

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
UNIVERSITY OF ADRAR
FACULTY OF MATTER SCIENCES, MATHEMATICS AND COMPUTER SCIENCE
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE



DOCTORAL THESIS

Submitted in partial fulfillment of the requirements for the degree of Doctor

In Computer Science

Specialization: Artificial Intelligence

Presented by:

BOUDAUD Arbia

THESIS TITLE

=====

Anonymisation des données par apprentissage profonde
(Data anonymization using deep learning)

=====

Publicly defended on June 23, 2026, before the examination committee:

Name	Academic Rank	Institution	Role
Mr. KADDI Mohammed	Professor	University of Adrar, Algeria	Chair
Mr. KABOU Salheddine	Professor	ENS - Bechar, Algeria	Supervisor
Mr. RABHI Seddik	Associate Professor	University of Adrar, Algeria	Examiner
Mr. MAMOUNI El Mamoun	Associate Professor	University of Adrar, Algeria	Examiner
Mr. KHOBZAOUI Abdelkader	Associate Professor	University of Sidi Bel Abbes	Examiner

Academic Year: 2025 – 2026

Contents

List of Figures	iv
List of Tables	vi
1 Classical Anonymization Methods	5
1.1 Introduction	5
1.2 Statistical Disclosure Control methods for microdata protection	5
1.3 Non-perturbative methods	7
1.3.1 k-anonymity	7
1.3.2 ℓ -diversity	8
1.3.3 t -closeness	9
1.4 Perturbative methods	10
1.4.1 Swapping	10
1.4.2 Re-sampling	10
1.4.3 Noise addition	11
1.4.4 Advantages	12
1.4.5 Challenges	12
1.5 Conclusion	13
2 Deep Learning: Fundamentals and Architectures	14
2.1 Introduction	14

2.2	Fundamentals of Deep Learning (models, optimization, loss)	15
2.2.1	Supervised and unsupervised learning	15
2.2.2	Neural networks as function approximators	15
2.2.3	Building blocks: layers, parameters, activation functions	16
2.2.4	Optimization objective: empirical risk minimization	16
2.2.5	Gradient-based learning: backpropagation and stochastic optimization	17
2.3	Core architectures (MLP, CNN, RNN, GANs)	18
2.3.1	Multilayer Perceptrons (MLPs)	18
2.3.2	Convolutional Neural Network Architecture	19
2.3.3	Recurrent Neural Networks	21
2.3.4	Generative Adversarial Networks (GANs)	27
2.3.5	Attention mechanism and Transformers	29
2.4	Conclusion	30
3	Generative Models and Probabilistic Sampling	31
3.1	Introduction	31
3.2	Generative Models	32
3.2.1	GANs-based methods	32
3.2.2	Markovian-based methods	39
3.3	Probabilistic Sampling	42
3.3.1	Metropolis–Hastings Sampling	42
3.4	Conclusion	43
4	Towards a Physiologically-Informed GAN	44
4.1	Introduction	44
4.2	Physiologically-Informed GAN Framework	44
4.2.1	Step 1:DataProcessor	45
4.2.2	Step 2:Differentiable Hidden Markov Model	46

4.2.3	Step 3: The Generator	51
4.2.4	Step 4: The Discriminator	58
4.2.5	Step 5: WGAN-GP Trainer	59
4.2.6	Step 6: Clinical Validation Step	61
4.3	Conclusion	62
5	Implementation and Evaluation	63
5.1	Introduction	63
5.2	Dataset Description	64
5.3	Evaluation Metrics	66
5.3.1	Fidelity	66
5.3.2	Utility	66
5.3.3	Privacy	66
5.4	Results	67
5.4.1	Fidelity Evaluation	67
5.4.2	Utility Evaluation	67
5.4.3	Privacy Evaluation	68
5.4.4	$t - SNE$ Visualization	69
5.4.5	Feature-Level Distribution Comparison	71
5.5	Ablation Study Results	71
5.5.1	Ablation Configurations	71
5.5.2	Ablation Results	72
5.6	Discussion of Results	74
5.7	Conclusion	74
	Bibliography	78

List of Figures

2.1	Recurrent Neural Network layer	21
2.2	Simple recurrent neural network illustrated as a feedforward network	22
2.3	Recurrent neural network illustrated as a forward-backward network	23
2.4	Different types of recurrent neural networks	24
2.5	GAN architecture and training loop	28
2.6	Saddle-point optimisation for GAN training	29
3.1	Tabular EHR	33
3.2	Time series EHR	35
4.1	Outline of the proposed approach	45
4.2	Data Processor Step	46
4.3	Differentiable Hidden Markov Model	46
4.4	Patient History Tracking	47
4.5	State Transition Probabilities	49
4.6	State Emission Probabilities	50
4.7	Generator	51
4.8	Metropolis–Hastings Sampling	53
4.9	Metropolis–Hastings stages	53
4.10	Discriminator Step	59
4.11	Clinical Validation	61

5.1	Comparison of t-SNE embeddings for HMM-Gan against (a) TimEHR and (b) TimeGAN.	69
5.2	Feature-level distribution comparison. (a) HMM-Gan vs TimEHR. (b) HMM-Gan vs TimeGAN.	70
5.3	t-SNE Embeddings of Real vs. Synthetic Data for Ablation Variants.	72

List of Tables

1.1	Comparison of original microdata vs. 4-anonymous generalization.	8
1.2	Comparison of data anonymization techniques (Original vs. k -anonymity vs. l -diversity)	9
3.1	Fundamental aspects of GAN-based approaches for synthetic EHR time series generation	37
4.1	Theoretically motivated initialization of MH scoring function weights.	57
5.1	Summary of Key Hyperparameters	65
5.2	Fidelity metrics	67
5.3	Utility metrics	68
5.4	Privacy and adversarial accuracy metrics	68
5.5	Ablation Study Results	73

Abstract

The increasing use of Electronic Health Records (EHRs) has created new opportunities for medical research and artificial intelligence, while raising significant concerns about patient privacy. Consequently, data anonymization has become essential for enabling secure data sharing without compromising the confidentiality of sensitive information. Traditional anonymization techniques, such as suppression, generalization, and differential privacy, often struggle to preserve the utility of complex longitudinal medical data. As an alternative, synthetic data generation has emerged as a promising anonymization approach, where artificial patient records are generated instead of releasing real ones, thereby preserving both privacy and data utility. Among deep learning methods, Generative Adversarial Networks (GANs) have shown great potential for synthetic medical data generation. However, existing GAN-based approaches suffer from two major limitations: mode collapse, which reduces the diversity of generated samples, and their black-box nature, which fails to explicitly model patients' physiological states. To address these challenges, this thesis proposes a novel anonymization framework based on a Differentiable Hidden Markov Model (dHMM)-guided GAN. The proposed model combines explicit physiological state modeling with an LSTM-based patient history tracker to improve temporal consistency, while a Metropolis-Hastings-inspired latent space exploration strategy enhances sample diversity and mitigates mode collapse. Experimental results demonstrate that the proposed approach generates realistic, diverse, and clinically coherent synthetic medical records, providing an effective solution for privacy-preserving healthcare data sharing.

Keywords: synthetic EHR, generative adversarial networks, differentiable HMM, Metropolis–Hastings refinement, privacy-preserving data synthesis, HMM-GAN.

الملخص

أدى الاستخدام المتزايد للسجلات الصحية الإلكترونية (EHRs) إلى خلق فرص جديدة للأبحاث الطبية والذكاء الاصطناعي، في حين أثار مخاوف كبيرة بشأن خصوصية المرضى. ونتيجة لذلك، أصبح إخفاء هوية البيانات أمراً ضرورياً لتمكين المشاركة الآمنة للبيانات دون المساس بسرية المعلومات الحساسة. وغالباً ما تواجه تقنيات إخفاء الهوية التقليدية، مثل الحجب، والتعميم، والخصوصية التفاضلية، صعوبة في الحفاظ على فائدة البيانات الطبية الطويلة المعقدة.

وكبدل لذلك، برز توليد البيانات التركيبية كنهج واعد لإخفاء الهوية، حيث يتم إنشاء سجلات مرضى اصطناعية بدلاً من نشر السجلات الحقيقية، مما يحافظ على كل من الخصوصية وفائدة البيانات. ومن بين طرق التعلم العميق، أظهرت شبكات التوليد التنافسية (GANs) إمكانات كبيرة في توليد البيانات الطبية التركيبية. ومع ذلك، تعاني الأساليب الحالية القائمة على شبكات (GANs) من قيدين رئيسيين: انهيار النمط، والذي يقلل من تنوع العينات المولدة، وطبيعة "الصندوق الأسود" الخاصة بها، والتي تفشل في نمذجة الحالات الفسيولوجية للمرضى بشكل صريح.

ولمعالجة هذه التحديات، تقترح هذه الأطروحة إطار عمل مبتكر لإخفاء الهوية يعتمد على شبكة توليد تنافسية موجهة بنموذج ماركوف المخفي القابل للتفاضل. يجمع النموذج المقترح بين النمذجة الصريحة للحالة الفسيولوجية ومتتبع لتاريخ المريض يعتمد على الذاكرة طويلة قصيرة المدى (LSTM) لتحسين الاتساق الزمني، في حين تعمل استراتيجية استكشاف الفضاء الكامن المستوحاة من خوارزمية "متروبوليس-هاستينغز" (Metropolis-Hastings) على تعزيز تنوع العينات والتخفيف من انهيار النمط. وثبتت النتائج التجريبية أن النهج المقترح يولد سجلات طبية تركيبية واقعية ومتنوعة ومتماسكة سريرياً، مما يوفر حلاً فعالاً لمشاركة بيانات الرعاية الصحية مع الحفاظ على الخصوصية.

الكلمات المفتاحية: السجلات الصحية الإلكترونية الاصطناعية، شبكات التوليد التنافسية، (GANs) نموذج ماركوف المخفي القابل للتفاضل، تنقيح متروبوليس-هاستينغز، توليد البيانات مع الحفاظ على الخصوصية.

Acknowledgements

A scientific journey often begins with the illusion of solitary discovery. We look toward the horizon, believing we are charting new territory entirely on our own. Yet, as the work deepens, a profound truth slowly reveals itself: we are merely looking from vantage points built by others.

“If I have seen further, it is by standing on the shoulders of giants.”

Isaac Newton

I first encountered Isaac Newton’s famous words in the acknowledgments section of my supervisor’s doctoral thesis, where he used them to honor his own mentors. Today, I return to them to acknowledge the same continuity of mentorship and knowledge, and to express my gratitude for the guidance that has shaped my path and made this work possible.

My deepest gratitude goes to Professor **Kabou Salaheddine** for his constant support, availability, and guidance throughout this research, as well as for the trust and valuable knowledge he shared beyond the scope of this thesis. I also sincerely thank my previous supervisor, **Mr. Kohili Mohammed**, whose early belief in my abilities and essential guidance laid the foundation for all the research that followed.

I am also deeply grateful to **the members of the jury**. I thank them for dedicating their time, attention, and expertise to examine and evaluate this work.

Finally, my sincere thanks extend to **the professors in my department** and to everyone who contributed, whether directly or indirectly, to bringing this journey to fruition.

Dedication

{وَأَخِرُ دَعْوَاهُمْ أَنِ الْحَمْدُ لِلَّهِ رَبِّ الْعَالَمِينَ}

To those who, by Divine grace, have been the strength behind my steps, **this work belongs to the people who made it possible...**

To my **father**, my first teacher and my foremost guide.

To my beloved **mother**, whose grace and sacrifice provided the sanctuary in which I could grow.

To my **Siblings**, my steadfast pillars and my strength in this life.

To every soul who sees a reflection of their kindness in these pages, and to all who recognize their invisible hand in this humble effort

I dedicate this work.

Abbreviations list

EHR	electronic health record
ICU	intensive care unit
GAN	generative adversarial network
WGAN-GP	Wasserstein generative adversarial network with gradient penalty
HMM	Hidden Markov Model
dHMM	differentiable Hidden Markov Model
MCMC	Markov chain Monte Carlo
GRU	gated recurrent unit
LSTM	long short-term memory
GDPR	General Data Protection Regulation
HIPAA	Health Insurance Portability and Accountability Act
MH	Metropolis–Hastings
CIA	confidentiality, integrity, and availability
DP	differential privacy

QI	quasi-identifier
SDC	statistical disclosure control
ACF	autocorrelation function
AUPRC	area under the precision–recall curve
AUROC	area under the receiver operating characteristic curve
FID	Fréchet inception distance
JSDMIA	Jensen–Shannon divergence membership-inference assessment
KL	Kullback–Leibler
TSTR	train on synthetic, test on real
NNAA	nearest-neighbour adversarial accuracy
PRDC	precision, recall, density, and coverage
MIA	membership inference attack
MIMICI	Medical Information Mart for Intensive Care
MIMICI-III	Medical Information Mart for Intensive Care III

General Introduction

This chapter introduces and motivates the research presented in this thesis. It describes the application domain, the key challenges that motivate our work, and outlines our main contribution. It also sketches the overall framework and the organization of the thesis.

Context and motivation

Electronic health records (EHRs) document a patient’s trajectory of clinical events over time: diagnoses, vital signs, lab results, medications, procedures and clinical notes. That record over time is a rich resource for understanding how diseases develop, building prediction tools and aiding clinical decisions. But EHRs are also sensitive: Names, identifiers and personal health information run through the data, so indiscriminate sharing presents real ethical and legal risks.

