter balancing exploration vs. exploitation to $\frac{1}{\sqrt{2}}$, as it is a standard value often used in the UCT algorithm [101]. For the selection policy, we always select the branch with the highest score (Equation 4.1), and not randomly as is sometimes done in MCTS. All experiments were completed on a Mac Studio 20-core CPU with 64 GB of memory. Experiments were run 10 times, with the average result and standard deviation reported. We experiment with different values of the valid rule threshold $\mathcal{V}$, exploration threshold θ and privacy loss budget ϵ. For the baseline protocols, we tested different numbers of queries and selected Q as 1000 and 5000 as they provided the best coverage and utility results.



(a) ICU (b) Sepsis (c) T1D

FIGURE 4.7: Coverage ($\mathcal{V} = 1\%$, $\theta = 5\%$)

4.3.2 Coverage

Coverage provides a way to measure how well the learned population ruleset captures the breadth of rule types in the client rulesets. We quantify coverage in terms of two metrics: *coverage* and *precision*. In a population ruleset, R_S, we define a *valid* rule as one that is contained in the client rule structures of at least $\mathcal{V}$ percent of clients. Coverage provides a measure of the number of different rule structures learned by the private model contained in the original client rulesets:

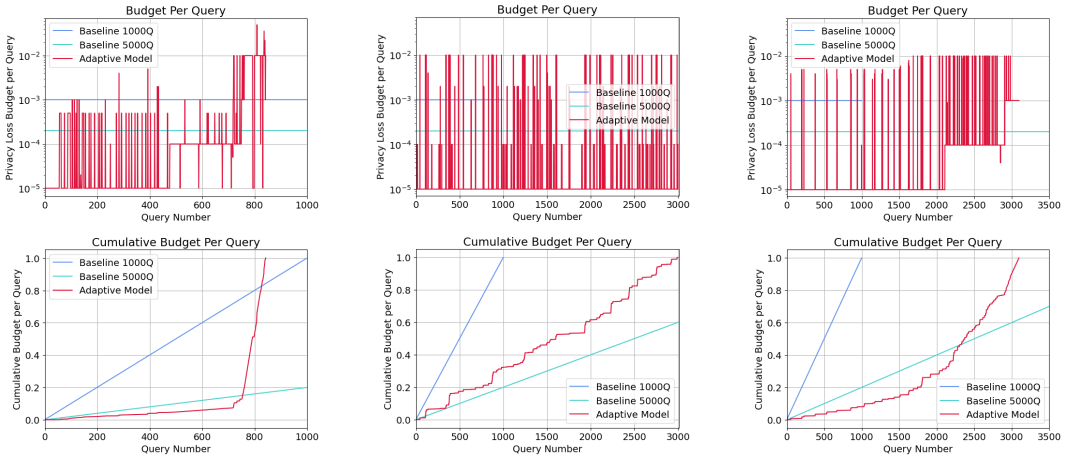
$$\text{Coverage} = \frac{|R_{valid}|}{|R_{C_{\mathcal{V}}}|} \tag{4.6}$$

where R_{valid} is the set of valid rules found in R_S and $R_{C_{\mathcal{V}}}$ is the set of client rules at the valid rule threshold. Precision provides a measure of quality of the overall population ruleset—Of the rules we found in R_S, how many are valid?:

$$\text{Precision} = \frac{|R_{valid}|}{|R_S|} \tag{4.7}$$

For reasonable privacy loss budgets, it will be impossible to learn all possible client rule structures, but our goal is to learn a set of rules that captures enough of the types of rules contained in the client rulesets to be clinically useful. Coverage, i.e., a wider breadth of rules, is important (as opposed to just learning the top k most common rules,) because the less common rule structures are usually the most informative [60, 61]. For example, when clinicians are trying to characterize new conditions or identify new associations indicative of various disease states, typically the less numerous rules characterizing the rare phenomenons are more useful than the more common ones. In this subsection, we evaluate coverage directly; later, in Section 4.3.3, we evaluate measures of clinical utility.

Comparing Protocols. Figure 4.7 compares the coverage and precision at different privacy loss budgets for the baseline protocols at 1000 (Baseline 1000Q) and 5000 (Baseline 5000Q) queries compared with the adaptive protocol. We set the valid rule threshold $\mathcal{V} = 0.01$ and use $\theta = 0.05$ for a controlled comparison and because these metrics align with clinical goals e.g., finding valid rules across 1% of the population



(a) ICU

(b) Sepsis

(c) T1D

FIGURE 4.8: Query Analysis ($\epsilon = 1$, $\mathcal{V} = 1\%$, $\theta = 5\%$). The x-axis is truncated at the max # of queries for the adaptive protocol in each graph to zoom in on interesting adaptive phenomenon (but the Baseline 5000Q lines continue uniformly to 5000 queries).

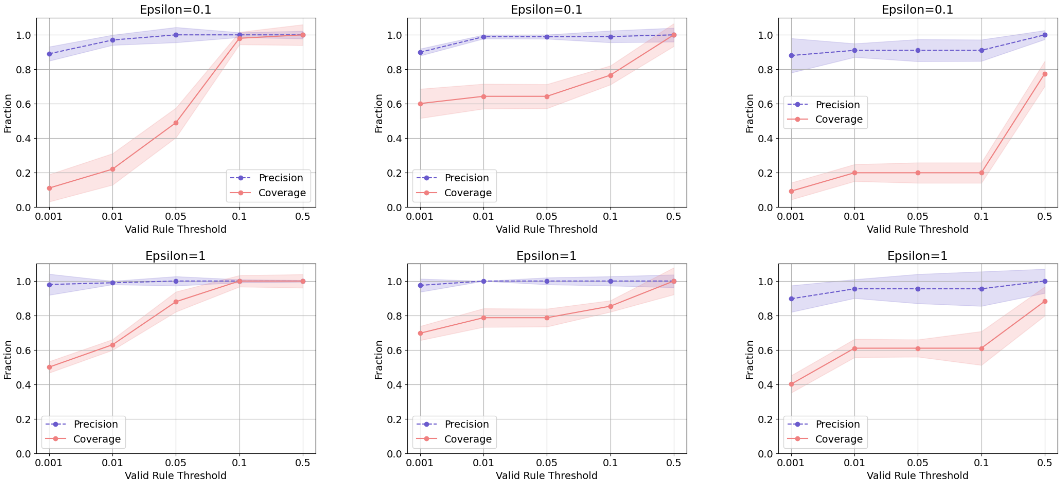
and only allowing a small amount of error at 5%. We experiment with different values of $\mathcal{V}$ and θ later. Across all rulesets, the adaptive protocol substantially outperforms the baseline ones.

Despite having the most rules, the adaptive protocol does the best on the Sepsis ruleset, reaching coverage of 80% at $\epsilon = 1$ with precision above 90% even for the lowest privacy loss budget we considered ($\epsilon = 0.01$). This is likely because the Sepsis ruleset has the least number of features and many very similar rule structures, requiring less searching through the STL grammar. Contrastingly, the ICU dataset does not have as many patients, resulting in higher noise addition and lower coverage; the T1D ruleset has the most complex rules, with highly disparate rule structures that require deep and wide exploration of the grammar tree, resulting in lower coverage.

Query Analysis. Figure 4.8 compares the privacy loss budget per query across all protocols. As to be expected, the adaptive budget jumps around between different amounts of budget, and the baseline methods use a uniform amount at each query. For the ICU and T1D rulesets, the adaptive budget tends to use higher amounts of budget in the later queries (e.g., after query number ∼1700 in T1D), whereas the budget jumps around fairly consistently through all the queries for sepsis.

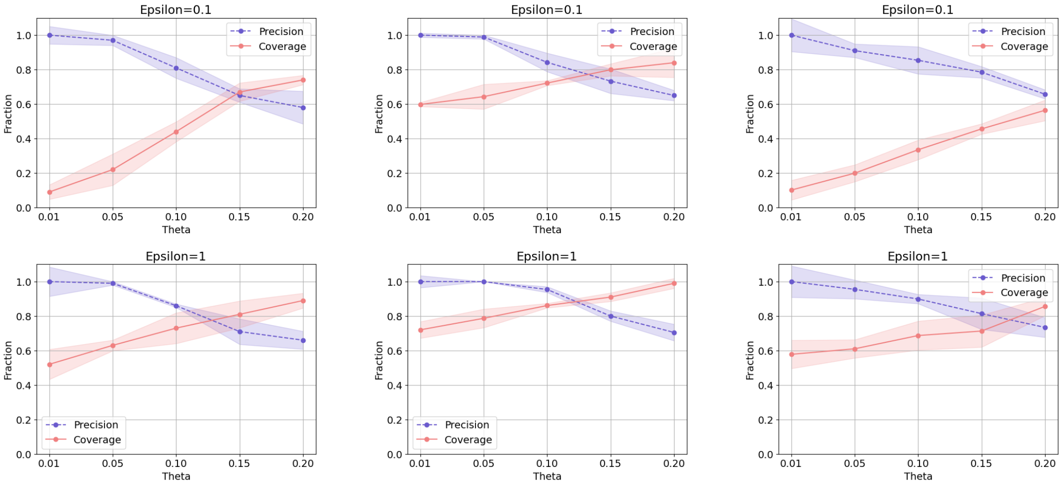
Comparing $\mathcal{V}$s. The valid rule threshold $\mathcal{V}$ influences the adaptive protocol's search as a result of the scoring function (Equation 4.1). Figure 4.9 looks at the effect of $\mathcal{V}$ on the coverage and precision. Across all the rulesets, the precision stays pretty stable and the coverage increases as $\mathcal{V}$ increases. This makes sense as there are fewer rules to find as $\mathcal{V}$ increases (see Figure 4.6 which shows the number of ground truth client rules contained at each population percentage). The coverage is decent at lower valid rule thresholds, providing evidence that the adaptive protocol is able to find rare client rules (e.g., rules that are contained by smaller percentages of clients).

Impact of Exploration Threshold (θ). θ is the exploration trade-off threshold and determines the probability of falsely ignoring a branch in the adaptive budget allocation. Figure 4.10 shows the effect of θ on the coverage results. There is a trade-off between the precision and coverage dependent on the amount of error allowed. Across all rulesets, as θ increases, the coverage increases, but this comes at a cost to



(a) ICU (b) Sepsis (c) T1D

FIGURE 4.9: Coverage at Different Valid Rule Thresholds $\mathcal{V}$ for $\epsilon = 0.1$ and $\epsilon = 1$



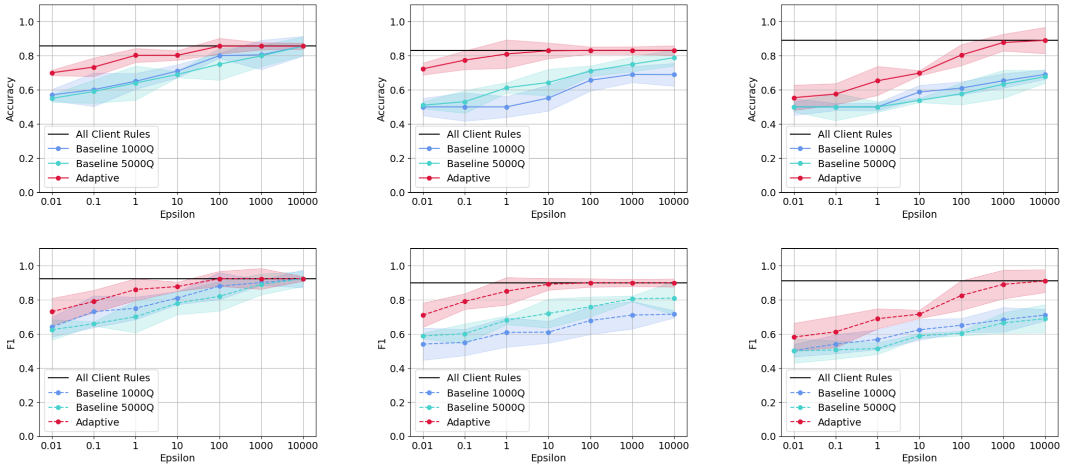
(a) ICU (b) Sepsis (c) T1D

FIGURE 4.10: Effect of Exploration Threshold θ on Coverage for $\epsilon = 0.1$ and $\epsilon = 1$

the precision, which drops significantly. This makes sense: as the searching permits more error, more rule structures are found (increasing the coverage) but more invalid rules, rules not actually contained by the client rulesets, are found and returned as a result of the noise in the randomized response querying. Alternatively, lower θ results in lower coverage, but higher precision. For clinical uses, it is better to favor higher precision since we want very few invalid rules in the learned population ruleset (and we note that this is why a more stringent bound of $\theta = 5\%$ were used for the experiments.)

4.3.3 Clinical Utility

For the second half of our experimental evaluation, we look at clinical utility, evaluating how useful the rulesets are for representative clinical applications. Since these applications are highly dependent on clinical context, we next describe motivating use



(a) ICU (b) Sepsis (c) T1D

FIGURE 4.11: Clinical Utility ($\mathcal{V} = 1\%$, $\theta = 5\%$)

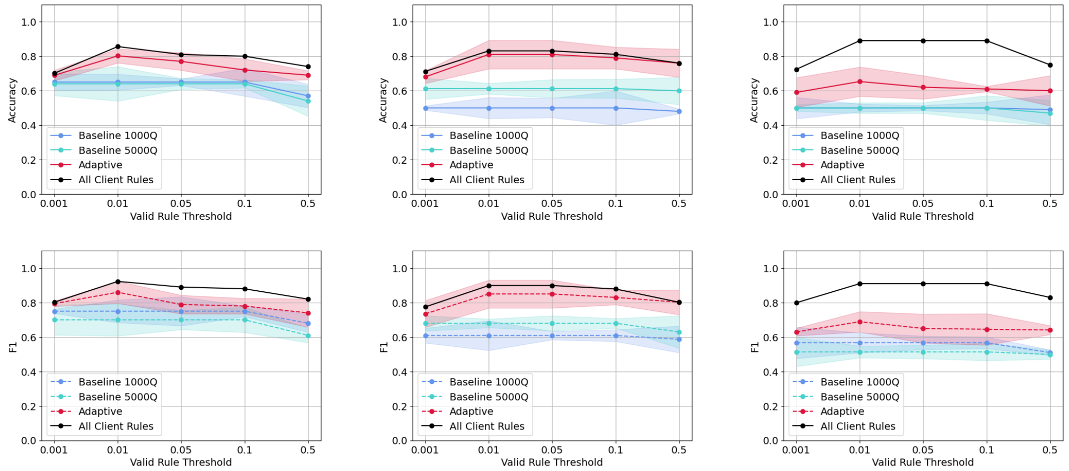
cases about how the rulesets may be used and to motivate how utility is evaluated within that context.

Intensive Care (ICU). This dataset seeks to understand predictors of clinical deterioration in the ICU. Deterioration refers to a patient's quick onset of a declining physical state that may result in life-threatening outcomes such as death. Symptoms of deterioration are highly variable between patients, especially because the condition may occur with little to no warning. For our experiments, we evaluate how predictive the learned population rules are at predicting ICU deterioration within the next 15 minutes for each patient.

Sepsis. The Sepsis dataset predicts the onset of sepsis. Sepsis is the body's extreme reaction to an infection it already had and is life-threatening as it can cause a cascade of biological damages such as tissue damage, shock, and organ failure. Similar to ICU deterioriation, sepsis occurs extremely quickly, and presents in highly variable ways between patients. In our experiments, and following what a CDSS would be used for, we evaluate how well the learned population rules predict sepsis within the next hour for each patient.

Type I Diabetes (T1D). The T1D dataset analyzes the glycemic control of individuals with Type I Diabetes. The T1D rules characterize the impact of patient behaviors on glycemic outcomes (which dictate better or worse control over the disease). A CDSS might wish to aggregate subgroup behaviors that characterize good or bad glycemic control. In our experiments, we evaluate how predictive the learned rules are for hypoglycemia as one indicator of glycemic control.

Metrics. For our use cases, clinical utility indicates the predictive quality of the rules in R_S when the rules are used to predict outcomes (deterioration, sepsis and hypoglycemia) on our clinical datasets. Since the rules characterize one of the label classes (the positive or negative class,) the quality of the rule can be judged based on its ability to correctly classify unseen data instances. The rules learned using the privacy-preserving protocols should have predictive quality similar to what would be obtained if the full client rulesets were available. Using a held-out validation data, we use the learned ruleset to predict the binary outcome for each patient using a weighted average taken from each rule in R_S, and compute balanced accuracy and F1 based on



(a) ICU (b) Sepsis (c) T1D

FIGURE 4.12: Comparing Utility Across $\mathcal{V}$s ($\epsilon = 1$, $\theta = 0.05$ for adaptive protocol)

these predictions.

Comparing Protocols. Figure 4.11 displays the utility results of accuracy and F1 scores for the adaptive protocol compared with the two baseline protocols. We set $\mathcal{V} = 0.01$ and $\theta = 0.05$ for a controlled comparison and because these metrics align with clinical goals e.g., finding valid rules across 1% of the population and only allowing a small amount of error at 5%. The black line in each figure displays the ground truth accuracy and F1 for the complete set of client rules at the selected $\mathcal{V}$ together (e.g., all rules from the client ruleset that 1% of the population have). Across all rulesets and privacy loss budgets, the adaptive protocol performs the best.

The ICU and Sepsis datasets have high accuracies and F1s even at lower privacy loss budgets ($\epsilon = 0.1$ and $\epsilon = 1.0$). However, the T1D dataset performs the worst relative to the other sets, for example with an accuracy of 0.59 for $\epsilon = 0.1$. This is likely because T1D individuals have the highest variability in terms of conditions and outcome presentation making it more difficult to correctly predict the outcome hypoglycemia.

Comparing $\mathcal{V}$s. Figure 4.12 looks at the effect of the valid rule threshold $\mathcal{V}$ on the clinical utility. All Client Rules refers to the complete set of client rules at the selected $\mathcal{V}$ together. Across all $\mathcal{V}$s, the adaptive protocol outperforms both baseline protocols. There is a trade-off between the size of $\mathcal{V}$ and the utility. As $\mathcal{V}$ becomes very small, e.g., $\mathcal{V} = 0.001$, accuracy and F1 decrease. This is likely because the number of rules increases substantially as $\mathcal{V}$ decreases, resulting in more varied symptom presentations and increasing disagreement in the rule predictions. In other words, too many unique rules (i.e., rules that only very few patients have) makes it difficult to find a consensus that generalizes across the entire patient cohort. Alternatively, as $\mathcal{V}$ becomes too large, e.g., $\mathcal{V} > 0.1$, accuracy and F1 also decrease. This is likely due to the fact that there are few rules contained by large numbers of patients, resulting in only a few general rules that are not as predictive of outcomes. As such, there is a sweet spot between the two extremes (which occurs between $\mathcal{V} = 0.01$ to 0.05 for these rulesets), where the population ruleset is generalizable enough to apply to the entire population, but not too general that its predictions are unhelpful. For all client rules, the highest accuracy and F1 are at $\mathcal{V} = 0.01$, which is also why this threshold was used for the prior experiments.

Summary of Findings. Our experiments demonstrate that the adaptive protocol parameters have a marked effect on the results. Varying θ results in a trade-off between precision and coverage. As the searching permits more error, more rule structures are found (increasing the coverage) but more invalid rules are also found (decreasing the precision) and vice-versa. Increasing $\mathcal{V}$ results in increased coverage, since there are less total rules to find, but varying $\mathcal{V}$ results in a trade-off for the utility. If $\mathcal{V}$ is too small there are too many rules and the population ruleset does not generalize; if $\mathcal{V}$ is too large, there are too few rules, and the population ruleset is too general to provide nuanced predictions. These findings are useful to help guide real-world instantiations of our protocol.

Across all experiments, the adaptive protocol outperforms both baseline ones. These results are very promising, because they demonstrate that the adaptive protocol is able to learn population rulesets with a breadth of rule types (high coverage) that are clinically useful (high clinical utility), even at low privacy budgets. Moreover, the adaptive protocol does well across all three rulesets, despite them having very different characteristics, including different application domains, ruleset sizes, population sizes, rule temporalities and complexity of the rule structures (e.g., length of rules, number of operators/rule.) This provides evidence that our protocol may generalize to many different distributed rule-based settings.

4.4 Related Work

We discuss the most relevant LDP prior work, focusing on term collection, tree-based methods and adaptive privacy budgeting. We note that the clinical rules we are collecting are different from the kinds of data collected in previous LDP work, and no previous work has developed LDP methods for learning logic-based rule structures or for CDSS-specific applications. Moreover, no previous methods when applied to the rule-based setting would present a perfect solution.

Frequent Term Collection. In the simplest case, one could treat the rules as strings and use prior methods for frequent term discovery and collection. Prior work in this area has developed LDP models in distributed settings for finding new frequent strings [28], discovering keystroke data [29, 30], text mining [31], frequent item mining [32, 33, 34] and data mining personal information [35]. These prior methods require large privacy budgets to discover new strings, especially long strings [106]. By taking advantage of the underlying logical structure in our rules (i.e., the rule grammar), we are able to learn long rule structures, even at low privacy budgets. Additionally, many of these methods seek to find only the most frequent strings or have poor trade-offs when it comes to finding less frequent strings. For example, [29] has high false positive rates for rare unknown words, and [28] has low utility for rare n-grams [106, 107]. We wish to find a *breadth* of rules, as the more rare rules tend to be the most informative [60]. By searching a rule grammar and balancing exploration vs. exploitation in our MCTS protocol, we are able to find rare rule structures, and not only the most frequent ones.

Tree-Based LDP Methods. There has also been prior work in distributed DP protocols that use tree-based methods, either for searching various data spaces or for allocating the privacy loss budget. PrivTrie collects new strings by iteratively building a tree and obtaining a rough estimate of each term prefix by adaptively grouping clients [30]. On a related note, LDPART, Zhao et al. develop a framework to publish location-record data. They use a hierarchical tree concept (called a partition tree) that extracts relevant location record information and partitions users into groups who are queried to determine whether to keep splitting the sub-nodes or not [108]. Our method searches a different data space (rule structures using a grammar) and we do not partition users, which allows them to be queried throughout multiple parts of the tree, and not only the subtree they were partitioned into. This is advantageous because we can use information about the history of previous responses to inform our searching (i.e., in the Backpropagation MCTS step,) and allows our users to be queried in multiple subtrees throughout the exploration tree, resulting in better generalizability of the final population ruleset.

Adaptive Privacy Budgeting. A straightforward method for adaptive budgeting is to allocate the privacy loss budget using a common scaling factor. For example, to adaptively allocate the budget at each iteration using an exponential decay mechanism [109] or at each level in a tree using an increasing geometric or Fibonacci factor [110]. Using a uniform scaling strategy as done by these methods is not applicable to our method as there is not a standard factor to guide the scaling (e.g., iteration or tree level). In general, our search dynamically jumps around to different parts of the exploration tree based on the scoring function so it would not make sense to allocate a standard budget amount (e.g., per iteration). Moreover, due to highly complex and varied rule structures, scores are highly variable across tree levels; as such, applying the same budget per level would not be ideal, since many nodes at the same level will have different sensitivities to noise.

Other methods determine the privacy loss budget based on specific algorithm computations, such as halting computations during algorithm runtime [111], algorithm learning rate for IoT blockchain data [112], ratio of eigenvalues in convolutional neural networks (CNNs) for DP-CNNs [55] and tree position and sensitivity for gradient boosted trees [113]. Although none of these methods are directly applicable to our problem as their computations are derived based on very different domains, they are similar in ideology to our approach: to adjust the privacy loss budget based on an algorithmic computation.

4.5 Summary

In this chapter, we developed and evaluated a locally differentally-private framework to learn population rulesets with high coverage and clinical utility for logic-based CDSS. This is a first work in a new direction about how to learn complex, structured rules with privacy. Although our work focuses on distributed CDSSs, our protocol can be adapted to fit other distributed settings where aggregating complex rules would be valuable, such as fraud detection and network security monitoring. Moreover, our methodology is amenable to any rule-based learner. Our experimental results demonstrate the promise of learning useful aggregate rulesets across populations while providing strong privacy guarantees.

This framework facillitates the sharing and aggregation of trajectories represented in logical structures, directly addressing model explainability and privacy challenges elucidated in Chapter 1. Additionally, this chapter introduces a natural bridge between Chapter 3 and Chapter 5 as it integrates differential privacy with explainable rule-based learning mechanisms in CDSS. In the next chapter (Chapter 5), we introduce a robust and interpretable risk stratification and phenotyping framework.

Chapter 5

Interpretable Learning for Risk Stratification (CARNA)

Heart failure (HF) is a complex disease condition with high morbidity and mortality [114].[1] On a fundamental level, HF is defined by the inability of the heart to deliver adequate blood flow to the body without an elevation in cardiac filling pressures [115]. Identifying high risk advanced HF patients early on in the care continuum is critical for timely allocation of advanced, life-saving therapies such as mechanical support, device implantation or transplant allocation. Due to high variability in patient conditions and complexity of the disease, determining patient risk involves a challenging, multi-faceted decision making process that places a high burden on clinicians [37]. Hemodynamic assessments can facilitate risk stratification and enhance understanding of HF trajectories [38]. Hemodynamics provide measures of cardiovascular function, and quantify distributions of pressures and flows within the heart and circulatory system [116]. However, obtaining a comprehensive picture of the patient state from these, particularly in the context of treatment-guiding outcomes, is difficult [39].

Many established HF risk scores such as the Seattle Heart Failure Risk model [40] use statistical or naive models which are difficult to optimize and may be prone to bias [41, 42, 43]. Machine learning (ML) models present a promising opportunity to outperform traditional risk assessment methods, especially when dealing with large, high-dimensional data [44]. However, despite the promise of machine learning for HF risk stratification, ML-based risk scores remain unpopular due to modest model performance and issues with model interpretability [45]. Moreover, no previous models (statistical of ML-based) incorporate invasive hemodynamics, or contain mechanisms to handle missing data.

To address these limitations, this chapter develops and validates an advanced HF hemodynamic risk stratification framework entitled CARNA (Characterizing Advanced heart failure Risk and hemodyNAmic phenotypes)[2]. We harness the explainability and expressivity of machine learned Multi-Valued Decision Diagrams (MVDDs) to learn a risk score that predicts the probability of patient outcomes, including mortality and rehospitalization, and provide descriptive patient phenotypes. MVDDs are discrete structures representing logical functions in directed, acyclic graphs where nodes represent features, edges represent logical operators (“and”, “or”) with parameter threshold values, and leaf nodes represent the final score classification [46]. An example MVDD is shown in Figure 5.1. Due to their use of logical operators, MVDDs

[1] This chapter is based on: Lamp, Josephine, Yuxin Wu, Steven Lamp, Prince Afriyie, Nicholas Ashur, Kenneth Bilchick, Khadijah Breathett et al. "Characterizing Advanced Heart Failure Risk and HemodyNAmic Phenotypes using Interpretable Machine Learning." American Heart Journal (2024).

[2] So named for the Roman healing goddess who presides over the heart.

can handle missing data, as multiple substitutable features may contribute to the same score prediction. Moreover, the "path" through the MVDD may be returned to provide a descriptive patient phenotype that characterizes the score. MVDDs have typically been applied in optimization and model checking contexts [117], and they do not inherently learn a risk stratification. Therefore, we develop an innovative method within our framework to first learn a risk stratification using a hierarchical clustering algorithm, and then develop a training regime to train the MVDDs on the learned risk scores and output explainable phenotypes. Although focused on advanced HF, CARNA is a general purpose risk stratification and phenotyping framework that can be used for other diseases and medical applications.

In summary, we present the following contributions:

1. We develop CARNA, an interpretable ML framework using Multi-Valued Decision Diagrams that works with missing data and includes invasive hemodynamics for risk stratifying advanced heart failure patients. In addition to producing a risk score, CARNA provides detailed patient phenotypes, i.e., sets of features and their thresholds that characterize a risk score.

2. We provide robust validation of the CARNA models using four independent HF cohorts, and compare them with six established HF risk scores and three traditional ML models. The CARNA models achieve high performance and outperform all benchmarks across metrics including Accuracy, Sensitivity, Specificity and AUC.

3. In order to facilitate practical use and promote open science, we provide an extensible, open-source tool implementation such that others can quickly and easily explore, extend, or prototype on top of the tool. In addition, our tool includes a deployed web server, which provisions live risk score prediction for ease of clinical use. All code is publicly available: `https://github.com/jozieLamp/CARNA`.

5.1 Preliminaries

Multi-Valued Decision Diagrams. MVDDs are discrete structures representing logical functions in directed, acyclic graphs where nodes represent features, edges represent logical operators ("and", "or") with parameter threshold values, and leaf nodes represent the final score classification [46]. As such, the "path" through the graph may be returned to provide a descriptive patient phenotype. An example MVDD is shown in Figure 5.1: the highlighted red path characterizes the high-risk score of 5 by the following phenotype: $Sex = Male \wedge BPSYS > 103.5 \wedge CPI > 0.621 \wedge (PAS > 74.5 \vee PCWP \leq 33) = Score\ 5$.

MVDDs are well suited to classification tasks and the representation of HF phenotypes over other black-box models because they allow increased flexibility in characterizing feature relationships and are highly interpretable [118]. This is advantageous over other models that do not provide any details about how a score was computed. Moreover, unlike other explainable models such as decision trees or random forests, MVDDs are resilient to missing data due to their use of logical operators; multiple substitutable features may contribute to the same prediction score. For example, in the above phenotype, PAS or PCWP may be used for calculation, and as such, when a feature is missing from the provided data, alternative features may be used to still allow for score prediction. This is advantageous in clinical scenarios where complete

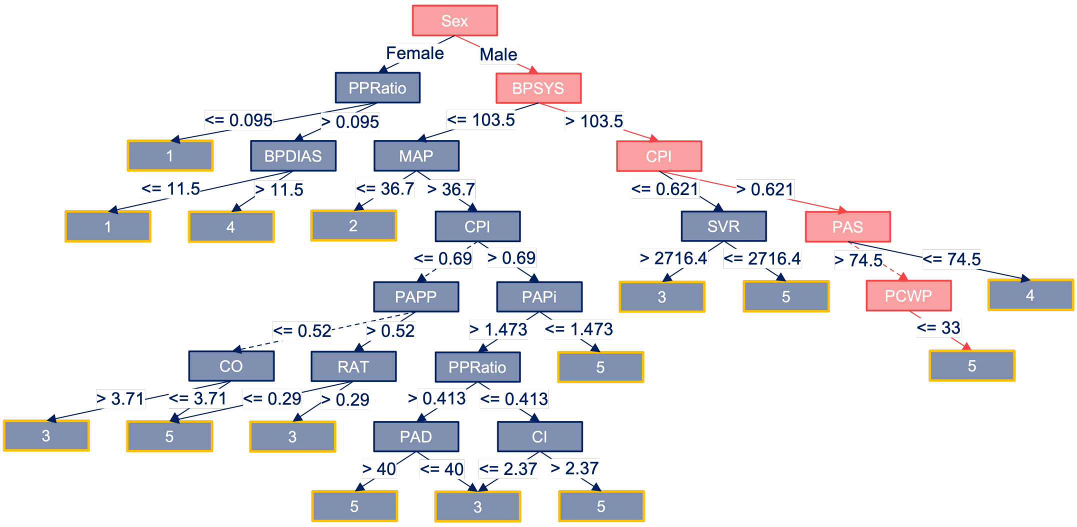


FIGURE 5.1: Example MVDD for the Invasive Hemodynamic Feature Set and DeLvTx Outcome. Dotted lines represent "or" boolean operators, and solid lines represent "and" boolean operators. The leaf nodes highlighted in yellow indicate the risk score. The highlighted red path indicates the example phenotype of Sex $=$ Male $\wedge$ BPSYS $> 103.5 \wedge$ CPI $> 0.621 \wedge$ (PAS $> 74.5 \vee$ PCWP ≤ 33) $=$ Score 5.

patient measurements may not be available and clinicians must make quick decisions on partial observations.