Deep generative models suggest one pathway forward: building synthetic datasets that retain much of the statistical and clinical structure of real records while reducing risk of disclosure. Still, applying these models to clinical time series is challenging. Patient trajectories are manifestations of hidden underlying physiological states and clinical interventions.

Our approach is deliberately hybrid. We combine a temporally explicit, physiology-constrained backbone with flexible neural observation models and an iterative latent-space refinement step. The goal is practical: provide more clinical realism and downstream usefulness while keeping privacy risk low.

Problem statement

Sharing patient-level EHRs would accelerate advances in medical AI, but secrecy and regulation render direct sharing challenging. Naïve data release is obstructed by identifiers and sensitive clinical attributes. In a time-series setting, standard mitigation strategies falter:

- **Classical anonymization.** These methods are generally interpretable and often

provide formal guarantees. But this process often dulls or erases temporal information, hindering performance on sequence-based tasks.

- **Black-box generative models.** Deep generators are capable of modeling highly complex, highdimensional distributions, but often lack interpretable temporal structure and can memorize individual records.
- **Privacy-enhancing techniques.** Formal methods provide provable safety guarantees, yet naively applying such techniques to irregular high-dimensional clinical time series often sacrifices fidelity and downstream utility.

This thesis tackles two central questions:

1. How can we synthesize clinical time series that preserve clinically meaningful temporal structure and downstream utility while reducing disclosure risk?
2. Can physiologically informed temporal conditioning be combined with adversarial learning to improve realism without substantially increasing privacy leakage?

Contributions

We propose a principled hybrid pipeline for EHR time-series synthesis and evaluate its utility and privacy properties. The main contributions are:

1. **Differentiable HMM .** Rather than fixing an HMM transition matrix in advance, we learn a differentiable HMM end-to-end. Transition probabilities are conditioned on an LSTM encoding of patient history and on previous latent states. In practice this lets the model capture long-range, non-Markovian effects and keeps generated sequences within sensible physiological regimes.
2. **Metropolis–Hastings for latent refinement.** To improve sample fidelity without encouraging memorization, we introduce a two-stage, geometry-aware Metropolis–Hastings refinement. It explores multiple latent modes, nudges the sampler away from

overfitting to single records, and promotes diversity, yielding more realistic but controlled outputs.

3. **Architecture tuned for clinical signals.** The generator is built with clinical data in mind: multi-frequency convolutional pathways pick up both fast and slow physiological rhythms, while a Transformer encoder enforces global consistency across long trajectories. Taken together, these design choices make synthesized sequences more clinically plausible.
4. **Practical contribution: an evaluative framework** Beyond model design, the thesis contributes a reproducible evaluation framework for privacy-sensitive deployment: how to design clinical-state conditioning, how to perform latent refinement without overfitting, and which suite of metrics jointly assess fidelity, utility and privacy.

thesis structure

The thesis is organized into three parts: Background, State of the Art, and Contributions, followed by conclusions.

Part I: Background

Chapter 2: Privacy foundations and Statistical Disclosure Control (SDC) Survey of classical anonymization models (k-anonymity, ℓ -diversity, t-closeness), perturbative techniques (noise addition, swapping, resampling), and disclosure risk types (membership inference, attribute inference, reconstruction). These foundations motivate the privacy-aware design choices adopted later.

Chapter 3: Deep learning building blocks for time-series modelling Review of optimization and loss fundamentals and the core architectures used in this work (MLPs, CNNs, RNNs/LSTMs, Transformers and GAN variants), providing the technical background

necessary to understand subsequent methods.

Part II: State of the Art

Chapter 4: Generative modelling for EHRs A state-of-the-art review of methods for tabular and time-series EHR synthesis, differential-privacy adaptations, and probabilistic/Markovian sampling strategies. The chapter identifies gaps in temporal modeling, diversity, interpretability and privacy guarantees that motivate the hybrid approach developed in Part III.

Part III: Contribution

Chapter 5: HMM-GAN Detailed presentation of the proposed architecture: the differentiable HMM for temporal conditioning, the WGAN-GP observation model, the LSTM encoder for patient history, preprocessing pipelines and the tempered Metropolis–Hastings latent refinement. Each component is justified clinically and technically.

Chapter 6: Experiments and evaluation Description of dataset selection and preprocessing, baselines, experimental protocol and hyperparameter choices, and the complete evaluation metric suite. Results include fidelity and utility tables, privacy diagnostics, visualizations and an ablation study isolating the contribution of each major component.

Chapter 1

Classical Anonymization Methods

1.1 Introduction

This chapter surveys classical anonymization and statistical disclosure control (SDC) techniques for microdata, focusing on methods that reduce disclosure risk while attempting to preserve analytic utility. The body of the chapter presents the main non-perturbative models, including k -anonymity, ℓ -diversity and t -closeness, then describes key perturbative approaches: swapping, re-sampling, and noise-addition mechanisms, and highlights the principal advantages and limitations.

1.2 Statistical Disclosure Control methods for microdata protection

Microdata are datasets made up of individual-level records: a microdata set T can be written as

$$T = \{t_1, t_2, \dots, t_n\},$$

where each t_i is a tuple of attribute values for one person. Statistical Disclosure Control (SDC) [Hundepool et al., 2012] encompasses the techniques used to protect microdata. Its

goal is to reduce the risk that an intruder can learn sensitive information about identified individuals from a released microdata file while preserving as much utility as possible for legitimate statistical or machine-learning purposes.

Attributes in microdata are commonly grouped into four categories: [Torra and Navarro-Arribas, 2023]

- **Identifiers:** Attributes that directly identify a person (e.g., name, Social Security Number, passport number).
- **Quasi-identifiers (QIs):** Attributes that do not identify by themselves but can, when combined, produce a unique signature that enables re-identification (e.g., date of birth, gender, ZIP code, profession, ethnic group).
- **Sensitive (confidential) attributes:** Highly private attributes (e.g., religion, disease, lab results, political opinion).
- **Non-sensitive attributes:** Attributes that do not contain sensitive information.

It was once thought that simply removing explicit identifiers would be sufficient to anonymise data. However, combinations of quasi-identifiers can often re-identify individuals despite the absence of direct identifiers. Sweeney’s study showed that 87% of the U.S. population can be uniquely identified using only the QI set {5-digit ZIP, gender, date of birth} [Sweeney, 2000]. Results like this motivated both data controllers and data subjects to demand stronger protections and inspired new SDC techniques to reduce disclosure risk. SDC methods therefore aim to prevent several risk types, including membership inference (deciding if an individual’s record was used to train an algorithm) [Shokri et al., 2017], attribute inference (inferring additional attributes from partial knowledge and model outputs) [Al Zamal et al., 2012], and reconstruction attacks (extracting full records from model outputs) [Dinur and Nissim, 2003].

SDC methods can be divided into two categories: non-perturbative and perturbative.

1.3 Non-perturbative methods

Non-perturbative methods change the granularity or presence of values without adding stochastic noise.

Some well-known non-perturbative techniques include: publishing only a sample of the original data file (*Sampling*); coarsening a categorical attribute by combining several categories into a more general one (*Generalisation*), or replacing numerical values with intervals; putting values above/below a given threshold into a single category (*Top/Bottom coding*) [Templ et al., 2015] or suppressing certain values of individual attributes to decrease the uniqueness of the individual (*Local suppression*), removing or blanking out specific values or records (e.g., dropping rare records or replacing a ZIP with ***). Group records into clusters and release cluster centroids or aggregated summaries (*Microaggregation*)

[Domingo-Ferrer and Mateo-Sanz, 2002] Exchange attribute values among similar records to preserve marginal distributions but break linkage (*Data swapping/permutation*) [?].

Syntactic privacy models are constraint-based methods that enforce equivalence classes on QIs. They are often realised through suppression, generalisation, and microaggregation. [Carvalho et al., 2023]

1.3.1 k-anonymity

The main method for privacy protection, from which many others have been developed, is *k*-anonymity [Sweeney, 2002]. The concept of *k*-anonymity was defined to release personal data while safeguarding the identities of the individuals to whom the data refer. A dataset is *k*-anonymous if each record is indistinguishable from at least $k - 1$ other records within the dataset when considering the values of its QIs. However, when the sensitive attributes on a group of *k*-anonymous records have low variability (e.g., when they are all equal), there is no need for reidentification to disclose the value of the sensitive attribute of a record. This

remark was made in [Machanavajjhala et al., 2007], who proposed the model of ℓ -diversity for solving this issue. Table 1.1 presents a Comparison of original microdata vs. 4-anonymous generalization.

Table 1.1: Comparison of original microdata vs. 4-anonymous generalization.

Inpatient microdata.					4-anonymous inpatient microdata.				
	Non-Sensitive			Sensitive		Non-Sensitive			Sensitive
	Zip Code	Age	Nationality	Condition		Zip Code	Age	Nationality	Condition
1	13053	28	Russian	Heart Disease	1	130**	< 30	*	Heart Disease
2	13068	29	American	Heart Disease	2	130**	< 30	*	Heart Disease
3	13068	21	Japanese	Viral Infection	3	130**	< 30	*	Viral Infection
4	13053	23	American	Viral Infection	4	130**	< 30	*	Viral Infection
5	14853	50	Indian	Cancer	5	1485*	≥ 40	*	Cancer
6	14853	55	Russian	Heart Disease	6	1485*	≥ 40	*	Heart Disease
7	14850	47	American	Viral Infection	7	1485*	≥ 40	*	Viral Infection
8	14850	49	American	Viral Infection	8	1485*	≥ 40	*	Viral Infection
9	13053	31	American	Cancer	9	130**	3*	*	Cancer
10	13053	37	Indian	Cancer	10	130**	3*	*	Cancer
11	13068	36	Japanese	Cancer	11	130**	3*	*	Cancer
12	13068	35	American	Cancer	12	130**	3*	*	Cancer

1.3.2 ℓ -diversity

ℓ -diversity [Machanavajjhala et al., 2007] strengthens k -anonymity by requiring that, within every equivalence class (i.e., every group of records sharing the same quasi-identifier values), the sensitive attribute exhibits sufficient diversity. Formally, a k -anonymous dataset is said to satisfy ℓ -diversity if, for each equivalence class, there are at least ℓ well-represented, distinct values of the sensitive attribute. This requirement mitigates inference risks that remain under k -anonymity: for instance, if all records in a class share the same sensitive value, an adversary can learn that value with certainty (a homogeneity attack). Likewise, knowing an individual belongs to a particular class may allow an adversary to infer that individual's sensitive value using external information (a background-knowledge attack).

Several refinements of ℓ -diversity have been proposed. *Entropy ℓ -diversity* requires that the entropy of the sensitive-attribute distribution within each equivalence class be at least $\log \ell$, thereby ensuring not only multiple distinct values but also an even spread among them. *Recursive (c, ℓ) -diversity* [Chakraborty and Tripathy, 2016] privacy imposes a stronger constraint on dominant values by bounding the cumulative frequency of the most frequent

sensitive values relative to ℓ , so that a single or small set of values cannot overwhelmingly determine the class distribution. As with k -anonymity, there is no universally optimal choice of ℓ ; typically $\ell \leq k$ and $\ell > 1$. It has been shown, however, that ℓ -diversity alone does not prevent attribute disclosure when the overall distribution of the sensitive attribute is highly skewed, which motivated the t -closeness model. Table 1.2 presents a comparison of data anonymisation techniques.

Table 1.2: Comparison of data anonymization techniques (Original vs. k -anonymity vs. ℓ -diversity)

Original Dataset					2-anonymous					2-anonymous 2-diverse				
Nr	Age	Sex	City	Disease	Nr	Age	Sex	City	Disease	Nr	Age	Sex	City	Disease
1	16	M	Rot.	Viral inf.	1	≤ 20	M	Rot.	Viral inf.	1	≤ 20	M	Rot.	Viral inf.
2	18	M	Rot.	Heart-rel.	2	≤ 20	M	Rot.	Heart-rel.	2	≤ 20	M	Rot.	Heart-rel.
3	19	M	Rot.	Cancer	3	≤ 20	M	Rot.	Cancer	3	≤ 20	M	Rot.	Cancer
4	22	F	Rot.	Viral inf.	4	20-30	F	Rot.	Viral inf.	4	20-30	F	●	Viral inf.
5	22	M	Zwo.	No illness	5	20-30	M	Zwo.	No illness	5	20-30	M	Zwo.	No illness
6	23	M	Zwo.	Tuberc.	6	20-30	M	Zwo.	Tuberc.	6	20-30	M	Zwo.	Tuberc.
7	24	M	Zwo.	Heart-rel.	7	20-30	M	Zwo.	Heart-rel.	7	20-30	M	Zwo.	Heart-rel.
8	25	F	Utr.	Cancer	8	20-30	F	Utr.	Cancer	8	20-30	F	●	Cancer
9	26	F	Rot.	Heart-rel.	9	20-30	F	Rot.	Viral inf.	9	20-30	F	●	Viral inf.
10	28	F	Utr.	Tuberc.	10	20-30	F	Utr.	Cancer	10	20-30	F	●	Cancer

1.3.3 t -closeness

t -closeness [Li et al., 2007] formalizes protection against global background knowledge by requiring that, for every equivalence class, the distribution of a sensitive attribute resembles the distribution of that attribute in the entire dataset: specifically, the distance between the two distributions must not exceed a threshold t . This constraint limits the amount of individual-specific information an adversary can infer from any class. To capture semantic or numeric relationships among sensitive values, the Earth Mover’s Distance (EMD) metric is used to quantify the distance between distributions [Andoni et al., 2008]. An equivalence class satisfies t -closeness when the measured distance between its sensitive-attribute distribution and the dataset’s overall distribution is at most t ; a table satisfies t -closeness if every equivalence class meets this criterion.