Despite these advantages, MVDDs have typically been used for optimization and model checking contexts [117], with limited use in medical classification and no applications to risk stratification. As such, we develop a training regime for MVDDs within our framework to learn risk scores and output HF phenotypes that characterize the predicted risk scores.

5.2 Outcomes and Cohort Selection

Outcomes. The primary outcome was a composite endpoint of death, left ventricular assist device (LVAD) implantation or heart transplantation (denoted as DeLvTx). A secondary outcome of rehospitalization within 6 months of follow up was included, as rehospitalizations have been shown to be predictive of adverse outcomes [62, 119].

Patient Cohorts. This study used 5 HF cohorts, three from randomized clinical trials and two from a real-world setting of a single quaternary healthcare system. Cohort characteristics are available in Table 5.1. We trained the model using the ESCAPE (Evaluation Study of Congestive Heart Failure and Pulmonary Artery Catheterization Effectiveness) trial [433 patients, mean age 56.1, 25.9% female], a randomized control trial studying the use of pulmonary artery catheters in severe HF patients [65]. The ESCAPE dataset contains a rich feature set of clinical and hemodynamic variables. Invasive hemodynamics (e.g., right atrial pressure (RAP) and pulmonary capillary wedge pressure (PCWP)) were recorded for 209 patients at baseline and prior to the removal of a heart catheter. Although smaller than the other cohorts, the ESCAPE dataset was selected for model training because, to the best of the authors' knowledge, it is the only cohort available with detailed invasive hemodynamics derived from a well-designed randomized HF clinical trial.

The other 4 cohorts were used for validation: The Beta-Blocker Evaluation of Survival Trial (BEST) [2707 patients, mean age 60.2, 21.9% female], was a randomized

TABLE 5.1: Characteristics of HF Cohorts

	ESCAPE [65]	**BEST [120]**	**GUIDE-IT [121]**	**UVA Shock**	**UVA Serial**
# Patients	433	2707	388	364	183
# Patients with Invasive Hemo	209	0	0	130	181
Baseline Data	Yes	Yes	Yes	Yes	Yes
Discharge Data	Yes	No	Yes	Yes	Yes
Total Records	866	2707	776	728	366
Total Data Missing (%)	7.8	2.0	15.1	10.4	7.3
Hemodynamics Missing (%)	12.0	N/A	N/A	5.9	9.2
Age (years)	56.1±13.9	60.2±12.3	62.2±13.9	59.4±18.5	60.6±15.1
Sex (% female)	25.9	21.9	66.2	35.2	43.2
Race (% white)	59.6	70.0	49.2	N/A	N/A
BMI (kg/m2)	28.4±6.7	N/A	31.2±8.6	29.8±8.8	30.5±8.0
LVEF (%)	19.3±6.6	23.0±7.3	24.0±8.2	31.7±17.4	31.3±18.0
SBP (mm Hg)	103.7±15.8	118.5±19.4	115.4±20.0	111.1±21.9	109.1±21.4
DBP (mm Hg)	64.1±11.5	71.9±11.7	70.2±13.5	62.2±15.5	59.9±17.2
Blood Urea Nitrogen (mg/dL)	36.3±22.5	24.6±15.3	31.3±22.6	34.9±24.2	39.1±25.7
Creatinine (mg/dL)	1.5±0.6	1.2±0.4	1.6±0.7	1.7±1.3	1.7±1.0
Potassium (mmol/L)	4.3±0.6	4.3±0.5	4.4±0.6	N/A	N/A
Sodium (mmol/L)	136.0±4.4	138.9±3.4	138.3±3.8	136.9±5.1	135.7±5.2
DeLvTx (%)	27.0	31.7	23.7	56.6	41.5
Rehospitalization (%)	57.0	62.9	51.8	47.5	78.7

N/A indicates data not available; LVEF = ejection fraction; SBP = systolic blood pressure; DBP = diastolic blood pressure; DeLvTx = composite endpoint of death, LVAD implantation or transplantation.

control trial that tested whether bucindolol hydrochloride reduced mortality among HF patients [120]. The Guiding Evidence Based Therapy Using Biomarker Intensified Treatment in Heart Failure (GUIDE-IT) [894 patients, mean age 61.5, 68% female] trial was a randomized controlled unblinded trial testing the efficacy and safety of adjusting therapy to maintain a N-terminal pro–B-type natriuretic peptide level of less than 1000 pg/ml in systolic HF patients [121]. We also performed external validation on two additional real-world cohorts from the University of Virginia: 1) a registry of cardiogenic shock patients [364 patients, mean age 59.4, 11.7% female], and 2) a registry of HF patients who had at least two serial right heart catheterizations for hemodynamic assessment during the same hospitalization [183 patients, mean age 60.6, 43.2% female].

Only New York Heart Association (NYHA) functional class III-IV were included in the study to ensure comparability. This study has been approved by the University of Virginia Institutional Review Board. The ESCAPE, BEST and GUIDE-IT data are available to other researchers for purposes of reproducing the results or replicating the procedure via data request from the National Heart, Lung, and Blood Institute Biological Specimen and Data Repository Information Coordinating Center.

Only the ESCAPE, UVA Cardiogenic Shock and UVA Serial Cardiac cohorts have invasive hemodynamics. GUIDE-IT had the highest percentage of missing data (15.07%), and ESCAPE had the highest percentage of missing hemodynamic data (12.04%). Additional characteristics for each of the cohorts are included in Table 5.1.

5.3 Methods

Method Overview. A high-level overview of the CARNA methodology is shown in Figure 5.2. First, the risk labels are generated (Section 5.3.2). Agglomerative clustering is used to stratify patients in all datasets into a specified number of cluster groups and risk categories are derived for each cluster (e.g., class 1-5 ordered numerically

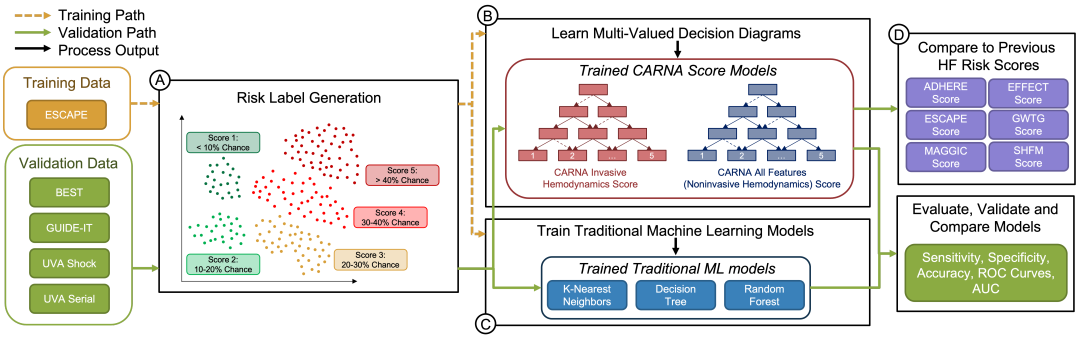


FIGURE 5.2: Overview of CARNA Methodology. (A) The risk labels are generated using a clustering-based derivation scheme using the training and validation datasets; (B) the training data is used to train the CARNA MVDD models as well as (C) three traditional ML models for comparison. Finally, the validation data is used to evaluate the performance of the models and the resulting CARNA risk scores are compared with six previous HF risk scores (D).

based on actual event rates). The output of this step is a set of risk labels that indicate the probability threshold of the outcome event happening (e.g., a patient record assigned to a class of 1 indicates an outcome probability of <10% for that patient). This clustering occurred twice (once for each feature set), and the probabilities from each cluster were derived for each outcome, resulting in a total of four risk label sets. Next, using the training data (ESCAPE cohort), Multi-Valued Decision Diagrams were trained to predict the risk labels (e.g., classes 1-5, Section 5.3.3). The trained MVDD models take in a set of features for a patient and output the predicted CARNA risk score. A total of four models were derived for each of the four risk label sets: one for each outcome (DeLvTx, Rehospitalization) and feature set (Invasive Hemodynamics, All Features) pair. Finally, the CARNA risk scores were evaluated using the four other validation cohorts and compared with traditional ML models (Section 5.3.4) and established HF risk scores (Section 5.3.5) based on their predictions of the risk classes. A step-by-step walkthrough of the methodology is provided next.

5.3.1 Data Preprocessing

For each dataset, we first preprocessed the data, including removing outliers (necessary to reduce bias in our ML training). If the dataset had multiple temporal recordings, the values recorded at baseline and discharge were treated as two separate records. Baseline values were included (as opposed to only discharge) as they have been shown to inform a range of hemodynamic and contractile metrics and are also important in predicting outcomes as shown in previous studies, e.g., [38]. Moreover, since it was our intention to provide a single point of care risk score that could make predictions even at initial hospital admission, we included baseline measurements in the models. This also increased the total number of training/validation records, especially helpful with the small training (ESCAPE) dataset. Which cohorts had baseline and discharge data, as well as the total number of records used is reported in Table 5.1.

Since our models support missing features, we did not impute or remove missing values from the data records. We also calculated noninvasive hemodynamics, additional metrics indicative of hemodynamic states, computed from features that were collected noninvasively. Examples include mean arterial pressure (MAP), cardiac power index (CPI), and pulse pressure (PP). These metrics were specifically selected a priori based on previous studies demonstrating incremental value in HF risk stratification [38, 122, 123]. Data was stratified into two subsets: one exploring phenotypes of invasive hemodynamics only, and the other for characterizing phenotypes between noninvasive hemodynamics and all available clinical variables including demographics, labs, and medications. Henceforth, we refer to these as the Invasive Hemodynamics and All Features feature sets, respectively.

5.3.2 Risk Label Generation

Each patient cohort had binary outcomes for the two endpoints (DeLvTx, rehospitalization), indicating if the outcome occurred or not. As such, there were no explicit risk thresholds for each of the patient records. Additionally, our MVDDs do not implicitly assign risk scores as a function of their learning. Since the goal of our approach was to generate a risk stratification and phenotyping score, the next step was to generate the categorical risk score values (i.e., 1–5) corresponding to real-valued outcome risks (e.g., 1 indicates a <10% risk of DeLvTx) for each record in the training and validation datasets. To this end, we reduced the dimensions of the covariates in the datasets using Principal Component Analysis (PCA) and then used a clustering approach to group patients and determine risk categorizations.

PCA. For each feature set and outcome, we performed PCA using two principal components to reduce the dimensions of the data. This was a necessary pre-step to reduce bias in the clustering, as clustering methods can be sensitive to outliers or slight changes in feature set distributions. Specifically, we use the LAPACK implementation of Singular Value Decomposition, following the PCA library available in the scikit-learn packages [124]. Since PCA cannot handle missing features, we imputed any missing values with the feature mean. We note however, that the original (non-imputed) datasets were used in the MVDD training steps later to ensure the models learned from the datasets with missing data. Importantly, the risk scores generated from this step were the labels used to train the MVDD models.

Hierarchical Clustering. Next, we clustered the patients into a specified number of groups using Agglomerative Clustering, a form of hierarchical clustering. Since the number of groups, k, is a hyperparameter, the users can select how many groups they wish to stratify the patients into. We argue this is an advantage of our approach because, based on details of the patient cohort being trained on or other user criteria (e.g., a clinician wish for only three risk groups), the number of risk groups can be adaptively selected. For our experimental purposes, we selected k as the optimal number of groups using the "Elbow" method, in which the sum of squares at each number of clusters is plotted on a graph [125]. The point on the graph where the slope changes from steep to shallow ("elbow" of the graph) indicates the optimal number of clusters to use. The clustering was performed across all datasets (including the four validation cohorts). In order to discriminate how well separated the clusters were, we computed the Hubert & Levin C Index for each feature set [126]. The C Index provides a metric to compare the dispersion of clusters compared to the overall dataset dispersion [127]. C Index should be minimized; a smaller index indicates more distinct (stable) clusters. Each cluster corresponds to one score value (e.g., five clusters for five score value assignments).

Derive Outcome Probabilities. From there, the outcome probability ranges for each score cluster were derived by computing the ground truth probability of the denoted outcome from the patients in each cluster. For example, cluster 1, corresponding to a score value of 1, had a ground truth probability of 0.041 for the DeLvTx outcome and Invasive Hemodynamics feature set; an outcome probability of <10% was derived. As a sanity check to ensure the derived score categories corresponded to the ground truth outcome probabilities across all the datasets, we reported the actual probabilities for each dataset in the Results. Finally, the score labels were assigned to each data record based on the associated cluster (e.g., a record in cluster 1 is assigned a score of 1). Using this process, we generated the risk score (labels) separately for each outcome and feature set, resulting in a total of four risk score label sets.

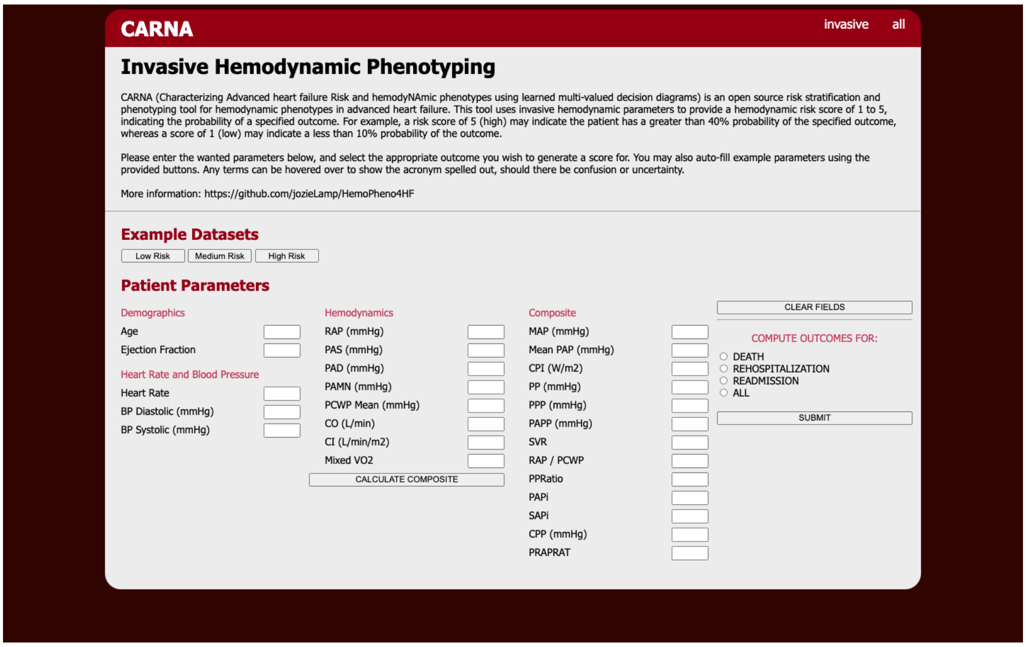
Label Method Reasoning. We decided to use this clustering approach because, in addition to risk stratifying patients, it uses an unsupervised method to holistically group patients, i.e., autonomously groups patients based on similar characteristics. This is highly advantageous over manually stratifying patients; manually grouping patients into risk groups is nontrivial due to large (potentially conflicting) sets of features and high variability in the presentation of patient conditions. Moreover, manual grouping is labor intensive (e.g., would require many clinician hours to characterize every patient's risk).

5.3.3 Learning Multi-Valued Decision Diagrams

Overall Training Details. As a reminder, the MVDDs were trained on the risk score labels (i.e., classes 1–5) generated during the previous step and the risk score labels indicate probability categories of outcomes. The resulting trained models take in a set of features for a patient and output the predicted CARNA risk score. All MVDDs were learned using an independent training set (ESCAPE dataset). To maximize the training capabilities of the small dataset, we used 5-fold cross validation, in which 80% of the data in the split was used for training and the other 20% was held out for validation purposes. A total of four models were derived: one for each outcome (DeLvTx, Rehospitalization) and feature set (Invasive Hemodynamics and All Features) pair.

MVDD Learning Process. Each MVDD was learned using a training process similar to the Iterative Dichotomiser 3 (ID3) multi-class decision tree algorithm [128]. Specifically, we learn a multi-class tree using the splitting criterion of gini index or entropy. Each time we add a node to the tree, we replace the boolean edge with logical operators (“and", “or") and select the operator that gives the best performance (e.g., lowest gini or entropy.) The MVDDs were trained iteratively until model convergence. The implementation was developed de novo in Python3 using publicly available packages [124].

Validating the MVDDs. After model training, we independently validated the models using the four other cohorts, which had not been used in the training phase. To assess the performance of our MVDDs, five receiver operator characteristic curves (ROC) for each risk class were plotted for each model based on the ground truth risk classes in the validation datasets. If the predicted risk class matched the ground truth risk class, this was considered a success for the ROC analysis. For example, in the case of class 1 patients, if the MVDD predicted class 1, it was considered a success, and if it predicted another class, it was considered a failure. The ROC curves were then constructed based on predictions of the risk classes, which is different from the conventional ROC method of predicting an actual event. To measure the overall model performance (e.g., as a summary metric across all risk classes,) we report a single averaged area under the curve (AUC) metric, calculated by taking the weighted average of the AUCs from each risk class, weighted by the number of individuals in each class. We also calculated accuracy, sensitivity and specificity in a similar manner. We note that ROC/AUC were used over a reclassification analysis



Invasive Hemodynamic Phenotyping

CARNA (Characterizing Advanced heart failure Risk and hemodyNAmic phenotypes using learned multi-valued decision diagrams) is an open source risk stratification and phenotyping tool for hemodynamic phenotypes in advanced heart failure. This tool uses invasive hemodynamic parameters to provide a hemodynamic risk score of 1 to 5, indicating the probability of a specified outcome. For example, a risk score of 5 (high) may indicate the patient has a greater than 40% probability of the specified outcome, whereas a score of 1 (low) may indicate a less than 10% probability of the outcome.

Please enter the wanted parameters below, and select the appropriate outcome you wish to generate a score for. You may also auto-fill example parameters using the provided buttons. Any terms can be hovered over to show the acronym spelled out, should there be confusion or uncertainty.

More information: https://github.com/jozieLamp/HemoPheno4HF

FIGURE 5.3: Example CARNA Web Portal – interface for predicting the invasive hemodynamic risk score.

due to limitations associated with reclassification such as systematic miscalibration on validation cohorts [129].

5.3.4 Comparison to Traditional Machine Learning Models

We compared the performance of CARNA models with traditional ML models, including K-nearest neighbors (KNN), Decision Trees (DT) and Random Forests (RF). Median imputation was used for any missing values. We followed the same training procedure used for the MVDDs; each model was trained on the ESCAPE dataset using 5-fold cross validation with a 80-20% split for training/validation. Performance was computed using the same metrics on the four validation cohorts. Additionally, to assess the concordance between the predicted risk and the ground truth outcomes, calibration plots were computed, using a bin size of 10.

5.3.5 Comparison to Other Heart Failure Risk Scores

For benchmark comparison, we compared our CARNA risk score models with six other established HF risk scores: ADHERE [130], EFFECT [131], ESCAPE [132], GWTG [133], MAGGIC [134], and SHFM [40]. We limited our comparison to the models predicting risk of mortality with similar feature sets and patient cohorts. In particular, we exclude scores that use biomarkers and pathology based features (e.g., QRS measurements) since those were not available in our cohorts. Since the comparison scores cannot handle missing data, missing values were imputed with the median. For each validation dataset, the predicted probability of an event was obtained from each score for each patient, and then a predicted class was assigned based on that probability. In other words, if the predicted probability of the event from the SHFM was 5% for a patient, we would say the SHFM predicted class 1, which had a probability range of 0-10% for an event. The accuracy of these other models for predicting the risk class (not the actual event) was again used for the comparison ROC analysis. To compare the AUCs between the established HF risk scores and CARNA, we

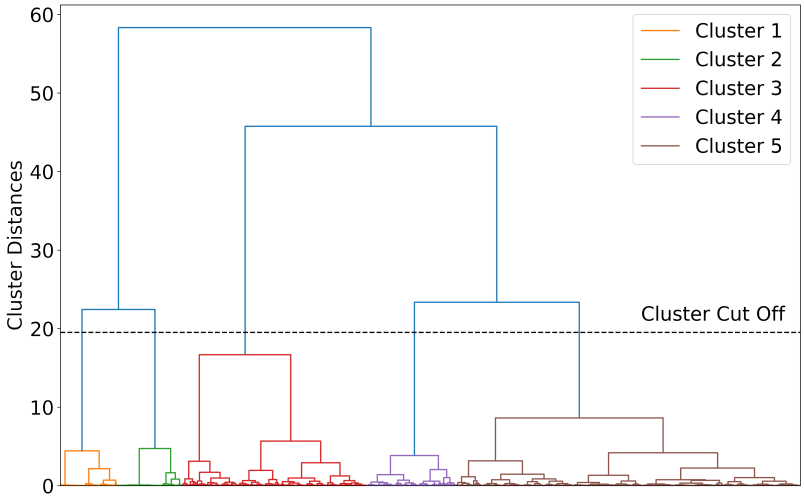


FIGURE 5.4: Agglomerative Clustering Dendogram for All Features feature set. Clusters are separated by horizontally dividing the top of the hierarchy based on the specified number of groups (5 in our case); this is illustrated by the horizontal dashed black line in the figures. Each leaf (end of the dendrogram) represents an individual data point.

performed hypothesis testing using the DeLong approach [135]. We report the scores' AUCs, the change in AUCs (CARNA AUC – other score AUC) and the p-value.

5.3.6 Open Source Tool Implementation

In order to promote open science, CARNA is an open source, extensible framework that others can easily use and build off of. Our implementation is developed in Python 3 using open source libraries. The tool package is clearly commented and includes a jupyter notebook runner file such that others can quickly and easily explore, extend, or prototype on top of the tool. In addition, our implementation includes a deployed web server which provides a live risk score prediction for ease of clinical use. An example web portal image is in Figure 5.3. All code is publicly available from the Github repository: https://github.com/jozieLamp/CARNA, and the live web server may be accessed here: http://hemopheno.pythonanywhere.com/.

5.4 Evaluation

5.4.1 Risk Label Generation Results

From the elbow plots, 5 was chosen as the optimal number of cluster groups corresponding to 5 risk categories. An example dendrogram displaying the cluster splits for the All Features feature set is shown in Figure 5.4. In hierarchical clustering methods, clusters are separated by horizontally dividing the top of the hierarchy based on the specified number of groups, illustrated by the horizontal dashed black line in the figures. **Our clusters are distinct with a high degree of separation, with low C Indexes of 0.063 for the Hemodynamics feature set and 0.051 for the All Features feature set.**

Table 5.2 reports the risk score meaning and corresponding real-valued average risk probabilities for each score category across all feature sets and outcomes. For example, for the Hemodynamics feature set and the DeLvTx outcome, a risk score of 3 indicates a 20–30% chance of the outcome, with a mean outcome probability of 0.245 computed from the patients in this cluster. For a sanity check, we also reported

TABLE 5.2: Risk Score Meaning and Ground Truth Risk Probabilities

DeLvTx Outcome

			Invasive Hemodynamics Cluster Means				All Features Cluster Means					
Risk Score	**Probability**	**Risk Category**	**Overall**	**ESCAPE**	**UVA Shock**	**UVA Serial**	**Overall**	**ESCAPE**	**BEST**	**GUIDE-IT**	**UVA Shock**	**UVA Serial**
1	<10%	Low	0.041	0.081	N/A	0.0	0.043	0.042	0.0	0.076	0.048	0.048
2	10 - 20%	Low - Intermediate	0.176	0.185	N/A	0.167	0.145	0.129	0.159	0.143	0.167	0.125
3	20 - 30%	Intermediate	0.245	0.25	0.227	0.259	0.255	0.265	0.275	0.235	0.201	0.299
4	30 - 40%	Intermediate - High	0.364	0.39	0.31	0.392	0.343	0.333	0.331	0.253	0.315	0.485
5	>40%	High	0.535	0.429	0.651	0.525	0.688	0.769	0.333	0.338	1.0	1.0

Rehospitalization Outcome

			Invasive Hemodynamics Cluster Means				All Features Cluster Means					
Risk Score	**Probability**	**Risk Category**	**Overall**	**ESCAPE**	**UVA Shock**	**UVA Serial**	**Overall**	**ESCAPE**	**BEST**	**GUIDE-IT**	**UVA Shock**	**UVA Serial**
1	<10%	Low	0.025	0.05	N/A	0.0	0.035	0.077	0.05	0.017	0.0	0.031
2	10 - 20%	Low - Intermediate	0.102	0.203	N/A	0.0	0.163	0.186	0.125	0.177	0.173	0.156
3	20 - 30%	Intermediate	0.261	0.276	0.216	0.291	0.286	0.309	0.275	0.259	0.275	0.312
4	30 - 40%	Intermediate - High	0.379	0.407	0.431	0.30	0.342	0.333	0.312	0.405	0.332	0.328
5	>40%	High	0.779	0.647	0.798	0.892	0.724	0.667	0.632	0.571	0.75	1.0

Tables display risk scores with corresponding outcome probability ranges and risk categories as well as cluster means, the ground truth mean outcome probability for each risk cluster in each dataset. Overall is the ground truth mean outcome probability across the entire cluster (i.e., across all datasets in the cluster). N/A = no data points assigned to that cluster; DeLvTx = composite endpoint of death, LVAD implantation or transplantation.

the average risk probabilities for each dataset individually. **These results provide evidence that the risk ranges correspond to the real observed risk in the patient cohorts.**

5.4.2 Learned MVDDs

We generated a total of four MVDD models for each of the feature sets (Invasive Hemodynamics and All Features) and outcomes (DeLvTx and Rehospitalization). The Invasive Hemodynamic models use a combination of 28 features that include basic demographics, invasive and noninvasive hemodynamics; the All Features models use a combination of 66 features across demographics, labs, medications, exercise, quality metrics, other medical diagnostics and noninvasive hemodynamics. We note that these are the *maximum* number of features per model and actual prediction paths through the MVDDs use smaller subsets with interchangeable combinations of features (e.g., the features that may be "or-ed" together along a path that provide choices for which feature is used for prediction in the phenotype.)

5.4.3 MVDD Performance

Table 5.3 presents the validation performance summary. The UVA Cardiogenic Shock and Serial Cardiac cohorts were used to validate the invasive hemodynamics models, since they were the only cohorts with invasive hemodynamics; all 4 validation cohorts were used to validate the All Features models. Figures 5.5 and 5.6 show the ROC curves and AUC values for each risk class for the Invasive Hemodynamics and All Features sets, respectively. Figures 5.7 and 5.8 show stacked bar graphs comparing the real vs. predicted risk categories for the Invasive Hemodynamics and All Features sets, respectively. Across all outcomes, our validation models performed extremely well with accuracies of 0.896±0.074 to 0.969±0.081 for the Invasive Hemodynamics feature set and 0.858±0.067 to 0.997±0.070 for the All Features feature set. **These validation results provide evidence that the CARNA models yield robust risk stratification.**

Table 5.3: Model Performance Summary (Validation Data)

Invasive Hemodynamic Feature Set

Outcome	Dataset	Accuracy	Averaged AUC	Sensitivity	Specificity
DeLvTx	UVA Shock	0.947±0.107	0.938±0.106	0.915±0.103	0.961±0.108
	UVA Serial	0.969±0.081	0.965±0.080	0.950±0.079	0.980±0.081
Rehospitalization	UVA Shock	0.907±0.102	0.861±0.096	0.791±0.086	0.935±0.105
	UVA Serial	0.896±0.074	0.896±0.074	0.852±0.070	0.940±0.078

All Features Feature Set

Outcome	Dataset	Accuracy	Averaged AUC	Sensitivity	Specificity
DeLvTx	BEST	0.997±0.037	0.994±0.037	0.990±0.037	0.998±0.037
	GUIDE-IT	0.997±0.070	0.996±0.070	0.995±0.069	0.998±0.070
	UVA Shock	0.865±0.049	0.871±0.050	0.811±0.045	0.931±0.054
	UVA Serial	0.858±0.067	0.871±0.068	0.815±0.063	0.927±0.073
Rehospitalization	BEST	0.997±0.037	0.994±0.037	0.990±0.037	0.998±0.037
	GUIDE-IT	0.997±0.070	0.996±0.070	0.995±0.069	0.998±0.070
	UVA Shock	0.891±0.051	0.895±0.051	0.846±0.048	0.944±0.054
	UVA Serial	0.890±0.070	0.798±0.061	0.653±0.044	0.942±0.075

Table displays value ± confidence interval. DeLvTx = composite endpoint of death, LVAD implantation or transplantation.

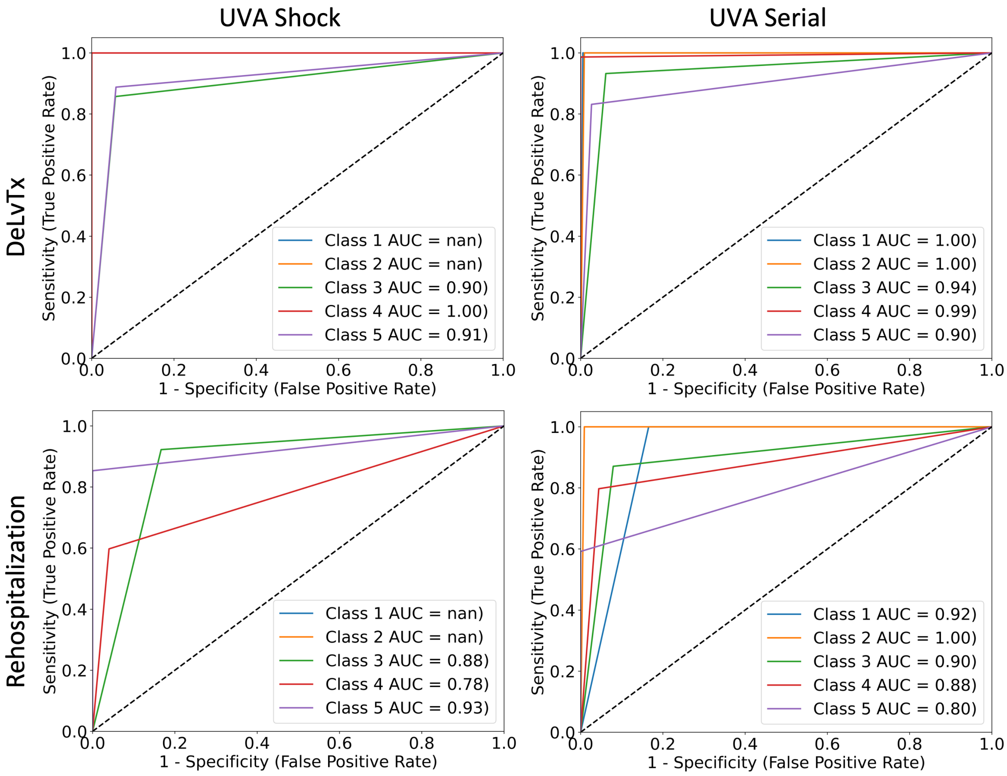


Figure 5.5: ROC Curves for Validation Datasets and Invasive Hemodynamics Feature Set. nan = no data in that class.