1.4 Perturbative methods

Perturbative techniques protect privacy by deliberately randomising data or outputs so that individual records cannot be recovered exactly while important statistical properties of the collection are partly preserved. Common perturbative methods include swapping, re-sampling, and noise addition; each family differs in where randomness is injected, the kind of utility preserved, and whether formal privacy guarantees are available.

1.4.1 Swapping

Attribute (value) swapping protects privacy by exchanging values of one or more attributes between records, breaking the direct link between an original record and its sensitive value while largely preserving marginal statistics. Typical procedures match records on a set of quasi-identifiers (for example, age group or region) and swap sensitive values between similar but distinct records [Dalenius and Reiss, 1982]. Two common variants are record (data) swapping and rank swapping [Moore, 1996]. Although data swapping was originally proposed for categorical attributes, rank swapping can be applied to continuous attributes by swapping ranks rather than raw values.

1.4.2 Re-sampling

Re-sampling methods hide individual contributions by replacing raw values with aggregates or by releasing draws from sampled subsets or fitted models. Examples include bootstrap-style masking, drawing many small samples, computing per-sample averages, and using those averages as masked values, as well as microaggregation, which clusters records into groups of size at least k and replaces values with the group centroid. These approaches hide individual contributions inside aggregates and often preserve means and variances when grouping is quality-driven, but they can introduce bias, remove fine-grained structure, and depend critically on grouping quality or generative-model fidelity. [Carvalho et al., 2023]

1.4.3 Noise addition

Noise addition is a perturbative privacy technique that protects sensitive information by injecting random perturbations into data, query responses, or model parameters. The fundamental aim is to obscure the precise contribution of any single record while preserving aggregate statistical properties that support analysis and downstream modeling. Noise may be applied at multiple points in the data-processing pipeline: directly to raw attribute values, to aggregated query answers (for example counts or means), or to model updates during training.

A widely used theoretical formalization of this idea is Differential Privacy (DP), introduced by Dwork *et al.* in 2006. [Dwork, 2006] Informally, a differentially private mechanism ensures that the output distribution of a randomized algorithm changes only minimally when any single record is added or removed from the input dataset. Thus, an adversary observing the output gains only limited information about any particular individual. DP treats privacy as an accumulative resource: each application of a private mechanism consumes a portion of a finite privacy budget, typically parameterized by ϵ . Formally, a randomized mechanism $M : \mathcal{D}^n \rightarrow \mathcal{R}$ satisfies ϵ -differential privacy if, for every pair of adjacent databases $d, d' \in \mathcal{D}^n$ and every measurable set $S \subseteq \mathcal{R}$,

$$\Pr[M(d) \in S] \leq e^\epsilon \Pr[M(d') \in S] \tag{1.1}$$

A common relaxation, (ϵ, δ) -differential privacy, allows a small probability δ of larger deviations by requiring

$$\Pr[M(d) \in S] \leq e^\epsilon \Pr[M(d') \in S] + \delta \tag{1.2}$$

1.4.4 Advantages

Noise-based and classical perturbative approaches remain attractive for several practical reasons:

- **Conceptual clarity and interpretability:** Methods such as additive noise, rank swapping, or microaggregation are straightforward to explain to stakeholders, auditors, and data subjects. This simplicity facilitates transparency and simplifies ethical and legal reviews.
- **Low implementation cost:** Many perturbative techniques are computationally lightweight and can be integrated into existing data pipelines with minimal engineering overhead.
- **Formal guarantees:** These perturbative approaches, and Differential Privacy in particular, provide mathematically proven assurances, shifting the paradigm away from merely “hiding” identifiers toward using probability theory to bound disclosure risk and guarantee anonymity.

1.4.5 Challenges

Despite their practical strengths, perturbative anonymization and noise-addition strategies possess significant limitations that must be carefully weighed when designing data-release protocols:

- **Vulnerability to auxiliary information:** Parameterized models, including k -anonymity and various simple perturbation schemes, are susceptible to adversaries possessing external, correlated datasets. This vulnerability can lead to background-knowledge or homogeneity attacks. Similarly, swapping and aggregation techniques can often be partially reverse-engineered if auxiliary data is available.

- **Parameter selection and interpretability:** Determining optimal parameter values (e.g., k , ℓ , and t in syntactic models, or the privacy budget ϵ in differential privacy) is highly context-dependent and frequently lacks principled, domain-specific guidelines. Suboptimal parameter choices risk either exposing residual sensitive information or unnecessarily degrading the downstream analytical utility of the data.
- **Limitation of Differential Privacy with multiple records per individual:** The standard DP definition is record-level: guarantees are formulated for neighbouring datasets that differ in a single row. When an individual contributes multiple records (longitudinal logs, repeated transactions, many medical entries), the effective privacy loss for that person can grow linearly with their number of records under basic composition or group-privacy bounds.

1.5 Conclusion

This chapter surveyed classical anonymization methods for microdata, contrasting non-perturbative techniques (suppression, generalisation, microaggregation, swapping) with perturbative approaches (re-sampling and noise addition). Perturbative methods, most notably Differential Privacy, provide formal, composable guarantees but require careful calibration and incur an explicit utility–privacy trade-off. There remains substantial work on balancing utility and privacy; the next chapter presents a deep-learning approach aimed at addressing these trade-offs.

Chapter 2

Deep Learning: Fundamentals and Architectures

2.1 Introduction

This chapter reviews the deep-learning tools and concepts that underpin the methods explored in this thesis. It begins with the fundamentals of deep learning: the view of neural networks as parametric function approximators, the common loss functions, and the optimisation machinery used to fit large models (backpropagation, stochastic optimisation and regularisation). Building on this foundation, the chapter surveys core architectures used in practice, including multilayer perceptrons (MLPs), convolutional neural networks (CNNs), recurrent architectures (RNNs, LSTM, GRU, and bidirectional variants), and generative adversarial networks (GANs).

2.2 Fundamentals of Deep Learning (models, optimization, loss)

2.2.1 Supervised and unsupervised learning

Supervised learning. In supervised learning the training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ contains input–output pairs where $x_i \in \mathcal{X}$ (features) and $y_i \in \mathcal{Y}$ (labels or targets). The goal is to learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ that generalises from the observed examples to unseen inputs. Typical supervised tasks include classification (discrete $\mathcal{Y}$) and regression (continuous $\mathcal{Y}$).

Unsupervised learning. In unsupervised learning, the dataset contains only inputs, $\mathcal{D} = \{x_i\}_{i=1}^N$, and the goal is to discover structure in the input data. Examples include clustering, dimensionality reduction, and generative modelling. Many deep-learning models (autoencoders, variational autoencoders, GANs) operate in unsupervised learning.

2.2.2 Neural networks as function approximators

A deep neural network defines a parameterised family of functions $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$, where θ denotes all learnable parameters (weights and biases). Learning reduces to choosing θ so that f_θ approximates the (unknown) data-generating mapping. This view emphasises two points:

1. **Parametric model.** The architecture (number of layers, layer types and connectivity) defines a parametric class of functions; the parameter vector θ lives in a high-dimensional space $\mathbb{R}^P$.
2. **Function approximation.** Neural networks are universal function approximators: with sufficient width/depth and suitable nonlinearity they can approximate a wide class of continuous functions to arbitrary accuracy (subject to optimisation and generalisation constraints). In practice, architecture choices and regularisation determine which functions are learned.

2.2.3 Building blocks: layers, parameters, activation functions

Layers and operations. A typical feedforward network is composed of layers that transform their input $h^{(\ell-1)}$ into an output $h^{(\ell)}$. A simple fully connected layer computes:

$$h^{(\ell)} = \phi(W^{(\ell)}h^{(\ell-1)} + b^{(\ell)}) \quad (2.1)$$

where $W^{(\ell)} \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ and $b^{(\ell)} \in \mathbb{R}^{d_\ell}$ are the layer's parameters and $\phi(\cdot)$ is an elementwise nonlinearity.

Parameters. Parameters $\theta = \{W^{(\ell)}, b^{(\ell)}, \dots\}$ are the quantities adjusted during training. The number of parameters P can range from thousands (small networks) to billions (modern large models).

Activation functions. Nonlinear activations enable networks to represent non-linear mappings. Common choices include: [\[Bai, 2022\]](#) [\[Kyurkchiev and Markov, 2015\]](#) [\[Franke and Degen, 2023\]](#)

- **ReLU:**

$$\text{ReLU}(z) = \max(0, z) \quad (2.2)$$

- **Sigmoid:**

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (2.3)$$

- **Tanh:**

$$\tanh(z) \in (-1, 1) \quad (2.4)$$

- **Softmax:**

$$\text{softmax}(z)_k = \exp(z_k) / \sum_j \exp(z_j) \quad (2.5)$$

2.2.4 Optimization objective: empirical risk minimization

Learning is typically cast as minimising an empirical risk (average loss) plus optional regularisation. Given a loss function $\ell(y, \hat{y})$ measuring discrepancy between true target y and

prediction $\hat{y} = f_{\theta}(x)$, the empirical risk is

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^N \ell(y_i, f_{\theta}(x_i)) + \lambda R(\theta), \quad (2.6)$$

where $R(\theta)$ is a regulariser (e.g., $\frac{1}{2}\|\theta\|^2$) and $\lambda \geq 0$ its weight. Minimising $\mathcal{L}(\theta)$ trades off data fit and model complexity.

2.2.5 Gradient-based learning: backpropagation and stochastic optimization

Directly minimising $\mathcal{L}(\theta)$ is done via iterative, gradient-based optimisation. The two central ideas are backpropagation (efficient gradient computation) and stochastic optimisation (scalable parameter updates).

Backpropagation (chain rule).

Backpropagation computes $\nabla_{\theta}\mathcal{L}(\theta)$ efficiently by applying the chain rule through the computational graph of the network. For each layer, the gradients of the loss with respect to its inputs and parameters are computed from gradients at the layer's outputs, proceeding from the output layer back to the input layer. This yields gradients for every weight and bias at a cost comparable to one or two forward passes.

Gradient descent and stochastic variants.

The basic gradient-descent update is

$$\theta \leftarrow \theta - \eta \nabla_{\theta} \mathcal{L}(\theta),$$

where $\eta > 0$ is the learning rate. For large datasets, full-batch gradients are expensive, so stochastic gradient descent (SGD) uses an unbiased estimate of the gradient computed on

a minibatch $\mathcal{B} \subset \{1, \dots, N\}$:

$$\theta \leftarrow \theta - \eta \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \nabla_{\theta} \ell(y_i, f_{\theta}(x_i)).$$

Mini-batch SGD balances computational efficiency, gradient variance and parallelism. Common extensions that improve convergence and robustness include:

- **Momentum**: accumulates a velocity vector to damp oscillations (e.g. $v \leftarrow \mu v - \eta \nabla_{\theta} \mathcal{L}(\theta)$, $\theta \leftarrow \theta + v$).
- **Adaptive methods** (Adam, RMSprop): scale updates per-parameter using running estimates of gradient moments (e.g. Adam uses biased first and second moment estimates with bias correction). [Shi et al., 2021](#)

2.3 Core architectures (MLP, CNN, RNN, GANs)

2.3.1 Multilayer Perceptrons (MLPs)

A Multilayer Perceptron represents a foundational class of feedforward artificial neural networks (ANNs). It is characterized by a hierarchical architecture consisting of an input layer, one or more hidden layers, and an output layer. As a cornerstone of connectionist models, the MLP is distinguished by its use of non-linear activation functions and the backpropagation algorithm, enabling the resolution of problems that are not linearly separable.

For an L -layer MLP the forward pass is

$$\begin{aligned} h^{(0)} &= x, \\ h^{(\ell)} &= \phi(W^{(\ell)}h^{(\ell-1)} + b^{(\ell)}), \quad \ell = 1, \dots, L, \\ \hat{y} &= f_{\theta}(x) = h^{(L)}. \end{aligned} \tag{2.7}$$

Here $W^{(\ell)}$ and $b^{(\ell)}$ are the weights and biases of layer ℓ , ϕ is the (elementwise) activation

function, and $\theta = \{W^{(\ell)}, b^{(\ell)}\}_{\ell=1}^L$ denotes the collection of all network parameters.

2.3.2 Convolutional Neural Network Architecture

Convolutional layers

Convolutional layers are foundational components of Convolutional Neural Networks (CNNs) [LeCun et al., 2002]. They apply a set of learnable filters (kernels) across the input volume; at each spatial position the filter and the corresponding input patch are multiplied element-wise and summed, which allows the layer to capture local patterns and salient features.

Mathematically, writing the input feature map as x and a single filter as w , the convolution at spatial location (i, j) can be written as

$$(y * w)[i, j] = \sum_u \sum_v x[i + u, j + v] w[u, v], \quad (2.8)$$

where the sums run over the spatial support of the filter. A convolutional layer typically contains C_{out} filters (also called channels), each producing one output feature map; the layer output therefore has depth C_{out} .

Important hyper-parameters are:

- Kernel (filter) size: K (spatial extent, e.g. 3×3).
- Stride: S (step size for sliding the filter).
- Padding: P (number of rows/columns added to the borders).
- Number of output channels (filters): C_{out} .

The spatial size of the output (assuming square input and kernel for simplicity) is

$$N_{\text{out}} = \left\lfloor \frac{N_{\text{in}} - K + 2P}{S} \right\rfloor + 1, \quad (2.9)$$

where N_{in} is the input spatial size and $\lfloor \cdot \rfloor$ denotes the floor operation (outputs are integer-valued sizes).

Pooling layers

Pooling layers are commonly interleaved with convolutional layers to reduce spatial dimensionality, improve computational efficiency and provide a degree of translational invariance. Common pooling functions include average pooling, L^2 -norm pooling, weighted average pooling and (most commonly) *max pooling*, which returns the maximum value in each local neighborhood.

Using the same notation (K = pooling kernel size, S = stride), the output size after a pooling operation (again, assuming no explicit padding) is

$$N_{\text{out}} = \left\lfloor \frac{N_{\text{in}} - K}{S} \right\rfloor + 1. \quad (2.10)$$

Fully connected (dense) layers

After a stack of convolutional and pooling layers, CNNs often use one or more fully connected (dense) layers to produce final predictions. The feature maps are first flattened into a vector and then passed through one or more dense layers of the form

$$z = W^{\top} \text{vec}(H) + b, \quad \hat{y} = \phi_{\text{out}}(z), \quad (2.11)$$

where $\text{vec}(H)$ denotes the flattened feature maps, W and b are the dense layer parameters, and ϕ_{out} is an appropriate output activation (e.g. softmax for multi-class classification, sigmoid for binary classification, or identity for regression).

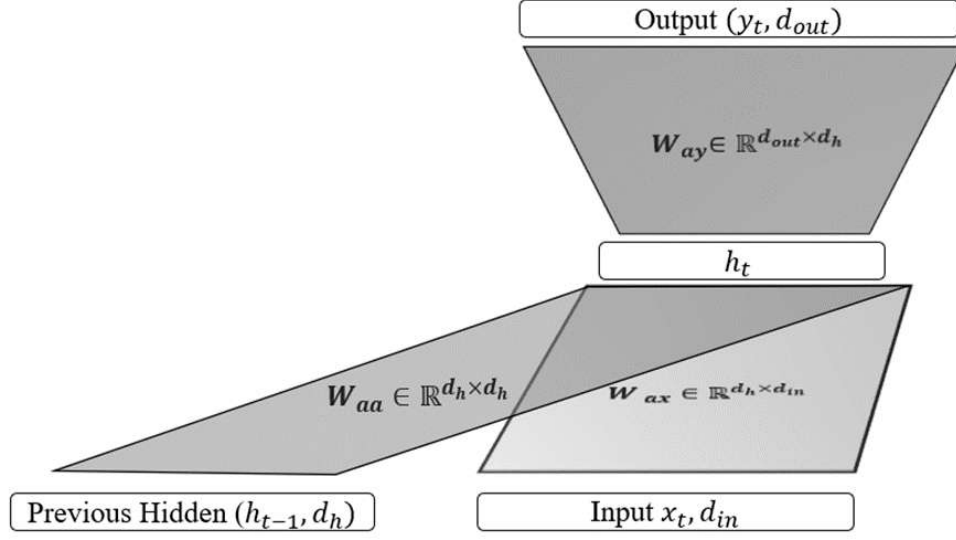


Figure 2.1: Recurrent Neural Network layer

2.3.3 Recurrent Neural Networks

Recurrent neural networks (RNNs) [Elman, 1990] are a class of neural architectures specifically designed to process sequential data, which is ubiquitous in domains such as natural language processing, speech recognition, and time-series analysis. Unlike feedforward networks, an RNN contains directed cycles in its computational graph so that information from previous time steps is preserved in an internal state.

This recurrence allows the network to accept variable-length input sequences: at each time step t , the model combines the current input $x^{(t)}$ with a memory of past computations (the previous hidden state $a^{(t-1)}$) to produce an updated hidden state $a^{(t)}$. This characteristic makes RNNs particularly valuable for tasks where the length of the input is not predetermined and needs to be accommodated dynamically during processing.

Denote the input, hidden, and output dimensions by d_{in} , d_h , and d_{out} , respectively. Given this, our three parameter matrices illustrated in Figure 2.1 are:

$$W_{ax} \in \mathbb{R}^{d_h \times d_{in}}, W_{aa} \in \mathbb{R}^{d_h \times d_h}, W_{ay} \in \mathbb{R}^{d_{out} \times d_h}.$$

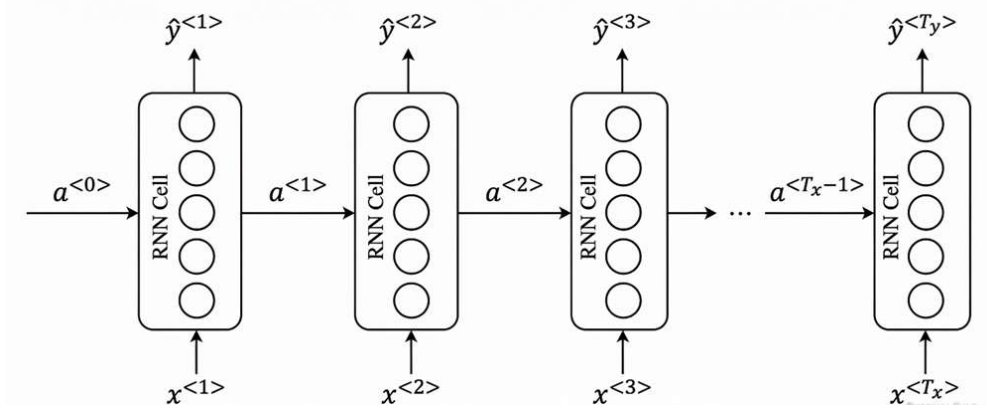


Figure 2.2: Simple recurrent neural network illustrated as a feedforward network

A simple standard RNN cell is shown in Figure [2.2](#), where the hidden state is usually initialized with zeros: $a^{(0)} = \mathbf{0}$.

The forward propagation for this RNN cell is written compactly as:

$$a^{(t)} = g_a(W_{aa}a^{(t-1)} + W_{ax}x^{(t)} + b_a) \quad (2.12)$$

$$y^{(t)} = g_y(W_{ya}a^{(t)} + b_y) \quad (2.13)$$

where $g_a(\cdot)$ is the activation applied to the hidden state and $g_y(\cdot)$ is the output activation chosen according to the task (e.g., tanh or ReLU for hidden activations; sigmoid for binary outputs; softmax for multi-class classification). Concretely, the first nontrivial step is

$$\mathbf{a}^{(1)} = g_a(W_{aa}\mathbf{a}^{(0)} + W_{ax}\mathbf{x}^{(1)} + \mathbf{b}_a), \quad (2.14)$$

and the corresponding output is

$$\hat{\mathbf{y}}^{(1)} = g_y(W_{ya}\mathbf{a}^{(1)} + \mathbf{b}_y), \quad (2.15)$$

with the recurrence above applied for $t = 1, \dots, T$ over a sequence of length T .

These equations illustrate the two essential roles of an RNN:

1. The trainable matrix W_{ax} maps the current input into hidden-state space.

2. The recurrent matrix W_{aa} propagates and transforms information carried by the previous hidden state.

The choice of activation functions and architectural variants (e.g., gated cells such as LSTM or GRU) governs the network's ability to capture long-range dependencies and to mitigate optimisation issues such as vanishing or exploding gradients; such considerations are treated in later sections.

Backpropagation Through Time

During the initial pass, forward inference is executed, involving the computation of h_t , y_t , and accumulating the loss at each step in time.

In the second pass, the sequence is processed in reverse. Gradients are calculated and stored as the computation progresses backward, and the error term for each step is computed and preserved for use in the hidden layer, as illustrated in Figure 2.3.

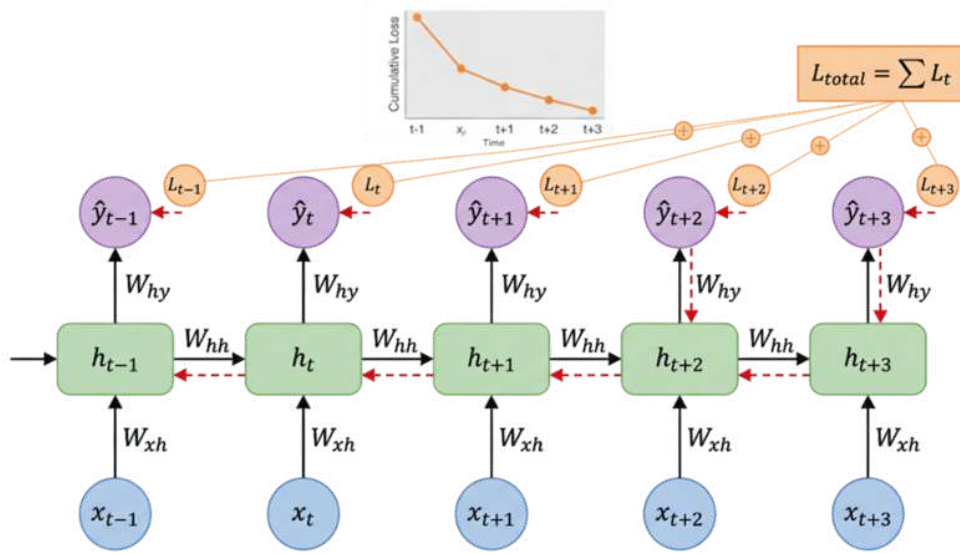


Figure 2.3: Recurrent neural network illustrated as a forward-backward network

We will use the cross-entropy loss function:

$$\mathcal{L}^{(t)}(\hat{y}^{(t)}, y^{(t)}) = -y^{(t)} \log \hat{y}^{(t)} - (1 - y^{(t)}) \log(1 - \hat{y}^{(t)}) \quad (2.16)$$

$$\mathcal{L}(\hat{y}, y) = \sum_{t=1}^T \mathcal{L}^{(t)}(\hat{y}^{(t)}, y^{(t)}) \quad (2.17)$$

Where the first equation is the loss for one example (at time step t) and the loss for the whole sequence is given by the summation over all the calculated single example losses.

Different Types of RNNs

Recurrent neural network (RNN) architectures are commonly classified by how inputs and outputs are arranged along the time axis. The configuration discussed above corresponds to a many-to-many setup, in which a sequence of inputs produces a sequence of outputs. Other frequently used patterns are:

- **One-to-Many:** A single input (for example, a latent code or a fixed context vector) is used to generate an entire output sequence (typical in sequence generation from a single conditioning signal).
- **Many-to-One:** A sequence of inputs is aggregated to produce a single output (for example, classification of a whole sequence or regression to a single scalar).

These patterns are interchangeable within the same RNN cell: the distinction lies in how the network is unrolled in time and where outputs are read out, not in the cell mechanics themselves. Figures that depict these families (e.g., Figure 2.4) are useful for clarifying how information flows during encoding and decoding phases.

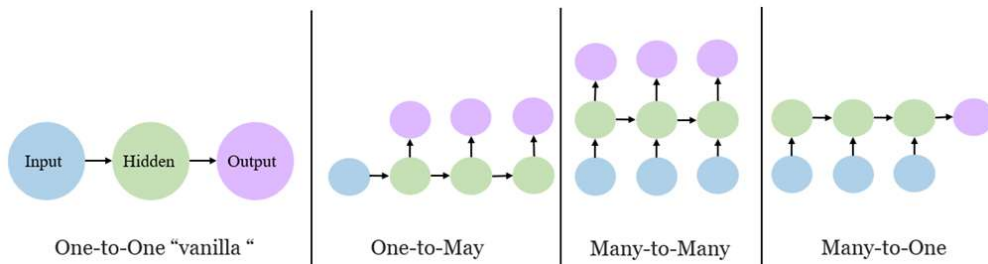


Figure 2.4: Different types of recurrent neural networks

Vanishing Gradients and Variant RNN Architectures

Training standard RNNs on long sequences is impeded by the well-known vanishing (and complementary exploding) gradient problem. During backpropagation through time, parameter updates are computed via repeated application of the chain rule, which involves a sequence of multiplicative terms from the recurrent transitions. When these terms are smaller than one, their repeated multiplication causes the gradient signal to attenuate exponentially with temporal depth. As a result, gradients propagated to early time steps become negligibly small, preventing the model from learning long-range dependencies. This failure mode motivated the development of RNN variants and training techniques designed to preserve or stabilize gradient flow and thereby capture long-term structure: [Kag et al., 2020]

Gated Recurrent Unit (GRU)

The Gated Recurrent Unit (GRU) [Cho et al., 2014] is a gated RNN cell designed to improve learning of long-range dependencies. A GRU replaces the simple recurrent update with two multiplicative gates that regulate information flow: the update gate $z^{(t)}$ and the reset gate $r^{(t)}$. Intuitively, the update gate controls how much of the previous hidden state should be retained versus replaced by new candidate information, while the reset gate controls how much past information to forget when forming the candidate.

Using $x^{(t)}$ for the input and $h^{(t-1)}$ for the previous hidden state, a common GRU formulation is:

$$z^{(t)} = \sigma(W_z x^{(t)} + U_z h^{(t-1)} + b_z) \quad (2.18)$$

$$r^{(t)} = \sigma(W_r x^{(t)} + U_r h^{(t-1)} + b_r) \quad (2.19)$$

$$\tilde{h}^{(t)} = \tanh(W_h x^{(t)} + U_h (r^{(t)} \odot h^{(t-1)}) + b_h) \quad (2.20)$$

$$h^{(t)} = (1 - z^{(t)}) \odot h^{(t-1)} + z^{(t)} \odot \tilde{h}^{(t)} \quad (2.21)$$

where $\sigma(\cdot)$ is the logistic sigmoid, $\tanh(\cdot)$ the hyperbolic tangent, and $\odot$ denotes element-wise multiplication. The hidden vector $h^{(t)}$ plays the role of the cell/memory content in GRU formulations. Because the update gate interpolates between old and new information, gradients can pass more easily across time steps than in vanilla RNNs.

Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997] cells implement a more elaborate gating architecture with an explicit cell state $C^{(t)}$ designed to carry long-term information with reduced modification. An LSTM typically uses three gates: forget $f^{(t)}$, input (or update) $i^{(t)}$, and output $o^{(t)}$ and a candidate $\tilde{C}^{(t)}$. A standard LSTM update is:

$$f^{(t)} = \sigma(W_f x^{(t)} + U_f h^{(t-1)} + b_f) \quad (2.22)$$

$$i^{(t)} = \sigma(W_i x^{(t)} + U_i h^{(t-1)} + b_i) \quad (2.23)$$

$$\tilde{C}^{(t)} = \tanh(W_c x^{(t)} + U_c h^{(t-1)} + b_c) \quad (2.24)$$

$$C^{(t)} = f^{(t)} \odot C^{(t-1)} + i^{(t)} \odot \tilde{C}^{(t)} \quad (2.25)$$

$$o^{(t)} = \sigma(W_o x^{(t)} + U_o h^{(t-1)} + b_o) \quad (2.26)$$

$$h^{(t)} = o^{(t)} \odot \tanh(C^{(t)}) \quad (2.27)$$

Here the forget gate $f^{(t)}$ selectively removes information from the previous cell state, the input gate $i^{(t)}$ controls incorporation of the new candidate, and the output gate $o^{(t)}$ determines which parts of the cell state are exposed as the hidden state. The direct additive update of the cell state C allows constant error flow across many time steps, which is key to alleviating the vanishing gradient problem.

Bidirectional RNNs (Bi-RNNs)

Standard (unidirectional) RNNs process a sequence in a single temporal direction and therefore condition predictions at time t only on the left (past) context. Many tasks, however, benefit from access to both past and future context. A bidirectional RNN [Schuster and Paliwal, 1997] addresses this by running two independent RNNs in parallel: a forward model processes the sequence from $t = 1$ to T producing forward states $\vec{h}^{(t)}$, while a backward model processes the sequence from $t = T$ to 1 producing backward states $\overleftarrow{h}^{(t)}$. The two representations are then combined:

$$h^{(t)} = [\vec{h}^{(t)}; \overleftarrow{h}^{(t)}] \quad (2.28)$$

and $h^{(t)}$ is used for downstream prediction. Bidirectional architectures capture both left and right context at each position and often improve accuracy on sequence-labelling tasks. Their principal limitation is the requirement to have the entire sequence available before computing outputs, which precludes online or streaming use where future tokens are not yet observed.

2.3.4 Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) are a class of generative models introduced by Goodfellow et al. [Goodfellow et al., 2020] that learn to approximate a data distribution via an adversarial game between two neural networks: a generator G and a discriminator D .

The generator G_{θ_g} maps samples z from a simple prior $p_z(z)$ (commonly a multivariate normal or uniform distribution) to synthetic instances $G_{\theta_g}(z)$ that aim to mimic real data. The discriminator D_{θ_d} is a classifier that receives an input x and outputs a scalar $D_{\theta_d}(x) \in (0, 1)$, interpreted as the estimated probability that x comes from the true data distribution p_{data} rather than from the generator. The two networks are trained simultaneously: D is trained to distinguish real from generated samples, while G is trained to produce samples

that fool D . Figure 2.5 depicts the GAN architecture and training loop.

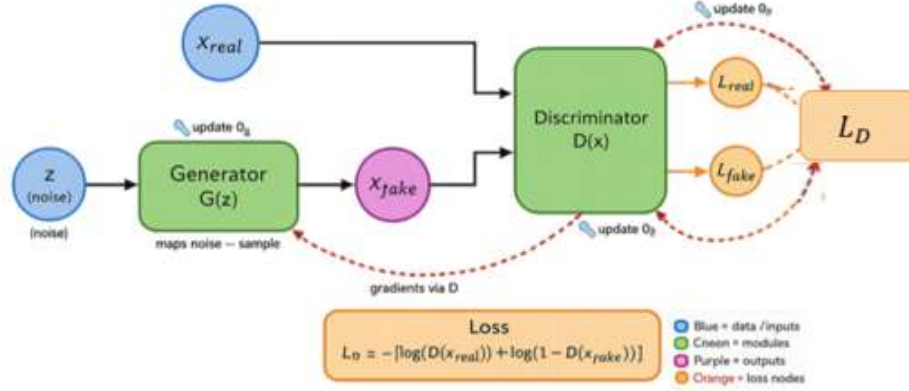


Figure 2.5: GAN architecture and training loop

The canonical GAN objective is a minimax game written as:

$$\min_{\theta_g} \max_{\theta_d} V(D, G) \quad (2.29)$$

$$\text{where } V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}} [\log D_{\theta_d}(x)] + \mathbb{E}_{z \sim p_z} [\log (1 - D_{\theta_d}(G_{\theta_g}(z)))] \quad (2.30)$$

Intuitively, the discriminator seeks to maximise the log-likelihood of correctly labelling real examples (first term) and of correctly rejecting generator samples (second term), while the generator seeks to minimise the second term so that D assigns high scores to generated examples.

The adversarial optimisation can be interpreted as a saddle-point problem in the joint parameter space of the generator and discriminator; Figure 2.6 visualises this geometry.

GAN training is notorious for practical difficulties. Typical failure modes include mode collapse (the generator produces limited diversity), training instability, and sensitivity to hyperparameters. These issues led to a variety of alternative objectives, such as the Wasserstein GAN (WGAN) and WGAN with gradient penalty (WGAN-GP), which often improve convergence behaviour and sample quality.

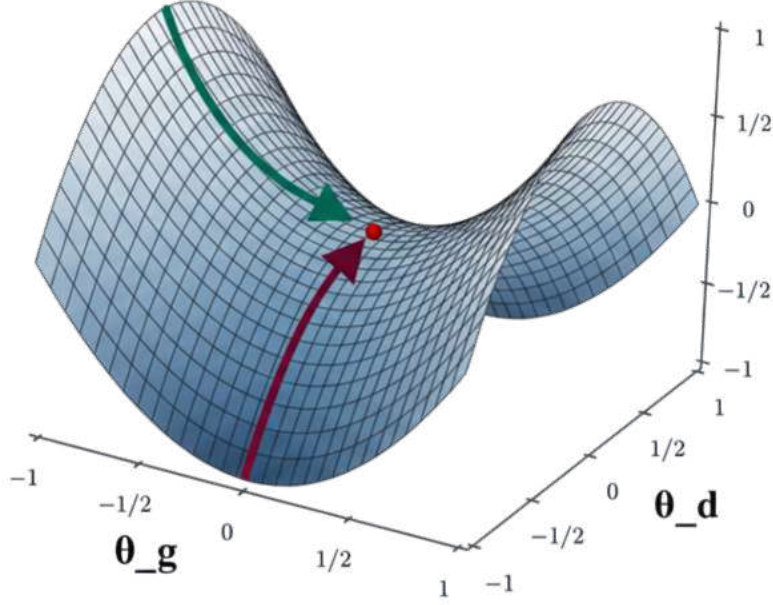


Figure 2.6: Saddle-point optimisation for GAN training

2.3.5 Attention mechanism and Transformers

An *attention* mechanism [Bahdanau et al., 2014] is a machine-learning technique that directs deep models to prioritise the most relevant parts of input data. The core of the Transformer [Vaswani et al., 2017] is the attention mechanism, which computes a context-aware representation for each sequence position by comparing a set of queries Q with keys K and combining the corresponding values V . In practice we use *scaled dot-product attention*:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V,$$

where the scale $\sqrt{d_k}$ stabilises gradients when the key/query dimensionality is large.

Transformers extend this idea with *multi-head attention*, which projects inputs into multiple query/key/value subspaces so the model can attend to different types of interactions in parallel (for example, recent trends, periodic signals, or rare events). Formally:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) W^O, \quad \text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V),$$

where W_i^Q, W_i^K, W_i^V and W^O are learned projection matrices.

The attention mechanism, therefore, enables direct, parallel modelling of long-range dependencies and flexible conditioning while remaining amenable to efficiency-oriented variants when handling very long clinical sequences such as extended ICU stays.

2.4 Conclusion

Deep learning offers powerful, flexible models capable of capturing complex patterns in high-dimensional, structured and sequential data, and the architectures surveyed in this chapter provide a rich toolbox for both predictive and generative tasks. At the same time, deep models present notable challenges: they are often opaque “black boxes,” sensitive to architecture and optimisation choices, and prone to instability or failure modes such as overfitting, mode collapse in generative settings, or training oscillations that require careful design, regularisation and evaluation. These strengths and limitations motivate a closer examination of generative and sampling techniques that can produce realistic synthetic data and characterise uncertainty. The next chapter therefore turns to two complementary families of methods central to generation and inference adversarial generative models (GANs) and probabilistic sampling approaches based on Markov chains, including the Metropolis–Hastings algorithm and broader MCMC techniques, analysing their principles, practical behaviour and suitability for the synthesis tasks pursued in this thesis.

Chapter 3

Generative Models and Probabilistic Sampling

3.1 Introduction

The rapid growth of machine learning applications in healthcare is often limited by the lack of large, well-labeled clinical datasets. In regulated domains such as medicine, access to patient data is tightly constrained by privacy laws (e.g., GDPR, HIPAA), and the data that are available frequently exhibit high dimensionality, class imbalance, missingness, and complex temporal dependencies. Generative modeling offers a practical path to mitigate these limitations: by learning the underlying data distribution, generative models can produce realistic, privacy-preserving synthetic records that retain the statistical and clinical structure needed for downstream research and algorithm development.

This chapter reviews contemporary generative techniques and probabilistic sampling methods that frame the methodology developed in this thesis. We begin by exploring Generative Adversarial Networks (GANs) as the primary engine for data synthesis. We analyze their architectural adaptations for structured data, ranging from static tabular records (Section 3.2.1) to multivariate clinical time series (Section 3.2.1). Given that Electronic Health

Records (EHRs) are often multi-faceted, we investigate various conditioning mechanisms (Section 3.2.1) and the technical refinements required to mitigate common pitfalls such as mode collapse (Section 3.2.1).

As data utility in healthcare cannot be decoupled from patient confidentiality, we examine the integration of differential privacy frameworks within the generative process (Section 3.2.1). Beyond purely adversarial approaches, we revisit the foundational theory of Markovian models, including Hidden Markov Models (HMMs) and their hybrid variants, to understand how transition dynamics are traditionally modeled (Section 3.2.2). Finally, we discuss Metropolis-Hastings sampling (Section 3.3.1) as a robust probabilistic baseline for distribution refinement. Together, these reviews identify the current limitations in temporal modeling and training stability that our proposed differentiable HMM-GAN framework seeks to address.

3.2 Generative Models

This section surveys two complementary threads in generative modeling. First, we consider Generative Adversarial Networks (GANs), then the Markov chain.

3.2.1 GANs-based methods

GANs for Tabular Data

Tabular data stores information that presents a representation of the patient’s encounter, such as demographic features, aggregated mean or a one-time measurement of vital signs, where each sample has one value for each feature. Deep generative modelling for tabular data poses particular challenges: heterogeneous feature types (discrete, categorical, continuous), as shown in Figure 3.1. Two influential GAN-based approaches that explicitly target these issues are TableGAN and CTGAN. **TableGAN**. TableGAN frames tabular synthesis as an adversarial game in which a generator network learns to produce realistic records and a

Discrete data		
Patient ID	visits per month	Age
2001	5	20
2002	2	37
...	...	...
20008	11	60

Categorical data		
Patient ID	Sex	diabetes
2001	Female	0
2002	Male	1
...	...	...
20008	Female	1

Continuous data		
Patient ID	Platelet	Sodium
2001	300	133.83
2002	186.86	37.66
...	...	...
20008	111.4	N/A

Figure 3.1: Tabular EHR

discriminator distinguishes real from synthetic rows. A recent practical refinement of this idea is a divide-and-conquer strategy proposed by Kang et al., who partition the generation task into subproblems and apply TableGAN within each partition to better preserve logical relationships and structural constraints in the data [Kang et al., 2023]. By decomposing the global generation task, this approach mitigates information loss that arises when a single model attempts to capture heterogeneous patterns across the entire table.

CTGAN (Conditional Tabular GAN). CTGAN adapts the GAN framework to the realities of tabular data by introducing two key mechanisms: a conditional generator that allows the model to explicitly condition on discrete column values, and a training-by-sampling scheme that balances learning across rare and common modes [Xu et al., 2019]. In particular, CTGAN’s mode-specific normalization enables better modelling of non-Gaussian and multimodal continuous features. Subsequent work has applied CTGAN to varied tabular synthesis tasks (e.g., Zhao et al. [Zhao et al., 2021]) and used its conditional sampling to address class imbalance and long-tailed distributions. Other practical studies (e.g., Kuili et al. [Kuili et al., 2024]) combine CTGAN with domain preprocessing (removal of quasi-identifiers, scaling) and unsupervised clustering methods (K-means, DBSCAN, GHMM) to extract latent behavioural patterns before synthesis.