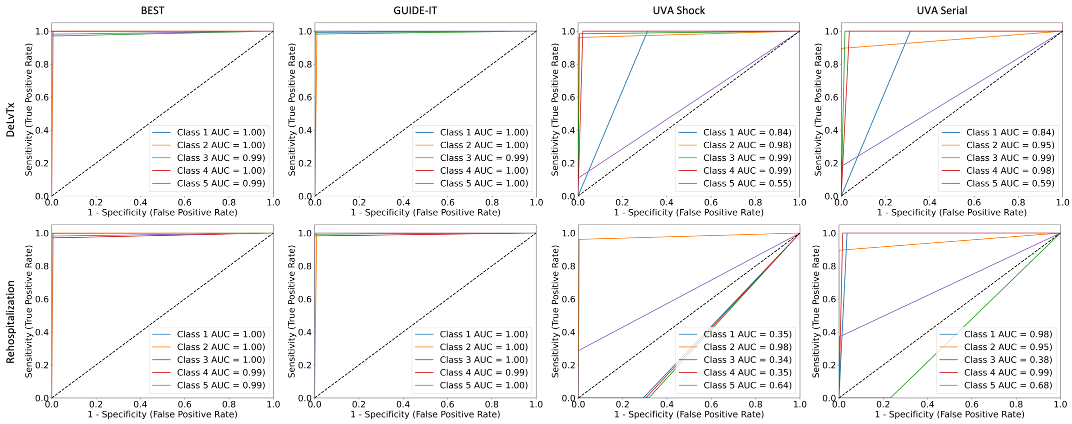


FIGURE 5.6: ROC Curves for Validation Datasets and All Features Feature Set.

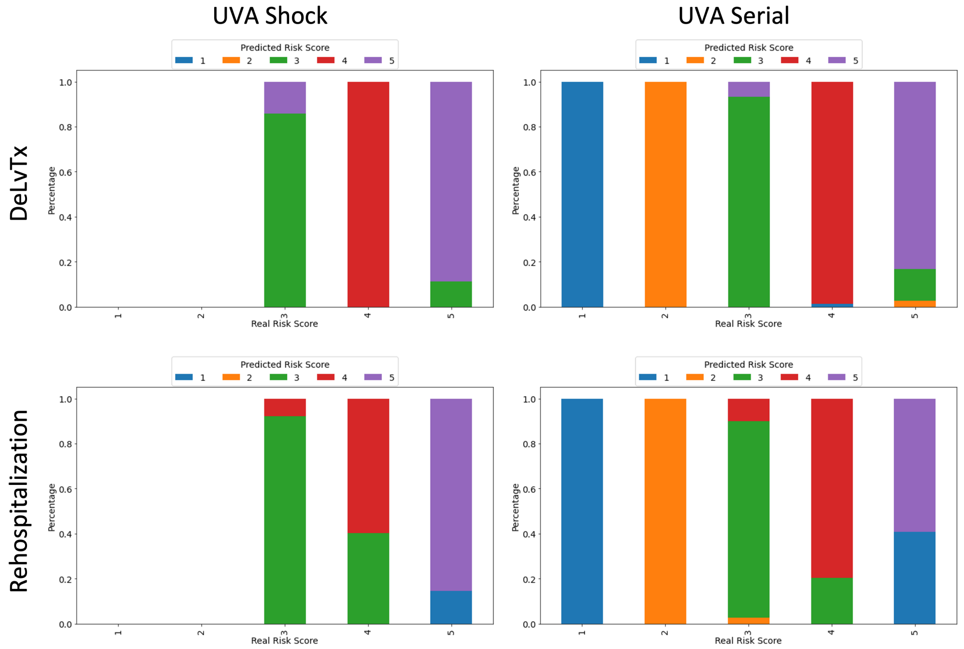


FIGURE 5.7: Stacked Bar Graphs Comparing Real Risk Scores vs. Predicted Risk Scores for Validation Datasets and Invasive Hemodynamics Feature Set

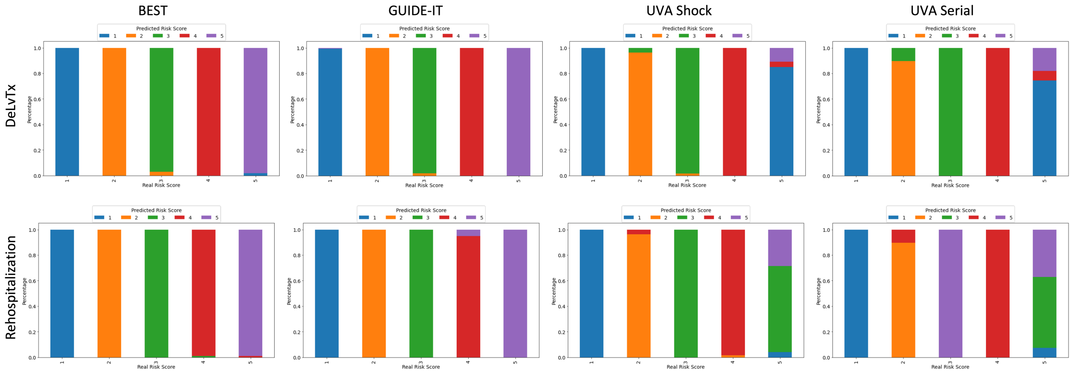


FIGURE 5.8: Stacked Bar Graphs Comparing Real Risk Scores vs. Predicted Risk Scores for Validation Datasets and All Features Feature Set

Table 5.4: CARNA Comparison to Traditional ML Models - Invasive Hemodynamics Feature Set

DeLvTx Outcome

Dataset	Model	Accuracy	Averaged AUC	Sensitivity	Specificity
UVA Shock	CARNA	**0.947±0.107**	**0.938±0.106**	**0.915±0.103**	**0.961±0.108**
	KNN	0.660±0.064	0.471±0.027	0.154±0.094	0.789±0.086
	DT	0.759±0.081	0.628±0.057	0.405±0.049	0.85±0.094
	RF	0.754±0.08	0.637±0.059	0.426±0.043	0.849±0.094
UVA Serial	CARNA	**0.969±0.081**	**0.965±0.080**	**0.950±0.079**	**0.980±0.081**
	KNN	0.685±0.051	0.5±0.001	0.199±0.065	0.801±0.065
	DT	0.776±0.062	0.649±0.045	0.438±0.029	0.86±0.071
	RF	0.772±0.061	0.628±0.042	0.4±0.037	0.855±0.07

Rehospitalization Outcome

Data Set	Model	Accuracy	Averaged AUC	Sensitivity	Specificity
UVA Shock	CARNA	**0.907±0.102**	**0.861±0.096**	**0.791±0.086**	**0.935±0.105**
	KNN	0.66±0.064	0.471±0.027	0.154±0.094	0.789±0.086
	DT	0.759±0.081	0.628±0.057	0.405±0.049	0.85±0.094
	RF	0.754±0.08	0.637±0.059	0.426±0.043	0.849±0.094
UVA Serial	CARNA	**0.896±0.074**	**0.896±0.074**	**0.852±0.070**	**0.940±0.078**
	KNN	0.685±0.051	0.5±0.001	0.199±0.065	0.801±0.065
	DT	0.776±0.062	0.649±0.045	0.438±0.029	0.86±0.071
	RF	0.772±0.061	0.628±0.042	0.4±0.037	0.855±0.07

Table reports value±confidence interval, bolded values indicate highest scoring item in each block. KNN = K-Nearest Neighbor; DT = Decision Tree, RF = Random Forest; DeLvTx = composite endpoint of death, LVAD implantation or transplantation.

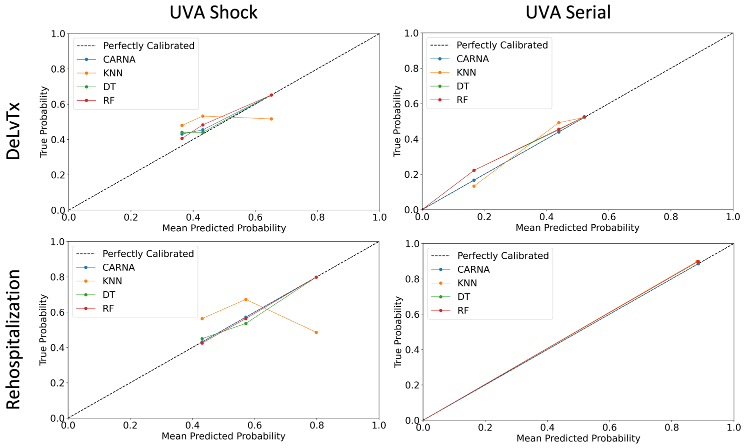


Figure 5.9: Calibration Plots for Invasive Hemodynamics using bin size of 10. True probability is the fraction of positives per bin.

5.4.4 Comparison to Traditional ML Models

For additional comparison, the performance of the CARNA models was compared with traditional ML models, including K-Nearest Neighbors (KNN), Decision Trees (DT) and Random Forests (RF). For the Invasive Hemodynamics feature set, performance is reported in Table 5.4 and calibration plots are shown in Figure 5.9. For the All Features feature set, performance is reported in Table 5.5 and calibration plots are shown in Figure 5.10. In the calibration plots, some bins have no samples, hence why some plots do not have complete points in the line graphs. Of the traditional models, RFs followed by DTs tend to perform the best. **Across all feature sets, outcomes and datasets, the CARNA models outperform traditional ML models**.

TABLE 5.5: CARNA Comparison to Traditional ML Models - All Features Feature Set

DeLvTx Outcome

Data Set	Model	Accuracy	Averaged AUC	Sensitivity	Specificity
BEST	CARNA	**0.997±0.037**	**0.994±0.037**	**0.990±0.037**	**0.998±0.037**
	KNN	0.822±0.03	0.677±0.022	0.525±0.008	0.83±0.03
	DT	0.831±0.03	0.652±0.021	0.469±0.009	0.835±0.03
	RF	0.844±0.031	0.722±0.025	0.602±0.017	0.842±0.031
GUIDE-IT	CARNA	**0.997±0.070**	**0.996±0.070**	**0.995±0.069**	**0.998±0.070**
	KNN	0.849±0.058	0.575±0.027	0.295±0.045	0.854±0.059
	DT	0.852±0.058	0.572±0.026	0.277±0.047	0.866±0.06
	RF	0.955±0.067	0.719±0.046	0.475±0.016	0.964±0.067
UVA Shock	CARNA	**0.865±0.049**	**0.871±0.050**	**0.811±0.045**	**0.931±0.054**
	KNN	0.349±0.032	0.391±0.027	0.042±0.055	0.624±0.029
	DT	0.386±0.028	0.319±0.035	0.135±0.049	0.596±0.025
	RF	0.604±0.026	0.377±0.029	0.016±0.057	0.771±0.042
UVA Serial	CARNA	**0.858±0.067**	**0.871±0.068**	**0.815±0.063**	**0.927±0.073**
	KNN	0.346±0.044	0.409±0.034	0.022±0.077	0.612±0.037
	DT	0.459±0.023	0.383±0.038	0.208±0.061	0.596±0.035
	RF	0.339±0.045	0.4±0.035	0.009±0.079	0.606±0.037

Rehospitalization Outcome

Data Set	Model	Accuracy	Averaged AUC	Sensitivity	Specificity
BEST	CARNA	**0.997±0.037**	**0.994±0.037**	**0.990±0.037**	**0.998±0.037**
	KNN	0.822±0.03	0.677±0.022	0.525±0.008	0.83±0.03
	DT	0.831±0.03	0.652±0.021	0.469±0.009	0.835±0.03
	RF	0.844±0.031	0.722±0.025	0.602±0.017	0.842±0.031
GUIDE-IT	CARNA	**0.997±0.070**	**0.996±0.070**	**0.995±0.069**	**0.998±0.070**
	KNN	0.611±0.033	0.646±0.038	0.489±0.01	0.803±0.054
	DT	0.612±0.033	0.582±0.028	0.369±0.036	0.794±0.053
	RF	0.611±0.033	0.623±0.035	0.446±0.023	0.801±0.054
UVA Shock	CARNA	**0.891±0.051**	**0.895±0.051**	**0.846±0.048**	**0.944±0.054**
	KNN	0.349±0.032	0.391±0.027	0.042±0.055	0.624±0.029
	DT	0.386±0.028	0.319±0.035	0.135±0.049	0.596±0.025
	RF	0.604±0.026	0.377±0.029	0.016±0.057	0.771±0.042
UVA Serial	CARNA	**0.890±0.070**	**0.798±0.061**	**0.653±0.044**	**0.942±0.075**
	KNN	0.346±0.044	0.409±0.034	0.022±0.077	0.612±0.037
	DT	0.459±0.023	0.383±0.038	0.208±0.061	0.596±0.035
	RF	0.339±0.045	0.4±0.035	0.009±0.079	0.606±0.037

Table reports value±confidence interval, bolded values indicate highest scoring item in each block. KNN = K-Nearest Neighbor; DT = Decision Tree, RF = Random Forest; DeLvTx = composite endpoint of death, LVAD implantation or transplantation.

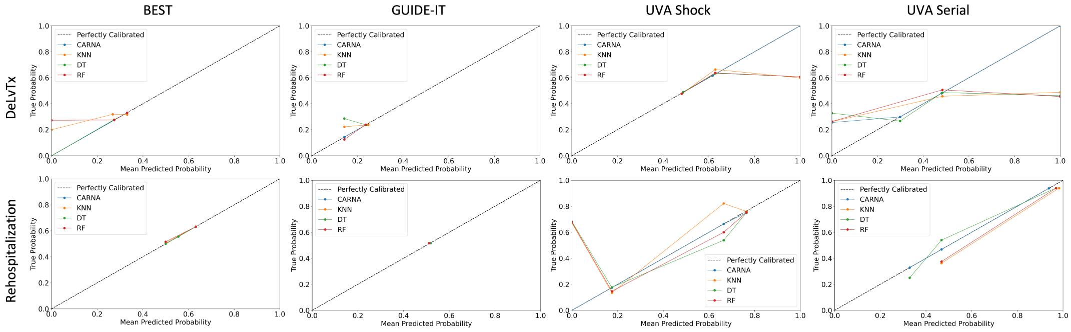


FIGURE 5.10: Calibration Plots for All Features using a bin size of 10. True probability is the fraction of positives per bin.

5.4.5 Comparison to Previous HF Risk Scores

Benchmark comparison between the CARNA risk scores and 6 other previously developed HF risk scores are shown in Table 5.6. The table reports the AUCs for each of the HF risk scores on all datasets. Table 5.7 displays results of the hypothesis testing between CARNA and previous scores. The delta AUC and p-values are reported; a p-value of <0.05 indicates there is a significant difference between the two scores. **CARNA outperforms all previous HF risk scores.**

TABLE 5.6: Comparison to Previous Scores - AUC for Outcome Mortality

Score	**Median Follow-Up**	**Dataset** ESCAPE	BEST	GUIDE-IT	UVA Shock	UVA Serial
CARNA - Hemo	6 months	0.952±0.091	N/A	N/A	**0.938±0.106**	**0.965±0.080**
CARNA - All Fts	6 months	**0.978±0.065**	**0.994±0.037**	**0.996±0.070**	0.871±0.050	0.871±0.068
ADHERE [130]	5.85 days	0.595±0.029	0.576±0.015	0.601±0.021	0.526±0.013	0.574±0.030
EFFECT 30D [131]	30 days	0.550±0.021	0.610±0.018	0.635±0.024	0.584±0.024	0.610±0.037
EFFECT Y1 [131]	1 year	0.548±0.021	0.638±0.020	0.632±0.024	0.612±0.027	0.644±0.043
ESCAPE [132]	6 months	0.681±0.057	0.587±0.016	0.715±0.043	0.595±0.025	0.565±0.029
GWTG [133]	4 days	0.601±0.030	0.538±0.010	0.537±0.013	N/A	N/A
MAGGIC Y1 [134]	2.5 years	0.640±0.035	N/A	0.689±0.029	0.678±0.034	N/A
MAGGIC Y3 [134]	2.5 years	0.640±0.035	N/A	0.689±0.029	0.678±0.034	N/A
SHFM Y1 [40]	1 year	0.623±0.033	0.613±0.018	0.623±0.023	0.587±0.024	0.588±0.033
SHFM Y3 [40]	3 years	0.623±0.033	0.616±0.018	0.625±0.023	0.588±0.024	0.584±0.032
SHFM Y5 [40]	5 years	0.622±0.033	0.615±0.018	0.619±0.023	0.573±0.022	0.579±0.032

Table displays AUC±confidence interval; bolded values indicate highest performing score for each dataset. N/A = score could not be calculated for the dataset; Hemo = Invasive Hemodynamic; All Fts = All Features; 30D = 30-day mortality; Y1 = 1 year mortality; Y3 = 3 year mortality; Y5 = 5 year mortality.