GANs for Time-series data

While tabular representations of electronic health records (EHRs) are useful for describing a patient’s state at a single time point, they do not capture the temporal dynamics intrinsic to clinical processes. Sequential and longitudinal data require models that represent how variables evolve over time and how past events influence future outcomes. Figure 3.2 illustrates this distinction by showing a schematic record (top) alongside its corresponding time-series trace (bottom). Generative modelling for this modality therefore poses additional challenges compared with tabular synthesis: models must learn temporal dependencies, preserve sequence-level statistics (sequence lengths, autocorrelations and transition dynamics), and maintain diversity without collapsing to a few common trajectories. Standard feedforward GANs, which treat each sample as an independent static record, are ill-suited to these requirements unless augmented with sequence-aware components. TimeGAN is a representative architecture explicitly designed for realistic time-series synthesis. This thesis addresses sequential EHRs: the proposed model is specifically designed for longitudinal clinical data, and its fidelity and utility are evaluated comparatively against TimeGAN in the experimental section. **TimeGAN**. integrates ideas from recurrent architectures and adversarial training to synthesise realistic multivariate sequences [Yoon et al., 2019]. The model couples an autoencoder-style reconstruction objective with an adversarial objective and an explicit supervised loss that enforces temporal dynamics. Concretely, TimeGAN’s components typically include an embedder (encoder) that maps real sequences into a latent representation, a recovery (decoder) network that reconstructs sequences from latent codes, a generator that produces latent trajectories from noise, a supervisor that encourages consistency between generated and real latent transitions, and a discriminator that discriminates real from generated sequences (either in data or latent space). Training proceeds in phases reconstruction to learn an informative embedding, supervised learning to align temporal transitions, and adversarial training to improve sample realism so that the final model retains both per-step fidelity and realistic temporal dynamics. preventing mode collapse persists.table 3.1 shows

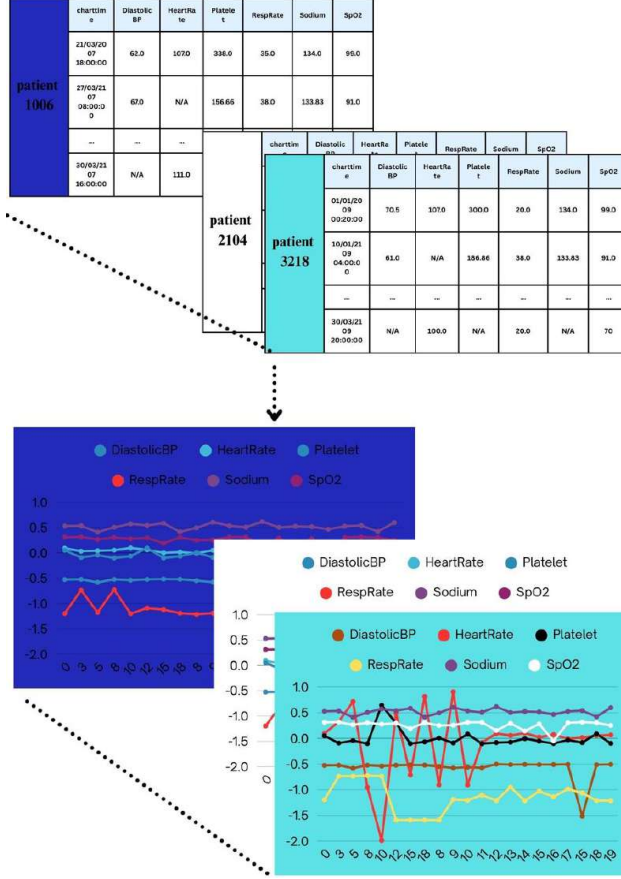


Figure 3.2: Time series EHR

the Fundamental aspects of GAN-based approaches for synthetic EHR time series generation

Conditioning Mechanisms in GANs for EHRs

To enhance the realism of synthetic EHRs, various conditioning strategies have been explored. Esteban et al[Esteban et al., 2017] were motivated by the fact that image-based GANs cannot natively model continuous, temporally structured measurements nor integrate cohort metadata. They therefore propose RCGAN, in which LSTM-based generator and discriminator both ingest, at each time step, a noise vector, the prior hidden state, and a static conditioning vector encoding patient subgroup labels to produce and evaluate coherent multivariate trajectories.

To address the challenges of noise and irregular sampling in time series augmentation,

Ramponi et al. [Ramponi et al., 2018] propose TCGAN, a conditional GAN framework that explicitly incorporates timestamp information as part of the conditioning signal. In TCGAN, the generator network takes as input both a random noise vector and a sequence of real observation times encoded via positional embeddings to produce synthetic values at those exact timestamps, while the discriminator receives the same time–value pairs and learns to distinguish real from generated observations. By training adversarially under the standard GAN objective, TCGAN learns to model the true underlying temporal patterns despite irregular gaps and measurement noise.

Foomani et al. [Foomani et al., 2022] tackle the problem of prognostic modeling for chronic wounds by synthesizing multivariate time series of wound related factors directly from raw EMR data using a GAN framework. Their generator and discriminator are both built on gated recurrent units (GRUs), with the generator conditioned on healing status (healed vs. not healed) to produce temporally coherent trajectories over the healing period, while the discriminator receives both real and synthetic sequences alongside the same static features to enforce fidelity

Lu et al. [Lu et al., 2023] extend conditional GANs to handle multi-label clinical time series, where each patient may present several interdependent diagnoses and interventions across time. Their model (MTGAN) augments a standard generator built from stacked LSTM layers with a multi-label embedding module that projects the patient’s label vector into a dense representation at every time step. The discriminator receives both the time-series signal and the corresponding label embeddings, and a label-consistency loss penalizes generated trajectories whose implied label sequences—when evaluated by a pretrained multi-label classifier—do not match the conditioning labels.

Karami et al. [Karami et al., 2025] propose TimeHR, an image-based approach to generate EHR time series by leveraging advances in visual GAN architectures. They map each patient’s multivariate, irregularly sampled time series into a 2D image: variables are assigned to spatial positions while timestamps run along one axis, and missing values are encoded as

masked pixels. A convolutional GAN generates synthetic images, which are decoded back into time-series form by a learned inversion network; the discriminator operates in the image domain to enforce global consistency. This formulation permits the model to exploit powerful convolutional priors while accommodating irregular sampling and structured missingness. Together, these works chart a clear progression of conditioning mechanisms from static cohort labels through hybrid losses and timestamp embeddings to multi label vectors and image based representations

Table 3.1: Fundamental aspects of GAN-based approaches for synthetic EHR time series generation

Methods	Area of Concern	Type of GAN	Noise Used	Condition Used (if any)	Time-series
RC-GAN [Esteban et al., 2017]	Medical time-series	CGAN	Random noise	Static patient class labels	Yes
TimeGAN [Yoon et al., 2019]	Preserving temporal dynamics	autoencoder GAN	Random noise	/	Yes
T-CGAN [Ramponi et al., 2018]	Handling irregular timestamps	CGAN	Random noise	Real observation timestamps	Yes
EMR-TCWGAN [Foomani et al., 2022]	EMR generation	CWGAN-GP	Random noise	Healing Status (Healed vs. Not Healed)	Yes
MTGAN [Lu et al., 2023]	Multi-label clinical time series generation	CGAN	Random noise	Multi-label embedding vector	Yes
TimeHR [Karami et al., 2025]	Modeling irregularly sampled data	Conv-GAN	Random noise	/	Yes
Zhang et al. [Zhang et al., 2020]	EMR simulation	CWGAN	Random noise	patient class labels	No
Faisal et al. [Faisal et al., 2022]	Privacy-preserving synthetic radiology	WGAN-GP	Differential privacy noise	/	No
Sig-Wasserstein [Liao et al., 2024]	time-series generation	CWGAN	Random noise	Signature features as implicit conditioning	Yes
Marchesi et al. [Marchesi et al., 2022]	time-series generation	CWGAN-GP	Random noise	static patient labels	Yes

Addressing Mode Collapse in GANs

Mode collapse, where the generator produces limited variations of data, remains a significant challenge in GAN-based EHR generation. To mitigate this, researchers have explored advanced GAN architectures which provide smoother loss landscapes and more stable training. Zhang et al [Zhang et al., 2020] build on this by integrating diversified pretraining, domain aware discriminators, and a WGAN GP objective: they first pretrain an autoencoder to capture a broad manifold of EMR patterns, then fine tune a WGAN GP [Arjovsky et al., 2017] with task specific losses to encourage exploration of multiple clinical modes. Faisal et al [Faisal et al., 2022] similarly incorporate the WGAN GP framework but extend it with differential privacy noise in both discriminator and generator updates. By perturbing gradients during the WGAN GP optimization, they regularize the generator away from high density regions of the real data, thus broadening the diversity of synthetic samples while providing formal privacy guarantees. Liao et al [Liao et al., 2024] advance the idea of rich feature spaces by training under a Sig WGAN objective: they replace the standard discriminator loss with a Wasserstein loss in signature coefficient space, matching high order path statistics rather than raw values. This Sig WGAN GP setup ensures that generated time series capture a wide array of temporal behaviors, effectively countering collapse into a few archetypes. Empirical results on synthetic and real clinical datasets confirm that signature based conditioning under the WGAN GP regime yields notably more diverse and realistic sequences compared to RNN based baselines trained with conventional GAN losses. Marchesi et al [Marchesi et al., 2022] focus on under represented cohorts, demonstrating that coupling class balanced minibatches with a WGAN GP objective and diversity promoting loss terms enables conditional GANs to span minority modes more effectively than oversampling. Their fairness centered evaluation shows that the WGAN GP-based generative augmentation not only improves equity for under sampled groups but also maintains overall data realism, highlighting the crucial role of advanced GAN objectives in both diversity and fairness.

Differential-Privacy-based GANs

Differential-privacy-based GANs adapt adversarial generative modelling to provide formal (ϵ, δ) -differential privacy guarantees by introducing carefully calibrated randomness into the training process. One common approach, often referred to as *DP-GAN* [Xie et al., 2018], implements Differentially Private Stochastic Gradient Descent (DP-SGD): per-example (or per-client) gradients are clipped to bound sensitivity and Gaussian noise is added prior to gradient aggregation. The resulting noisy gradient is used to update the discriminator and/or the generator, thereby limiting the influence of any single training example and producing an auditable privacy budget under composition accounting. An alternative strategy, *PATE-GAN* [Jordon et al., 2018], combines the Private Aggregation of Teacher Ensembles (PATE) framework with GAN training: multiple teacher models are trained on disjoint partitions of the sensitive dataset, their outputs are aggregated with calibrated noise to yield a differentially private supervisory signal, and a student model, trained on public or unlabeled data using this noisy supervision, serves as the discriminator that guides the generator. PATE-based designs concentrate much of the privacy cost in a single noisy aggregation step and can avoid per-example gradient perturbation, but they require partitionable data (reducing per-teacher sample sizes), commonly depend on auxiliary public data for effective student training, and increase engineering and model-capacity demands.

3.2.2 Markovian-based methods

3.2.2.1 Markov Chains

A Markov Chain is a mathematical model used to analyze systems that evolve over time based on probabilistic transitions between states. In medical time series data, such as patient vital signs or disease progression, Markov Chains help model dependencies between consecutive observations. The key principle is the **Markov property**, which states that the next state depends only on the current state. A Markov model of order K extends this

by assuming that the state at time t depends on the previous K states: [Douc et al., 2018]

$$P(X_t \mid X_{t-1}, X_{t-2}, \dots, X_1) \approx P(X_t \mid X_{t-1}, \dots, X_{t-K}) \quad (3.1)$$

where X is the state variable and t represents the time step.

These models are typically represented as a set of states with transition probabilities that define how the system evolves over time. In medical applications, first-order Markov models ($K = 1$) are often used. Markov Chains also form the basis for more advanced models like Hidden Markov Models.

3.2.2.2 Hidden Markov Model (HMM)

A Hidden Markov Model (HMM) is an extension of a Markov model where the sequence of states cannot be directly observed. Instead, only the symbols emitted by these states are observed. HMMs are doubly stochastic processes, where observations are generated as a probabilistic function of hidden states, and state transitions follow a Markov process. [Rabiner and Juang, 2003]

Core Components

- **Latent States (Z):** The unobserved states of the system

$$Z = \{z_1, z_2, \dots, z_N\} \quad (3.2)$$

- **Observations (X):** The observed outputs generated by the states

$$X = \{x_1, x_2, \dots, x_T\} \quad (3.3)$$

- **Transition Matrix (A):** Defines the probability of moving between states:

$$A_{ij} = P(z_t = j \mid z_{t-1} = i) \quad (3.4)$$

- **Emission Probabilities (B):** Defines the likelihood of an observation given a hidden state:

$$B_j(x_t) = P(x_t \mid z_t = j) \quad (3.5)$$

Key Equations

- **Forward-Backward Algorithm** for state probability estimation:

$$\alpha_t(i) = P(x_1, \dots, x_t, z_t = i \mid \lambda) \quad (3.6)$$

- **Viterbi Decoding** to find the most likely sequence of hidden states:

$$\delta_t(i) = \max_{z_1, \dots, z_{t-1}} P(z_1, \dots, z_t = i, x_1, \dots, x_t \mid \lambda) \quad (3.7)$$

3.2.2.3 Markov Chain and Duration Modeling

To model explicit state durations, Dai et al. [Dai et al., 2017] introduce the Recurrent Hidden Semi Markov Model (RHSMM), where an RNN predicts dwell-time distributions before EM estimates semi Markov parameters. While more expressive than standard HMMs, RHSMM still decouples neural training from transition learning. Yu et al. [Yu et al., 2017] apply a hybrid NN–HMM for automatic story segmentation, using a feed-forward network for emission posteriors and a fixed HMM for temporal smoothing, again without enabling neural influence on transitions. Liu et al. [Liu et al., 2023] propose a hybrid GRU–attention–HMM for ICU mortality and length of stay prediction. A GRU with attention produces time-point saliency weights that guide a fixed HMM transition matrix and a Mixture Density Network emission layer; training proceeds in two stages (neural likelihood then Baum–Welch), with no joint optimization.

3.3 Probabilistic Sampling

Probabilistic sampling is central to both traditional statistics and modern generative work. Neural generators such as GANs are great at producing realistic-looking examples from noise, but they do not by themselves ensure that those examples follow a known probability model or satisfy explicit likelihood constraints. Classical samplers fill that gap: they let us either draw directly from a target density or refine approximate samples so that they better match the intended distribution. That capability matters when we must enforce constraints or honour observed-data likelihoods that a purely neural approach might overlook. Markov Chain Monte Carlo (MCMC) methods[Carlo, 2004] are a classic solution: they construct a Markov chain whose stationary distribution is the target density and therefore produce asymptotically unbiased samples. In this thesis we use Metropolis–Hastings as our main latent-space refinement tool.

3.3.1 Metropolis–Hastings Sampling

Metropolis–Hastings (MH) is a simple but powerful algorithm for sampling from a target density $\pi(x)$ when direct sampling is not practical. Starting from an initial state $x^{(0)}$, the procedure repeats the following steps:

1. **Proposal:** Draw a candidate x^* from a proposal distribution $q(x^* | x^{(t)})$.
2. **Acceptance probability:** Compute α :

$$\alpha(x^{(t)}, x^*) = \min \left(1, \frac{\pi(x^*)q(x^{(t)} | x^*)}{\pi(x^{(t)})q(x^* | x^{(t)})} \right) \quad (3.8)$$

3. **Update:** With probability α , set $x^{(t+1)} = x^*$; otherwise keep $x^{(t+1)} = x^{(t)}$.

With fairly mild conditions on the proposal and the target, the chain $\{x^{(t)}\}$ converges to $\pi(x)$ as $t \rightarrow \infty$. In practice MH is an effective way to nudge samples from a simple prior

toward a complex target: the proposal step explores, and the acceptance step corrects for any mismatch. This combination of flexibility and a principled correction is why MH is a natural choice for the tempered latent refinement described later.

3.4 Conclusion

This chapter surveyed generative modeling and probabilistic sampling with an eye on electronic health records (EHR). We began by identifying data scarcity and privacy rules as major hurdles for healthcare ML, then reviewed how modern generative approaches address those hurdles, especially for longitudinal data where sequence-aware architectures are essential. We examined the trade-offs between producing high-fidelity samples and meeting formal privacy guarantees, and argued that classical, likelihood-aware tools such as MCMC and Metropolis–Hastings still play an important role alongside neural generators. Based on these insights, our framework couples a differentiable HMM with LSTMs so that short-term transitions and long-range patient histories are learned jointly; conditioning HMM transitions on LSTM hidden states relaxes strict Markov assumptions and enables end-to-end training. To reduce mode collapse and increase diversity we apply a tempered Metropolis–Hastings refinement to the latent noise vector before generation, and we stabilize training using a WGAN-GP objective. The result is a system that produces high-fidelity, privacy-conscious synthetic EHR traces that retain realistic temporal dynamics. The next sections detail the model architecture and experimental protocols.

Chapter 4

Towards a Physiologically-Informed GAN

4.1 Introduction

This chapter presents a physiologically informed generative framework that integrates differentiable Hidden Markov Models (HMMs) with adversarial neural networks to synthesize medical time series. The objective is to produce synthetic electronic health records (EHRs) that preserve clinically meaningful state dynamics and longitudinal patient context, enabling downstream research while reducing disclosure risk.

4.2 Physiologically-Informed GAN Framework

The proposed framework proceeds through six principal steps, each designed to address specific challenges inherent to clinical time-series synthesis.

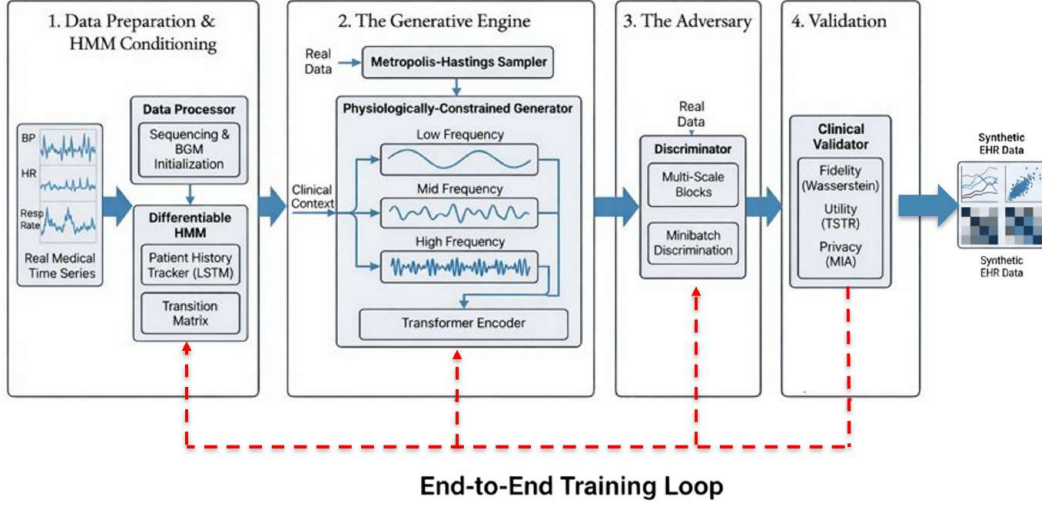


Figure 4.1: Outline of the proposed approach

4.2.1 Step 1: DataProcessor

The DataProcessor converts raw intensive care unit (ICU) time series into a standardized representation for modeling and extracts initial latent structure. The preprocessing pipeline addresses common EHR characteristics:

- **Outlier Handling:** Features are winsorized at the 1st and 99th percentiles to limit the influence of extreme values while retaining clinically relevant trends.
- **Temporal Segmentation:** Irregularly sampled ICU records are segmented into overlapping fixed-length sequences (20 timesteps) using a sliding window. This yields consistent inputs for sequence models and facilitates learning of short-term temporal patterns.
- **Per-Subject Normalization:** A RobustScaler is applied on a per-patient basis to preserve within-subject feature relationships and individual baselines, which is important because physiological ranges vary substantially across patients.
- **State Discovery:** Emission parameters for the HMM are initialized using a Bayesian

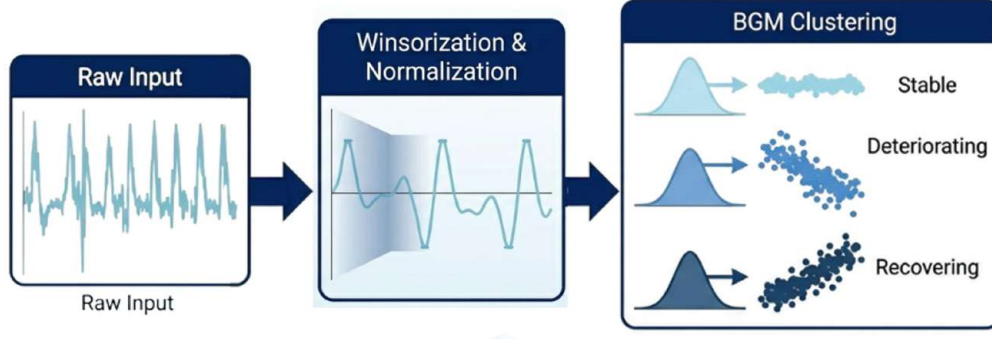


Figure 4.2: Data Processor Step

Gaussian Mixture model rather than arbitrary clustering. This initialization promotes discovery of clinically interpretable latent states (for example, "stable," "deteriorating," "recovering") and provides an interpretable scaffold for subsequent HMM learning.

4.2.2 Step 2: Differentiable Hidden Markov Model

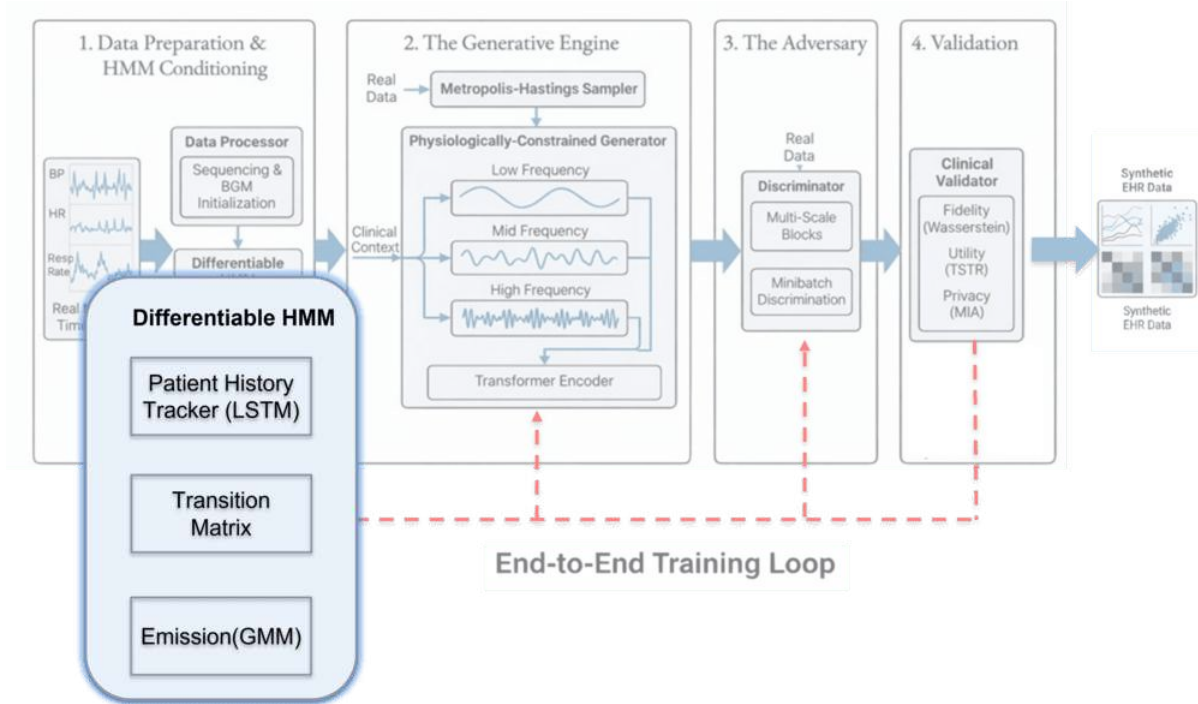


Figure 4.3: Differentiable Hidden Markov Model

The Differentiable HMM is the framework’s core contribution. Unlike conventional HMMs with fixed transition matrices, our differentiable formulation is trainable end-to-end within a neural optimization loop, allowing simultaneous learning of HMM and GAN parameters. The differentiable HMM comprises three modules: the patient history tracker, the state-transition module, and the state-emission module (see Figure 4.3).

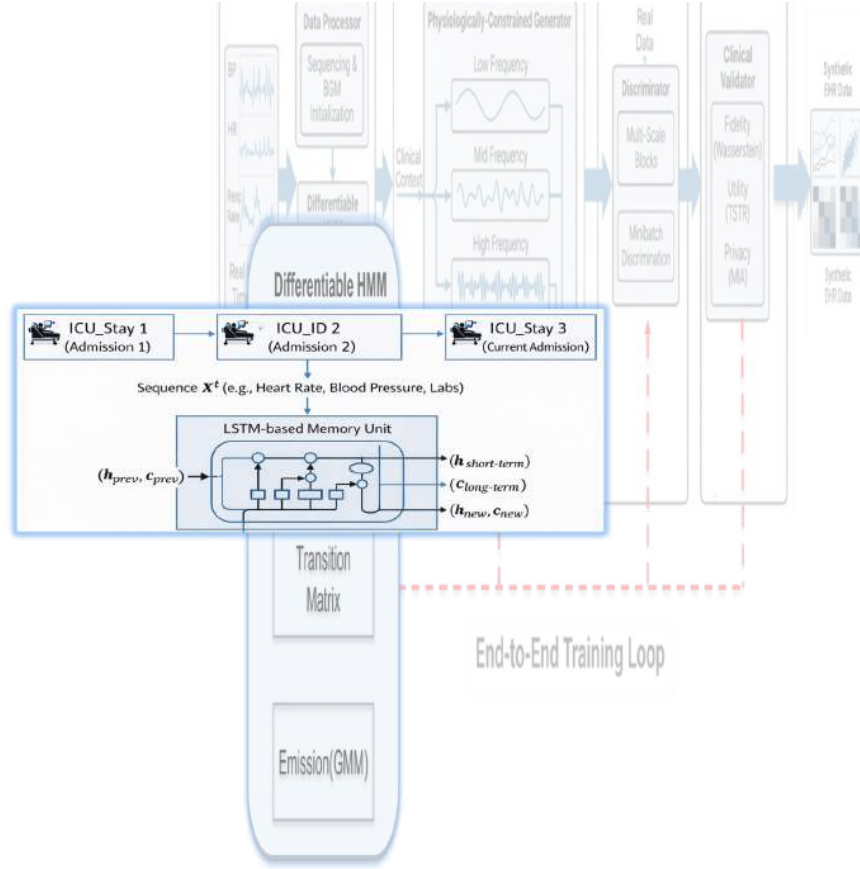


Figure 4.4: Patient History Tracking

Patient History Tracking

Patient status depends not only on immediate observations but also on prior clinical context. We maintain an LSTM-based memory per ICU stay to capture both short-term physiological trends (hidden state h) and longer-term clinical context (cell state c). Given a sequence $X^t = x_1, x_2, \dots, x_T$

$$(h_{new}, c_{new}) = LSTM(X^t, (h_{prev}, c_{prev})) \quad (4.1)$$

where $h_{\text{prev}}, c_{\text{prev}}$ are the prior LSTM states for the ICU_{ID} and $h_{\text{new}}, c_{\text{new}}$ are the updated states after processing X^t . This continuous memory conditions downstream components on an evolving patient representation rather than a sequence-independent snapshot.