TABLE 5.7: Hypothesis Testing Between CARNA and Comparison HF Scores

	Invasive Hemodynamic Feature Set			**All Features Feature Set**				
Score	**Dataset** ESCAPE	UVA Shock	UVA Serial	**Dataset** ESCAPE	BEST	GUIDE-IT	UVA Shock	UVA Serial
ADHERE [130]	-0.357, 0.262	-0.412, <0.001	-0.391, 0.413	-0.383, 0.031	-0.418, <0.001	-0.395, 0.005	-0.345, <0.001	-0.297, 0.413
EFFECT 30D [131]	-0.402, 0.881	-0.354, <0.001	-0.355, 0.315	-0.163, 0.020	-0.428, <0.001	-0.384, 0.069	-0.361, <0.001	-0.287, 0.315
EFFECT Y1 [131]	-0.404, 0.832	-0.326, <0.001	-0.321, 0.028	-0.430, 0.026	-0.356, <0.001	-0.364, 0.096	-0.259, <0.001	-0.227, 0.028
ESCAPE [132]	-0.271, 0.008	-0.343, <0.001	-0.400, 0.311	-0.297, <0.001	-0.407, <0.001	-0.281, <0.001	-0.276, <0.001	-0.306, 0.311
GWTG [133]	-0.351, 0.593	N/A	N/A	-0.377, 0.002	-0.456, 0.001	-0.459, 0.001	N/A	N/A
MAGGIC Y1 [134]	-0.312, 0.151	-0.260, 0.018	N/A	-0.338, 0.094	N/A	-0.307, 0.048	-0.193, 0.018	N/A
MAGGIC Y3 [134]	-0.312, 0.151	-0.260, 0.018	N/A	-0.338, 0.094	N/A	-0.307, 0.048	-0.193, 0.018	N/A
SHFM Y1 [40]	-0.329, 0.011	-0.351, <0.001	-0.377, 0.784	-0.355, <0.001	-0.381, <0.001	-0.373, 0.201	-0.284, <0.001	-0.283, 0.784
SHFM Y3 [40]	-0.329, 0.012	-0.350, <0.001	-0.381, 0.878	-0.355, <0.001	-0.378, <0.001	-0.371, 0.173	-0.283, <0.001	-0.287, 0.878
SHFM Y5 [40]	-0.330, 0.013	-0.365, <0.001	-0.386, 0.996	-0.356, <0.001	-0.379, <0.001	-0.377, 0.268	-0.298, <0.001	-0.292, 0.996

Table reports ΔAUC, p-value. N/A = score could not be calculated for the dataset; 30D = 30-day mortality; Y1 = 1 year mortality; Y3 = 3 year mortality; Y5 = 5 year mortality.

Table 5.8: Comparison of HF Risk Score Approaches

Score	**Method Used**	**# Features**	**Hemo?**	**Allows Missing Data**
CARNA Hemo	MVDD	28	**Yes**	**Yes**
CARNA All Fts	MVDD	66	**Yes**	**Yes**
EFFECT [131]	Logistic Regression	11	No	No
GWTG [133]	Logistic Regression	7	No	No
MAGGIC [134]	Poisson Regression	13	No	No
ESCAPE [132]	CPH	8	No	No
SHFM [40]	CPH	30	No	No
ADHERE [130]	Decision Tree (CART)	3	No	No
MARKER [136]	Decision Tree (BDT)	8	No	No
TOPCAT [137]	Various ML	86	No	No

Hemo = Invasive Hemodynamics; All Fts = All Features; CPH = Cox Proportional Hazards; CART = Classification and Regression Tree; BDT = Boosted Decision Tree

5.5 Related Work

HF Risk Scores. There are a variety of HF risk scores that provide risk stratifications in HF populations using statistical and machine learning models; a comparison is available in Table 5.8. The EFFECT [131], GWTG [133] and MAGGIC [134] risk scores predict risk of mortality in HF patients using various regression methods. The ESCAPE Risk Model and Discharge Score [132] and SHFM [40] stratify mortality risk using Cox Proportional Hazards models (CPH). The ESCAPE score was derived using the same dataset that we use for our training cohort. Finally, in the TOPCAT [137], ADHERE [130], and MARKER [136] risk models, machine learning algorithms, including decision trees, boosted decision trees, support vector machines and random forests are used to predict risk of mortality.

Some of these risk models use small, selective feature sets, or only stratify risk into a small number of groups (e.g., only two groups of high and low risk as in MARKER), and none of them incorporate invasive hemodynamics. Moreover, these methods suffer from limitations associated with statistical and naive machine learning models, such as being prone to bias, and lacking mechanisms to handle missing data [41, 42]. In fact, in external validation of these scores, a common issue cited is that some variables are not readily available in routine clinical practice or are missing from collected data cohorts so the score cannot be computed [43].

CARNA uses a larger, more diverse feature set than most scores, is able to provide more fine-grained risk stratification, i.e., can have more risk groups, and incorporates invasive hemodynamics. In addition, our model is explainable and can handle missing data. Ultimately, it is our intention that our risk score would be complementary to previous risk methodologies, in which our score is used to provision risk stratification for advanced HF patients requiring invasive hemodynamic monitoring, and others may be used to gain an understanding of risk for more general HF patients.

5.6 Summary and Discussion

In this chapter, we developed an explainable ML approach using Multi-Valued Decision Diagrams to derive and validate a novel HF risk score that incorporates invasive hemodynamic and other clinical variables to stratify risk of adverse outcomes in advanced HF patients. The CARNA risk scores were highly predictive of adverse outcomes in a broad spectrum of HF patients. Accurately identifying high-risk advanced

HF patients early on is fundamental for timely allocation of life-saving therapies and improvement of patient outcomes. CARNA can handle trajectories that are missing, variable and conflicting, directly addressing the model and data challenges elucidated in Chapter 1.

Model Design Choices and Limitations. Our models use single point-of-care measurements, and do not take advantage of multiple follow-up recordings. As a result, they may lose interrelations available from multiple temporal recordings (i.e., changes between measurements). However, using single measurements in our models allows for clinician ease-of-use. Furthermore, although only the "OR" nodes in the MVDD model explicitly handle missing data, we chose to use "AND/OR" MVDDs because the "OR"-only MVDDs become very large and overfit the data. We used single MVDD models for interpretability purposes throughout the project evaluation. However, ensemble approaches (e.g., ensembles of MVDDs) have been shown to outperform single model methods [138], and this will be investigated in future work. Additionally, we note that an aspect of model interpretability may be lost due to the model predicting risk classes generated from an unsupervised clustering method as opposed to predicting the binary outcome(s) directly. Even so, we believe such a tradeoff may be acceptable due to the improved ability to risk-stratify HF patients.

Despite fewer patients and shorter follow-up time (6-months) compared to other datasets, the ESCAPE trial was selected for model training because, to the best of the authors' knowledge, it is the only cohort available with detailed invasive hemodynamics derived from a well-designed randomized HF clinical trial. There is potential for selection bias by choosing trial data and higher-risk patients in the two UVA cohorts. In addition, many of our validation datasets did not have invasive hemodynamics so we were unable to validate the invasive hemodynamic models on all four of the patient cohorts. Further, there were heterogeneities in HF acuity status in the datasets used. Even so, validation of the CARNA models yielded robust risk stratification compared to other conventional HF risk score and ML models.

CARNA Outperforms Benchmarks. As shown in Tables 5.3–5.7, the CARNA risk scores highly outperform previous risk scores across all datasets, feature sets and outcomes. The CARNA Invasive Hemodynamics score was more predictive than other scores including the ESCAPE risk score which was derived on the same cohort as our training data using linear statistical methods. The CARNA All Features score also outperformed previous risk scores, indicating noninvasive hemodynamics are also predictive of outcomes. Moreover, as evidenced by Tables 5.4 and 5.5, the CARNA models outperform traditional ML models across all datasets, feature sets and outcomes.

Comment on Hemodynamics. The CARNA Invasive Hemodynamic models do better than the CARNA All Features models, which suggests that invasive hemodynamics (along with integrated metrices) improve outcome prediction for advanced HF patients. Integrated hemodynamic indices such as Cardiac Power Index, Mean Arterial Pressure, and Pulmonary Artery Pulsatility Index were highly predictive of patient outcomes. This aligns with findings from previous studies, demonstrating the incremental utility of integrated metrics in risk assessment [38, 122, 123, 139].

Study Strengths. We speculate our models outperform all previous approaches due to a combination of three key reasons: First, our models use a richer, more diverse feature set, beyond what is used in other clinical risk scores, and use integrated hemodynamic metrics, which are very sensitive to hemodynamic changes. Most other models look at isolated metrics, which may not be as predictive as the integrated ones.

In our models, integrated hemodynamic indices such as CPI, MAP, and Pulmonary Artery Pulsatility Index were highly predictive of patient outcomes. This aligns with other studies, which have shown the incremental utility of integrated metrics in risk assessment [38, 122, 123, 139].

Second, our models are able to handle missing data. Even if the datasets do not have high amounts of missingness, this is an important model advantage compared to other models that do not have built-in mechanisms to handle missing data, and instead must impute missing values. Previous studies such as [140, 141] have found that, even with small amounts of missingness (e.g., 10% missing data or less) the type of imputation method used can have a big impact (bias) on model performance; and that models that use complete data or do not use data points with missing features outperform imputed datasets. As such, we believe our models' built-in method to handle missing data may provide an important performance advantage.

Finally, our models are interpretable and provide clear sets of features and thresholds used to make each risk prediction. Elucidation of these phenotypes used to make risk characterizations by our models allow clinicians to better understand how and why a risk score was given. Such phenotypes may identify possible HF subgroups that can be further investigated in clinical studies.

A New Paradigm for Risk Stratification. This study introduces a new paradigm for HF risk stratification, in which predicting risk categories is used over singular binary events. We believe prediction of which patients fall into groups categorized by escalating ranges of event rates is very clinically relevant since many clinical management decisions are based on general categories of event rates/risk. Risk ascertainment in advanced heart failure patients is often challenging and more nuanced, requiring careful consideration of the competing risks of the need for advanced HF therapies (LVAD and heart transplant) against the "conditional risk category" of a given patient. We believe this approach may also facilitate complementary evidence-based modeling of "risk - benefit" trade-offs when it comes to the challenging shared decision discussions between clinicians and patients concerning HF prognostication and the timing of advanced heart failure therapies.

Chapter 6

Conclusion

Targeting fundamental challenges in developing trustworthy CDSS for medical trajectories, in this dissertation we develop robust, explainable and privacy-preserving machine learning frameworks. GlucoSynth is a novel privacy-preserving GAN framework to generate high-quality, private univariate time series data. GlucoSynth is the first framework to successfully generate synthetic glucose traces usable in real clinical applications and with strong privacy guarantees. DP-RuL is a locally differentially-private framework to learn population rulesets with high coverage and clinical utility for logic-based CDSS. This is the first LDP framework to solve the rule discovery learning problem. CARNA is a novel general-purpose explainable risk stratification and phenotyping machine learning framework, applied specifically for risk stratifying trajectories from advanced heart failure patients. CARNA develops a new clinical risk paradigm through its use of predicting *risk categories*, and is the first interpretable ML framework to incorporate invasive hemodynamics.

The research in this thesis is a successful fusion of important aspects of machine learning, XAI, privacy, and clinical application areas including diabetes and heart failure. The primary motivation for this work is to solve challenges within clinical applications of CDSS. At the same time, we advance state-of-the-art theory and algorithms in machine learning and privacy methodologies. GlucoSynth and DP-RuL promote sharing and aggregation of medical trajectories collected from personal wearable devices with reduced legal and privacy concerns. CARNA facilitates better understanding of underlying disease mechanisms for heart failure and other chronic diseases.

Societal Implications. This dissertation produced extensible privacy-preserving and ML frameworks and toolkits usable by application developers, decision support developers, data scientists, clinical researchers, privacy researchers, general users, patients and clinicians. GlucoSynth and DP-RuL address the need for effective privacy-preserving learning methodologies for CDSS to protect patient health data and develop concrete solutions for major gaps in the current state-of-the-art for private rule-based learning and univariate time series data synthesis. CARNA addresses shortcomings in robust, interpretable clinical machine learning and clinical risk stratification. These frameworks can help decrease instances of unsanctioned use and compromise of patient data, and increase patient trust and utilization of trajectory-based CDSS. Furthermore, the frameworks introduced in this dissertation can be applied to areas beyond healthcare, including fraud detection, network security monitoring and other decision support application areas such as behavioral health, business analytics, finance, sports prediction, and injury prevention. Moreover, this dissertation can help address current issues surrounding inadequate privacy protections and lacking user trust in many industries, including social media, news outlets, advertising, and the development of cyber-physical systems.

Bibliography

[1] Beau Norgeot, Benjamin S Glicksberg, and Atul J Butte. “A call for deep-learning healthcare”. In: *Nature medicine* 25.1 (2019), pp. 14–15.

[2] James E. Tcheng, Suzanne Bakken, David W. Bates, Hugh Bonner III, Tejal K. Gandhi, Meredith Josephs, Kensaku Kawamoto, Edwin A. Lomotan, Erin Mackay, Blackford Middleton, Jonathan M. Teich, Scott Weingarten, and Marianne Hamilton Lopez. “Optimizing Strategies for Clinical Decision Support: Summary of a Meeting Series”. In: *National Academy of Medicine* (2017).

[3] Christopher J Kelly, Alan Karthikesalingam, Mustafa Suleyman, Greg Corrado, and Dominic King. “Key challenges for delivering clinical impact with artificial intelligence”. In: *BMC medicine* 17 (2019), pp. 1–9.

[4] Halis Kaan Akturk, Robert Dowd, Kaushik Shankar, and Mark Derdzinski. “Real-world evidence and glycemic improvement using Dexcom G6 features”. In: *Diabetes Technology & Therapeutics* 23.S1 (2021), S–21.

[5] Reed T Sutton, David Pincock, Daniel C Baumgart, Daniel C Sadowski, Richard N Fedorak, and Karen I Kroeker. “An overview of clinical decision support systems: benefits, risks, and strategies for success”. In: *NPJ digital medicine* 3.1 (2020), p. 17.

[6] Mark A. Musen, Blackford Middleton, and Robert A. Greenes. “Clinical Decision-Support Systems”. In: *Biomedical Informatics: Computer Applications in Health Care and Biomedicine*. Ed. by Edward H. Shortliffe and James J. Cimino. Cham: Springer International Publishing, 2021, pp. 795–840. ISBN: 978-3-030-58721-5. DOI: 10.1007/978-3-030-58721-5_24. URL: https://doi.org/10.1007/978-3-030-58721-5_24.

[7] Yuan Lu, Edward R Melnick, and Harlan M Krumholz. “Clinical decision support in cardiovascular medicine”. In: *bmj* 377 (2022).

[8] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. “Deep learning for healthcare: review, opportunities and challenges”. In: *Briefings in bioinformatics* 19.6 (2018), pp. 1236–1246.

[9] Marzyeh Ghassemi, Tristan Naumann, Peter Schulam, Andrew L Beam, Irene Y Chen, and Rajesh Ranganath. “A review of challenges and opportunities in machine learning for health”. In: *AMIA Summits on Translational Science Proceedings* 2020 (2020), p. 191.

[10] Rahul C Deo. “Machine learning in medicine”. In: *Circulation* 132.20 (2015), pp. 1920–1930.

[11] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “"Why should I trust you?" Explaining the predictions of any classifier”. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, pp. 1135–1144.

[12] Andreas Holzinger, Chris Biemann, Constantinos S Pattichis, and Douglas B Kell. “What do we need to build explainable AI systems for the medical domain?” In: *arXiv preprint arXiv:1712.09923* (2017).

[13] Changhee Lee and Mihaela Van Der Schaar. “Temporal phenotyping using deep predictive clustering of disease progression”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 5767–5777.

[14] Ilija Šimić, Vedran Sabol, and Eduardo Veas. “XAI Methods for Neural Time Series Classification: A Brief Review”. In: *arXiv preprint arXiv:2108.08009* (2021).

[15] W. Nicholson Price and I. Glenn Cohen. “Privacy in the age of medical big data”. In: *Nature Medicine* 25.1 (2019). Available: `http://dx.doi.org/10.1038/s41591-018-0272-7`, pp. 37–43. ISSN: 1546170X. DOI: `10.1038/s41591-018-0272-7`.

[16] Jong Wook Kim, Beakcheol Jang, and Hoon Yoo. “Privacy-preserving aggregation of personal health data streams”. In: *PLoS ONE* 13.11 (2018), pp. 1–15. ISSN: 19326203. DOI: `10.1371/journal.pone.0207639`.

[17] Katherine E Britton and Jennifer D Britton-Colonnese. “Privacy and security issues surrounding the protection of data generated by continuous glucose monitors”. In: *Journal of diabetes science and technology* 11.2 (2017), pp. 216–219.

[18] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. “Time-series generative adversarial networks”. In: *Advances in neural information processing systems* 32 (2019).

[19] Xiaomin Li, Vangelis Metsis, Huangyingrui Wang, and Anne Hee Hiong Ngu. “TTS-GAN: A Transformer-Based Time-Series Generative Adversarial Network”. In: *Artificial Intelligence in Medicine*. Ed. by Martin Michalowski, Syed Sibte Raza Abidi, and Samina Abidi. Cham: Springer International Publishing, 2022, pp. 133–143.

[20] Amirsina Torfi, Edward A Fox, and Chandan K Reddy. “Differentially private synthetic medical data generation using convolutional gans”. In: *Information Sciences* 586 (2022), pp. 485–500.

[21] Cynthia Dwork. “Differential privacy: A survey of results”. In: *International conference on theory and applications of models of computation*. Springer. 2008, pp. 1–19.

[22] Clive WJ Granger. “Investigating causal relations by econometric models and cross-spectral methods”. In: *Econometrica: journal of the Econometric Society* (1969), pp. 424–438.

[23] Jahanzaib Latif, Chuangbai Xiao, Shanshan Tu, Sadaqat Ur Rehman, Azhar Imran, and Anas Bilal. “Implementation and use of disease diagnosis systems for electronic medical records based on machine learning: A complete review”. In: *IEEE Access* 8 (2020), pp. 150489–150513.

[24] David S Watson, Jenny Krutzinna, Ian N Bruce, Christopher EM Griffiths, Iain B McInnes, Michael R Barnes, and Luciano Floridi. “Clinical applications of machine learning algorithms: beyond the black box”. In: *Bmj* 364 (2019), p. l886.

[25] Cosima Gretton. “Trust and Transparency in Machine Learning-Based Clinical Decision Support”. In: *Human and Machine Learning*. Springer, 2018, pp. 279–292.

[26] Lucas Lange, Tobias Schreieder, Victor Christen, and Erhard Rahm. “Privacy at Risk: Exploiting Similarities in Health Data for Identity Inference”. In: *arXiv preprint arXiv:2308.08310* (2023).

[27] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. “What can we learn privately?” In: *SIAM Journal on Computing* 40.3 (2011), pp. 793–826.

[28] Giulia Fanti, Vasyl Pihur, and Úlfar Erlingsson. “Building a RAPPOR with the Unknown: Privacy-Preserving Learning of Associations and Data Dictionaries”. In: 2016.3 (2015). Available: `http://arxiv.org/abs/1503.01214`, pp. 41–61. DOI: `10.1515/popets-2016-0015`. arXiv: `1503.01214`.

[29] Sungwook Kim, Hyejin Shin, Chunghun Baek, Soohyung Kim, and Junbum Shin. “Learning new words from keystroke data with local differential privacy”. In: *IEEE Transactions on Knowledge and Data Engineering* 32.3 (2018), pp. 479–491.

[30] Ning Wang, Xiaokui Xiao, Yin Yang, Ta Duy Hoang, Hyejin Shin, Junbum Shin, and Ge Yu. “PrivTrie: Effective frequent term discovery under local differential privacy”. In: *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE. 2018, pp. 821–832.

[31] Yansheng Wang, Yongxin Tong, and Dingyuan Shi. “Federated latent dirichlet allocation: A local differential privacy based framework”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 6283–6290.

[32] Tianhao Wang, Ninghui Li, and Somesh Jha. “Locally differentially private frequent itemset mining”. In: *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2018, pp. 127–143.

[33] Tianhao Wang, Ninghui Li, and Somesh Jha. “Locally differentially private heavy hitter identification”. In: *IEEE Transactions on Dependable and Secure Computing* (2019).

[34] Jinyuan Jia and Neil Zhenqiang Gong. “Calibrate: Frequency estimation and heavy hitter identification with local differential privacy via incorporating prior knowledge”. In: *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE. 2019, pp. 2008–2016.

[35] Sam Fletcher and Md Zahidul Islam. “Decision tree classification with differential privacy: A survey”. In: *ACM Computing Surveys (CSUR)* 52.4 (2019), pp. 1–33.

[36] James Dom Dera. “Risk stratification: a two-step process for identifying your sickest patients”. In: *Family practice management* 26.3 (2019), pp. 21–26.

[37] Larry A Allen, Lynne W Stevenson, Kathleen L Grady, Nathan E Goldstein, Daniel D Matlock, Robert M Arnold, Nancy R Cook, G Michael Felker, Gary S Francis, Paul J Hauptman, et al. “Decision making in advanced heart failure: a scientific statement from the American Heart Association”. In: *Circulation* 125.15 (2012), pp. 1928–1952.

[38] Kenneth C Bilchick, Eliany Mejia-Lopez, Peter McCullough, Khadijah Breathett, Jamie L Kennedy, Jose Tallaj, James Bergin, Salpy Pamboukian, Mohammad Abuannadi, and Sula Mazimba. "Clinical impact of changes in hemodynamic indices of contractile function during treatment of acute decompensated heart failure". In: *Journal of cardiac failure* 24.1 (2018), pp. 43–50.

[39] Barry A Borlaug and David A Kass. "Invasive hemodynamic assessment in heart failure". In: *Cardiology clinics* 29.2 (2011), pp. 269–280.

[40] Wayne C Levy, Dariush Mozaffarian, David T Linker, Santosh C Sutradhar, Stefan D Anker, Anne B Cropp, Inder Anand, Aldo Maggioni, Paul Burton, Mark D Sullivan, et al. "The Seattle Heart Failure Model: prediction of survival in heart failure". In: *Circulation* 113.11 (2006), pp. 1424–1433.

[41] Gian Luca Di Tanna, Heidi Wirtz, Karen L Burrows, and Gary Globe. "Evaluating risk prediction models for adults with heart failure: A systematic literature review". In: *PLoS One* 15.1 (2020), e0224135.

[42] Marco Canepa, Candida Fonseca, Ovidiu Chioncel, Cécile Laroche, Maria G Crespo-Leiro, Andrew JS Coats, Alexandre Mebazaa, Massimo F Piepoli, Luigi Tavazzi, Aldo P Maggioni, et al. "Performance of prognostic risk scores in chronic heart failure patients enrolled in the European Society of Cardiology Heart Failure Long-Term Registry". In: *JACC: Heart Failure* 6.6 (2018), pp. 452–462.

[43] Pau Codina, Josep Lupón, Andrea Borrellas, Giosafat Spitaleri, Germán Cediel, Mar Domingo, Joanne Simpson, Wayne C Levy, Evelyn Santiago-Vacas, Elisabet Zamora, et al. "Head-to-head comparison of contemporary heart failure risk scores". In: *European Journal of Heart Failure* 23.12 (2021), pp. 2035–2044.

[44] Barry Greenberg, Alison Brann, Claudio Campagnari, Eric Adler, and Avi Yagil. "Machine Learning Applications in Heart Failure Disease Management: Hype or Hope?" In: *Current Treatment Options in Cardiovascular Medicine* 23.6 (2021), p. 35.

[45] Dineo Mpanya, Turgay Celik, Eric Klug, and Hopewell Ntsinjana. "Predicting mortality and hospitalization in heart failure using machine learning: A systematic literature review". In: *IJC Heart & Vasculature* 34 (2021), p. 100773.

[46] Arvind Srinivasan, Timothy Ham, Sharad Malik, and Robert K Brayton. "Algorithms for discrete function manipulation". In: *1990 IEEE international conference on computer-aided design*. IEEE Computer Society. 1990, pp. 92–93.

[47] Cynthia Dwork and Aaron Roth. "The Algorithmic Foundations of Differential Privacy". In: *Foundations and Trends® in Theoretical Computer Science* 9.3-4 (2014), pp. 211–407. ISSN: 1551-305X. DOI: 10.1561/0400000042.

[48] ADP Team et al. "Learning with privacy at scale". In: *Apple Machine Learning Journal* 1.8 (2017).

[49] Abhradeep Guha Thakurta, Andrew H Vyrros, Umesh S Vaishampayan, Gaurav Kapoor, Julien Freudinger, Vipul Ved Prakash, Arnaud Legendre, and Steven Duplinsky. *Emoji frequency detection and deep link frequency*. US Patent 9,705,908. 2017.

[50] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. "Rappor: Randomized aggregatable privacy-preserving ordinal response". In: *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. 2014, pp. 1054–1067.

[51] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and Xiaofeng Wang. “Privacy loss in apple’s implementation of differential privacy on macos 10.12”. In: *arXiv preprint arXiv:1709.02753* (2017).

[52] WeiKang Liu, Yanchun Zhang, Hong Yang, and Qinxue Meng. “A Survey on Differential Privacy for Medical Data Analysis”. In: *Annals of Data Science* (2023), pp. 1–15.

[53] Joseph Ficek, Wei Wang, Henian Chen, Getachew Dagne, and Ellen Daley. “Differential privacy in health research: A scoping review”. In: *Journal of the American Medical Informatics Association* 28.10 (2021), pp. 2269–2276.

[54] Samuel Maddock, Graham Cormode, Tianhao Wang, Carsten Maple, and Somesh Jha. “Federated boosted decision trees with differential privacy”. In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 2022, pp. 2249–2263.

[55] Jinyan Wang, Zhou Tan, Xianxian Li, Yuhang Hu, et al. “Differential privacy preservation in interpretable feedforward-designed convolutional neural networks”. In: *2020 IEEE 19th international conference on trust, security and privacy in computing and communications (TrustCom)*. IEEE. 2020, pp. 631–638.

[56] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI”. In: *Information fusion* 58 (2020), pp. 82–115.

[57] Subrato Bharati, M Rubaiyat Hossain Mondal, and Prajoy Podder. “A Review on Explainable Artificial Intelligence for Healthcare: Why, How, and When?” In: *IEEE Transactions on Artificial Intelligence* (2023).

[58] Shishir Rao, Yikuan Li, Rema Ramakrishnan, Abdelaali Hassaine, Dexter Canoy, John Cleland, Thomas Lukasiewicz, Gholamreza Salimi-Khorshidi, and Kazem Rahimi. “An explainable Transformer-based deep learning model for the prediction of incident heart failure”. In: *IEEE Journal of Biomedical and Health Informatics* 26.7 (2022), pp. 3362–3372.

[59] Ahmad Chaddad, Jihao Peng, Jian Xu, and Ahmed Bouridane. “Survey of Explainable AI Techniques in Healthcare”. In: *Sensors* 23.2 (2023), p. 634.

[60] Slawek Kierner, Jacek Kucharski, and Zofia Kierner. “Taxonomy of Hybrid Architectures involving Rule-Based Reasoning and Machine Learning in Clinical Decision Systems: A Scoping Review”. In: *Journal of Biomedical Informatics* (2023), p. 104428.

[61] Geir Thore Berge, Ole-Christoffer Granmo, Tor Oddbjørn Tveit, Anna Linda Ruthjersen, and Jivitesh Sharma. “Combining unsupervised, supervised and rule-based learning: the case of detecting patient allergies in electronic health records”. In: *BMC Medical Informatics and Decision Making* 23.1 (2023), p. 188.

[62] Ali Ahmed, Wilbert S Aronow, and Jerome L Fleg. “Higher New York Heart Association classes and increased mortality and hospitalization in patients with heart failure and preserved left ventricular function”. In: *American heart journal* 151.2 (2006), pp. 444–450.

[63] Deborah Young-Hyman, Mary De Groot, Felicia Hill-Briggs, Jeffrey S Gonzalez, Korey Hood, and Mark Peyrot. "Psychosocial care for people with diabetes: a position statement of the American Diabetes Association". In: *Diabetes care* 39.12 (2016), pp. 2126–2140.

[64] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial networks". In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[65] Cynthia Binanay, Robert M Califf, Vic Hasselblad, Christopher M O'Connor, Monica R Shah, George Sopko, Lynne W Stevenson, Gary S Francis, Carl V Leier, Leslie W Miller, et al. "Evaluation study of congestive heart failure and pulmonary artery catheterization effectiveness: the ESCAPE trial." In: *Jama* 294.13 (2005), pp. 1625–1633.

[66] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control.* John Wiley & Sons, 2015.

[67] Lexiang Ye and Eamonn Keogh. "Time series shapelets: a new primitive for data mining". In: *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 2009.

[68] Helmut Lütkepohl. *New introduction to multiple time series analysis.* Springer Science & Business Media, 2005.

[69] Stephen A Billings. *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains.* John Wiley & Sons, 2013.

[70] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. "A review of recurrent neural networks: LSTM cells and network architectures". In: *Neural computation* 31.7 (2019), pp. 1235–1270.

[71] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. "Deep learning with differential privacy". In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security.* 2016, pp. 308–318.

[72] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. "Scalable private learning with pate". In: *arXiv preprint arXiv:1802.08908* (2018).

[73] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. "Calibrating Noise to Sensitivity in Private Data Analysis". In: *Theory of Cryptography Conference.* 2006.

[74] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* https://www.tensorflow.org/. 2015.

[75] Ali Borji. "Pros and cons of GAN evaluation measures: New developments". In: *Computer Vision and Image Understanding* 215 (2022), p. 103329.

[76] Ahmed Alaa, Alex James Chan, and Mihaela van der Schaar. "Generative time-series modeling with fourier flows". In: *International Conference on Learning Representations.* 2021.

[77] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using real nvp". In: *arXiv preprint arXiv:1605.08803* (2016).

[78] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. "Real-valued (medical) time series generation with recurrent conditional gans". In: *arXiv preprint arXiv:1706.02633* (2017).

[79] Lorenzo Frigerio, Anderson Santana de Oliveira, Laurent Gomez, and Patrick Duverger. “Differentially private generative adversarial networks for time series, continuous, and discrete open data”. In: *IFIP International Conference on ICT Systems Security and Privacy Protection*. Springer. 2019, pp. 151–164.

[80] Fred B Bryant and Paul R Yarnold. “Principal-components analysis and exploratory and confirmatory factor analysis.” In: *American Psychological Association* (1995).

[81] Taisa Kushner, Marc D Breton, and Sriram Sankaranarayanan. “Multi-hour blood glucose prediction in type 1 diabetes: A patient-specific approach using shallow neural network models”. In: *Diabetes Technology & Therapeutics* 22.12 (2020), pp. 883–891.

[82] William L Clarke. “The original Clarke error grid analysis (EGA)”. In: *Diabetes technology & therapeutics* 7.5 (2005), pp. 776–779.

[83] *Open Humans*. https://www.openhumans.org/. 2023.

[84] *T1D Exchange Registry*. https://t1dexchange.org/registry/. 2023.

[85] Juan Miguel Lopez Alcaraz and Nils Strodthoff. “Diffusion-based time series imputation and forecasting with structured state space models”. In: *arXiv preprint arXiv:2208.09399* (2022).

[86] Mihai Dogariu, Liviu-Daniel Ştefan, Bogdan Andrei Boteanu, Claudiu Lamba, Bomi Kim, and Bogdan Ionescu. “Generation of realistic synthetic financial time-series”. In: *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 18.4 (2022), pp. 1–27.

[87] Mina Razghandi, Hao Zhou, Melike Erol-Kantarci, and Damla Turgut. “Variational Autoencoder Generative Adversarial Network for Synthetic Data Generation in Smart Home”. In: *arXiv preprint arXiv:2201.07387* (2022).

[88] Debapriya Hazra and Yung-Cheol Byun. “SynSigGAN: Generative adversarial networks for synthetic biomedical signal generation”. In: *Biology* 9.12 (2020), p. 441.

[89] Eoin Brophy, Zhengwei Wang, Qi She, and Tomas Ward. “Generative adversarial networks in time series: A survey and taxonomy”. In: *arXiv preprint arXiv:2107.11098* (2021).

[90] Hao Ni, Lukasz Szpruch, Magnus Wiese, Shujian Liao, and Baoren Xiao. “Conditional sig-wasserstein gans for time series generation”. In: *arXiv preprint arXiv:2006.05421* (2020).

[91] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. “Differentially private generative adversarial network”. In: *arXiv preprint arXiv:1802.06739* (2018).

[92] James Jordon, Jinsung Yoon, and Mihaela Van Der Schaar. “PATE-GAN: Generating synthetic data with differential privacy guarantees”. In: *International conference on learning representations*. 2018.

[93] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. “Using GANs for sharing networked time series data: Challenges, initial promise, and open questions”. In: *Proceedings of the ACM Internet Measurement Conference*. 2020, pp. 464–483.

[94] Borja Martínez-Pérez, Isabel de la Torre-Díez, Miguel López-Coronado, Beatriz Sainz-De-Abajo, Montserrat Robles, and Juan Miguel García-Gómez. “Mobile clinical decision support systems and applications: a literature and commercial review”. In: *Journal of medical systems* 38.1 (2014), p. 4.

[95] Cynergistek Insights Center. “Measuring Progress: Expanding the Horizon: 2019 Annual Report.” In: (2019). Available: `https://insights.cynergistek.com/reports/2019-healthcare-cybersecurity-privacy-report?utm_source=press_release&utm_medium=pr&utm_campaign=&utm_content=2019_report`.

[96] Ezio Bartocci, Jyotirmoy Deshmukh, Alexandre Donzé, Georgios Fainekos, Oded Maler, Dejan Ničković, and Sriram Sankaranarayanan. “Specification-based monitoring of cyber-physical systems: a survey on theory, tools and applications”. In: *Lectures on Runtime Verification*. Springer, 2018, pp. 135–175.

[97] Josephine Lamp, Simone Silvetti, Marc Breton, Laura Nenzi, and Lu Feng. “A Logic-Based Learning Approach to Explore Diabetes Patient Behaviors”. In: *International Conference on Computational Methods in Systems Biology*. Springer. 2019, pp. 188–206.

[98] Ezio Bartocci, Cristinel Mateis, Eleonora Nesterini, and Dejan Nickovic. “Survey on mining signal temporal logic specifications”. In: *Information and Computation* (2022), p. 104957.

[99] Laura Nenzi, Simone Silvetti, Ezio Bartocci, and Luca Bortolussi. “A robust genetic algorithm for learning temporal specifications from data”. In: *International Conference on Quantitative Evaluation of Systems*. Springer. 2018, pp. 323–338.

[100] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. “A survey of monte carlo tree search methods”. In: *IEEE Transactions on Computational Intelligence and AI in games* 4.1 (2012), pp. 1–43.

[101] Levente Kocsis and Csaba Szepesvári. “Bandit based monte-carlo planning”. In: *European conference on machine learning*. Springer. 2006, pp. 282–293.

[102] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. “Locally differentially private protocols for frequency estimation”. In: *26th {USENIX} Security Symposium*. 2017, pp. 729–745.

[103] Travis J Moss, Matthew T Clark, James Forrest Calland, Kyle B Enfield, John D Voss, Douglas E Lake, and J Randall Moorman. “Cardiorespiratory dynamics measured from continuous ECG monitoring improves detection of deterioration in acute care patients: A retrospective cohort study”. In: *PloS one* 12.8 (2017).

[104] Matthew A Reyna, Chris Josef, Salman Seyedi, Russell Jeter, Supreeth P Shashikumar, M Brandon Westover, Ashish Sharma, Shamim Nemati, and Gari D Clifford. “Early prediction of sepsis from clinical data: the PhysioNet/Computing in Cardiology Challenge 2019”. In: *2019 Computing in Cardiology (CinC)*. IEEE. 2019, Page–1.

[105] Roy W Beck, William V Tamborlane, Richard M Bergenstal, Kellee M Miller, Stephanie N DuBose, Callyn A Hall, and T1D Exchange Clinic Network. “The T1D Exchange clinic registry”. In: *The Journal of Clinical Endocrinology & Metabolism* 97.12 (2012), pp. 4383–4389.

[106] Teng Wang, Xuefeng Zhang, Jingyu Feng, and Xinyu Yang. "A comprehensive survey on local differential privacy toward data statistics and analysis". In: *Sensors* 20.24 (2020), p. 7030.

[107] Xingxing Xiong, Shubo Liu, Dan Li, Zhaohui Cai, and Xiaoguang Niu. "A comprehensive survey on local differential privacy". In: *Security and Communication Networks* 2020 (2020), pp. 1–29.

[108] Xiangguo Zhao, Yanhui Li, Ye Yuan, Xin Bi, and Guoren Wang. "LDPart: effective location-record data publication via local differential privacy". In: *IEEE Access* 7 (2019), pp. 31435–31445.

[109] Qingqing Ye, Haibo Hu, Xiaofeng Meng, and Huadi Zheng. "PrivKV: Key-value data collection with local differential privacy". In: *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2019, pp. 317–331.

[110] Yan Yan, Xin Gao, Adnan Mahmood, Yang Zhang, Shuang Wang, and Quan Z Sheng. "An arithmetic differential privacy budget allocation method for the partitioning and publishing of location information". In: *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE. 2020, pp. 1395–1401.

[111] Justin Whitehouse, Aaditya Ramdas, Ryan Rogers, and Steven Wu. "Fully-adaptive composition in differential privacy". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 36990–37007.

[112] Kai Zhang, Jiao Tian, Hongwang Xiao, Ying Zhao, Wenyu Zhao, and Jinjun Chen. "A numerical splitting and adaptive privacy budget-allocation-based LDP mechanism for privacy preservation in blockchain-powered IoT". In: *IEEE Internet of Things Journal* 10.8 (2022), pp. 6733–6741.

[113] Qinbin Li, Zhaomin Wu, Zeyi Wen, and Bingsheng He. "Privacy-preserving gradient boosting decision trees". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 01. 2020, pp. 784–791.

[114] Shannon M Dunlay, Véronique L Roger, Jill M Killian, Susan A Weston, Philip J Schulte, Anna V Subramaniam, Saul B Blecker, and Margaret M Redfield. "Advanced heart failure epidemiology and outcomes: a population-based study". In: *Heart Failure* 9.10 (2021), pp. 722–732.

[115] Frederik H Verbrugge, Marco Guazzi, Jeffrey M Testani, and Barry A Borlaug. "Altered hemodynamics and end-organ damage in heart failure: impact on the lung and kidney". In: *Circulation* 142.10 (2020), pp. 998–1012.

[116] Steven Hsu, James C Fang, and Barry A Borlaug. "Hemodynamics for the heart failure clinician: a state-of-the-art review". In: *Journal of cardiac failure* 28.1 (2022), pp. 133–148.

[117] David Bergman, Andre A Cire, Willem-Jan Van Hoeve, and John Hooker. *Decision diagrams for optimization*. Vol. 1. Springer, 2016.

[118] Alexandre M Florio, Pedro Martins, Maximilian Schiffer, Thiago Serra, and Thibaut Vidal. "Optimal decision diagrams for classification". In: *arXiv preprint arXiv:2205.14500* (2022).

[119] Akshay S Desai and Lynne W Stevenson. "Rehospitalization for heart failure: predict or prevent?" In: *Circulation* 126.4 (2012), pp. 501–506.

[120] Beta-Blocker Evaluation of Survival Trial Investigators. "A trial of the beta-blocker bucindolol in patients with advanced chronic heart failure". In: *New England Journal of Medicine* 344.22 (2001), pp. 1659–1667.

[121] G Michael Felker, Tariq Ahmad, Kevin J Anstrom, Kirkwood F Adams, Lawton S Cooper, Justin A Ezekowitz, Mona Fiuzat, Nancy Houston-Miller, James L Januzzi, Eric S Leifer, et al. “Rationale and design of the GUIDE-IT study: guiding evidence based therapy using biomarker intensified treatment in heart failure”. In: *JACC: Heart Failure* 2.5 (2014), pp. 457–465.

[122] Kenneth C Bilchick, Nathaniel Chishinga, Alex M Parker, David X Zhuo, Mitchell H Rosner, LaVone A Smith, Hunter Mwansa, Jacob N Blackwell, Peter A McCullough, and Sula Mazimba. “Plasma volume and renal function predict six-month survival after hospitalization for acute decompensated heart failure”. In: *Cardiorenal medicine* 8.1 (2018), pp. 61–70.

[123] Sula Mazimba, Jamie LW Kennedy, David Zhuo, James Bergin, Mohammad Abuannadi, Jose Tallaj, and Kenneth C Bilchick. “Decreased pulmonary arterial proportional pulse pressure after pulmonary artery catheter optimization for advanced heart failure is associated with adverse clinical outcomes”. In: *Journal of cardiac failure* 22.12 (2016), pp. 954–961.

[124] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* (2011).

[125] Robert Tibshirani, Guenther Walther, and Trevor Hastie. “Estimating the number of clusters in a data set via the gap statistic”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.2 (2001), pp. 411–423.

[126] Lawrence J Hubert and Joel R Levin. “A general statistical framework for assessing categorical clustering in free recall.” In: *Psychological bulletin* 83.6 (1976), p. 1072.

[127] Pypi. *C-index*. URL: https://pypi.org/project/c-index/#description (visited on 2022).

[128] J. Ross Quinlan. “Induction of decision trees”. In: *Machine learning* 1 (1986), pp. 81–106.

[129] Maarten JG Leening, Moniek M Vedder, Jacqueline CM Witteman, Michael J Pencina, and Ewout W Steyerberg. “Net reclassification improvement: computation, interpretation, and controversies: a literature review and clinician’s guide”. In: *Annals of internal medicine* 160.2 (2014), pp. 122–131.

[130] Gregg C Fonarow, Kirkwood F Adams, William T Abraham, Clyde W Yancy, W John Boscardin, ADHERE Scientific Advisory Committee, et al. “Risk stratification for in-hospital mortality in acutely decompensated heart failure: classification and regression tree analysis”. In: *Jama* 293.5 (2005), pp. 572–580.

[131] Douglas S Lee, Peter C Austin, Jean L Rouleau, Peter P Liu, David Naimark, and Jack V Tu. “Predicting mortality among patients hospitalized for heart failure: derivation and validation of a clinical model”. In: *Jama* 290.19 (2003), pp. 2581–2587.

[132] Christopher M O’Connor, Vic Hasselblad, Rajendra H Mehta, Gudaye Tasissa, Robert M Califf, Mona Fiuzat, Joseph G Rogers, Carl V Leier, and Lynne W Stevenson. “Triage after hospitalization with advanced heart failure: the ESCAPE (Evaluation Study of Congestive Heart Failure and Pulmonary Artery Catheterization Effectiveness) risk model and discharge score”. In: *Journal of the American College of Cardiology* 55.9 (2010), pp. 872–878.

[133] Pamela N Peterson, John S Rumsfeld, Li Liang, Nancy M Albert, Adrian F Hernandez, Eric D Peterson, Gregg C Fonarow, and Frederick A Masoudi. “A validated risk score for in-hospital mortality in patients with heart failure from the American Heart Association get with the guidelines program”. In: *Circulation: Cardiovascular Quality and Outcomes* 3.1 (2010), pp. 25–32.

[134] Stuart J Pocock, Cono A Ariti, John JV McMurray, Aldo Maggioni, Lars Køber, Iain B Squire, Karl Swedberg, Joanna Dobson, Katrina K Poppe, Gillian A Whalley, et al. “Predicting survival in heart failure: a risk score based on 39 372 patients from 30 studies”. In: *European heart journal* 34.19 (2013), pp. 1404–1413.

[135] Elizabeth R DeLong, David M DeLong, and Daniel L Clarke-Pearson. “Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach”. In: *Biometrics* (1988), pp. 837–845.

[136] Eric D Adler, Adriaan A Voors, Liviu Klein, Fima Macheret, Oscar O Braun, Marcus A Urey, Wenhong Zhu, Iziah Sama, Matevz Tadel, Claudio Campagnari, et al. “Improving risk prediction in heart failure using machine learning”. In: *European journal of heart failure* 22.1 (2020), pp. 139–147.

[137] Suveen Angraal, Bobak J Mortazavi, Aakriti Gupta, Rohan Khera, Tariq Ahmad, Nihar R Desai, Daniel L Jacoby, Frederick A Masoudi, John A Spertus, and Harlan M Krumholz. “Machine learning prediction of mortality and hospitalization in heart failure with preserved ejection fraction”. In: *JACC: Heart Failure* 8.1 (2020), pp. 12–21.

[138] Mohamed Hosni, Juan M Carrillo de Gea, Ali Idri, Manal El Bajta, José Luis Fernández Alemán, Ginés García-Mateos, and Ibtissam Abnane. “A systematic mapping study for ensemble classification methods in cardiovascular disease”. In: *Artificial Intelligence Review* 54 (2021), pp. 2827–2861.

[139] Sula Mazimba, Hunter Mwansa, Khadijah Breathett, Jarred E Strickling, Kajal Shah, Coleen McNamara, Nishaki Mehta, Younghoon Kwon, Josephine Lamp, Lu Feng, et al. “Systemic arterial pulsatility index (SAPi) predicts adverse outcomes in advanced heart failure patients”. In: *Heart and Vessels* 37.10 (2022), pp. 1719–1727.

[140] Tolou Shadbahr, Michael Roberts, Jan Stanczuk, Julian Gilbey, Philip Teare, Sören Dittmer, Matthew Thorpe, Ramon Viñas Torné, Evis Sala, Pietro Lió, et al. “The impact of imputation quality on machine learning classifiers for datasets with missing values”. In: *Communications Medicine* 3.1 (2023), p. 139.

[141] Alireza Farhangfar, Lukasz Kurgan, and Jennifer Dy. “Impact of imputation of missing values on classification error for discrete data”. In: *Pattern Recognition* 41.12 (2008), pp. 3692–3705.