Compute State Transition Probabilities

We estimate the probability of transitioning between latent states by conditioning on both the patient’s long-term history and the immediately preceding state. First, the PatientHistoryTracker retrieves the prior LSTM hidden and cell states $(h_{\text{prev}}, c_{\text{prev}})$, which encode the patient’s memory across sequences. The incoming observation sequence $X_{1:t}$ is processed by a bidirectional LSTM to produce a history embedding h_{hist} , summarizing temporal dynamics. Meanwhile, the discrete previous state s_{t-1} is represented in two complementary forms: (1) a one-hot vector $e_{\text{one-hot}}(s_{t-1})$, providing an explicit identity signal, and (2) a learned embedding $e_{\text{emb}}(s_{t-1})$, which captures semantic similarities among states (e.g., “stable” vs. “recovering”). These three vectors are concatenated, and fed into a transition network $\mathcal{F}_{\text{trans}}(\cdot)$, implemented as an LSTM followed by a softmax layer (Figure 4.5), to yield dynamic transition probabilities:

$$P(s_t \mid s_{t-1}, h_{\text{hist}}) = \text{softmax}(f_{\text{trans}}(e_{\text{one-hot}}(s_{t-1}) \parallel e_{\text{emb}}(s_{t-1}))) \quad (4.2)$$

By replacing a static $k \times k$ matrix with a neural network conditioned on h_{hist} and s_{t-1} , the model naturally captures higher-order, non-Markovian influences such as gradual improvement or decline. This directly addresses a significant limitation of traditional Markov models, which typically assume that the next state depends only on the immediate previous state, failing to capture the complex, multi-factorial progression of clinical conditions.

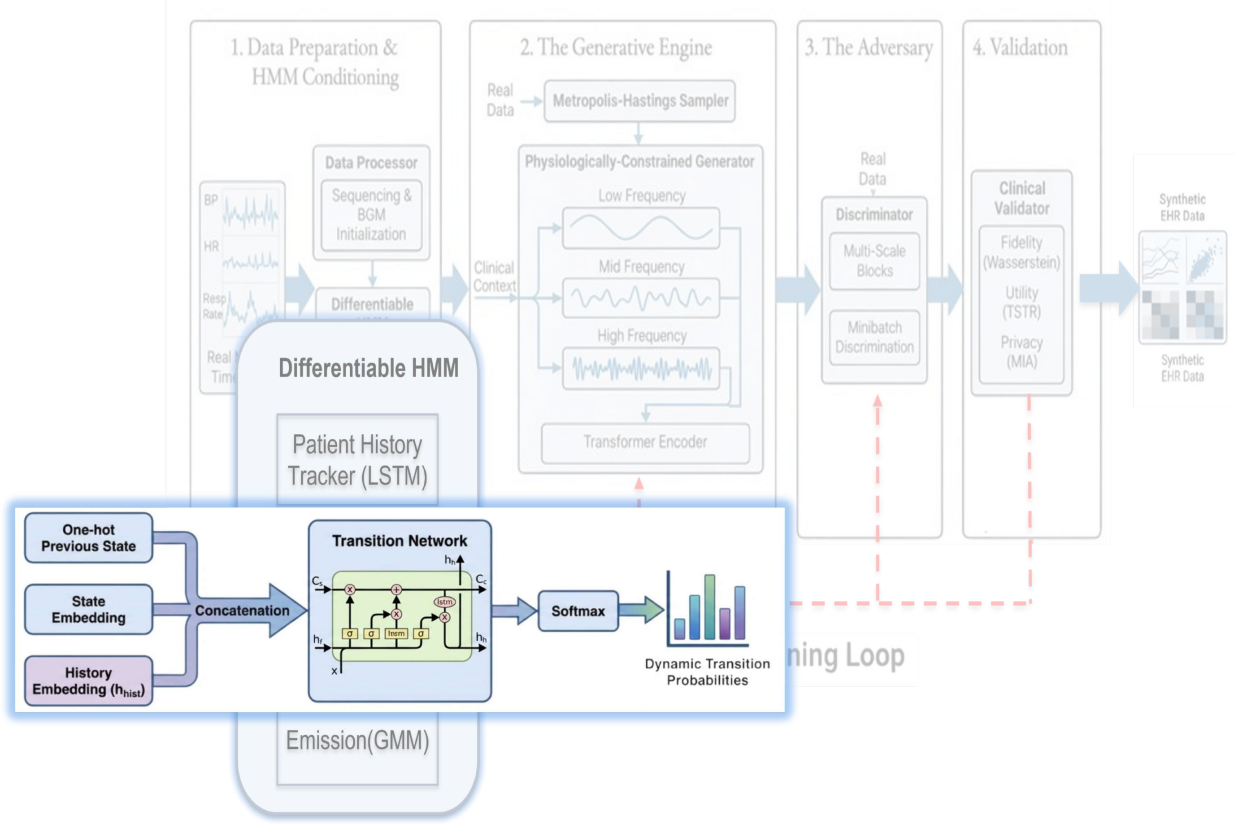


Figure 4.5: State Transition Probabilities

Compute Emission Probabilities

We calculate the likelihood of observing a patient's time-series data $X_{1:t}$ under each state by modeling the observation distribution as a Gaussian mixture. At each time step t and for each hidden state s , the model assumes that the observation x_t arises from one of K Multivariate Normal (MVN) components

$$x_t \sim \sum_{k=1}^K \pi_{s,k}(h_t) \mathcal{N}(\mu_{s,k}, \Sigma_{s,k}) \quad (4.3)$$

Here, $\{\mu_{s,k}, \Sigma_{s,k}\}$ are the emission parameters (means and Cholesky covariance matrices) for component k of state s , initialized by fitting a Bayesian Gaussian Mixture to the data. A dedicated mixture-selector network ingests the current features h_t and outputs dynamic mixture weights $\{\pi_{s,k}(h_t)\}_{k=1}^K$, representing the probability that x_t was generated by each MVN

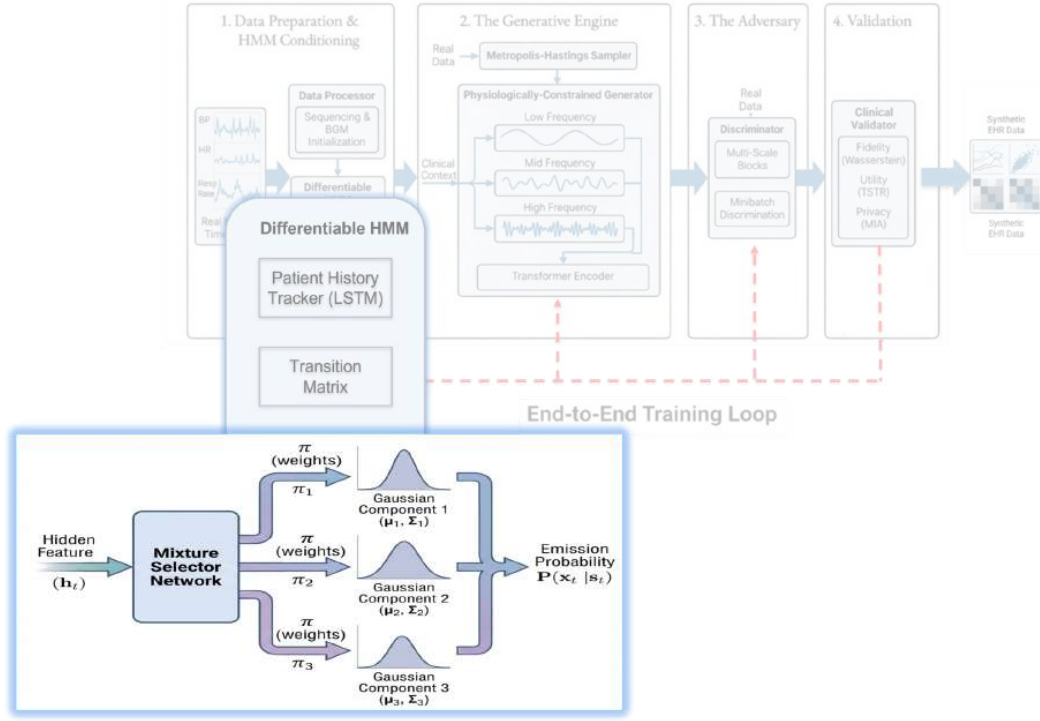


Figure 4.6: State Emission Probabilities

component. The component log-densities $\log p(x_t|s, k)$ are aggregated via a numerically stable log-sum-exp:

$$\log p(x_t|s) = \log \sum_{k=1}^K \pi_{s,k}(h_t) \exp(p(x_t|s, k)) \quad (4.4)$$

and then normalized across states with a softmax to yield the posterior probability of each hidden state at time t (Figure 4.6). This approach is vital for capturing the natural variability and heterogeneity within clinical states. For instance, a "stable" patient state might still encompass a range of normal vital sign readings, and a single Gaussian distribution would be insufficient to model this inherent diversity.

HMM Conditioner

The HMMConditioner coordinates patient-specific differentiable HMM instances. During training it supplies the generator and discriminator with patient history (h_{hist}) and the HMM-derived state probabilities from real data. Conditioning on these HMM signals grounds

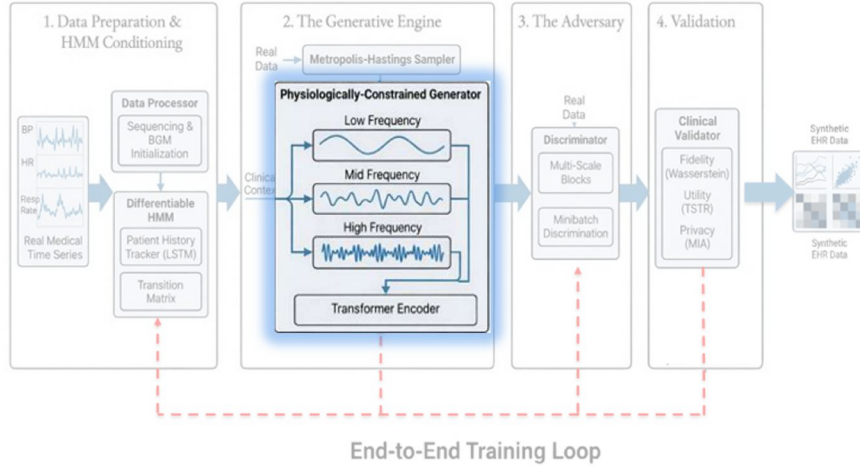


Figure 4.7: Generator

generation in physiologically plausible state trajectories and aligns synthetic outputs with plausible clinical transitions.

4.2.3 Step 3: The Generator

The Generator maps refined latent inputs into multivariate time series while incorporating HMM-derived conditioning and Metropolis–Hastings Sampling.

4.2.3.1 Tempered Metropolis–Hastings Sampling

Clinical data occupy a complex, high-dimensional latent manifold with many distinct modes corresponding to different patient subpopulations and event trajectories. To avoid mode collapse and to encourage coverage of rare but clinically important modes, we refine latent samples using a tempered Metropolis–Hastings (MH) procedure rather than sampling naïvely from an isotropic Gaussian.

During training, a gradient-preserving diversity sampler injects multi-scale perturbations into latent batches to promote variability without breaking backpropagation. At inference,

the tempered MH routine runs multiple parallel chains with adaptive temperatures and proposal scales. Chains incorporate momentum terms and adaptive proposal scales that depend on each chain’s radial distance from the origin; periodic repulsion forces between nearby chains and acceptance-rate-driven temperature adjustments further encourage broad exploration. Accepted proposals are stored in a global buffer for subsequent selection. Unlike the original MH which begins from a single random seed, employs a fixed Gaussian proposal and unit temperature our approach simultaneously launches dozens of parallel chains seeded from a stratified, data-driven pool; replaces the crude log-density with a Structured Latent Scoring Function (SLS)(Section 4.2.3) that fuses prior regularization, smoothness and sparsity rewards, attractor pulls, and repulsion penalties; dynamically adapts each chain’s proposal scale and temperature based on radial position and acceptance rate; and explicitly enforces diversity through periodic inter-chain repulsion and a cluster-based final selection of representatives. The complete algorithmic specification is given in Algorithm 1.

A. Training-Time Sampling To avoid disrupting the end-to-end differentiable computation graph, stochastic perturbations are introduced in a manner that retains gradient connectivity. Rather than performing discrete accept/reject steps, the latent vector is divided into thirds. The first third receives small noise (exploitation), the second third receives medium noise (balanced exploration), and the last third receives larger noise (exploration). This ensures that the generated samples remain diverse while still allowing for gradient updates

B. Inference-Time Sampling In the absence of a training objective, the generator reverts to the full tempered Metropolis–Hastings sampler organized into three stages

B.1 Stage 1: Evaluate and Select Diverse Starting Points

A pool of up to 100 candidate latent points is first generated via multiple strategies (random normal, data-driven based on empirical statistics and structured functions) and evaluated based on their log-probabilities using SLS function. A subset of these candidates

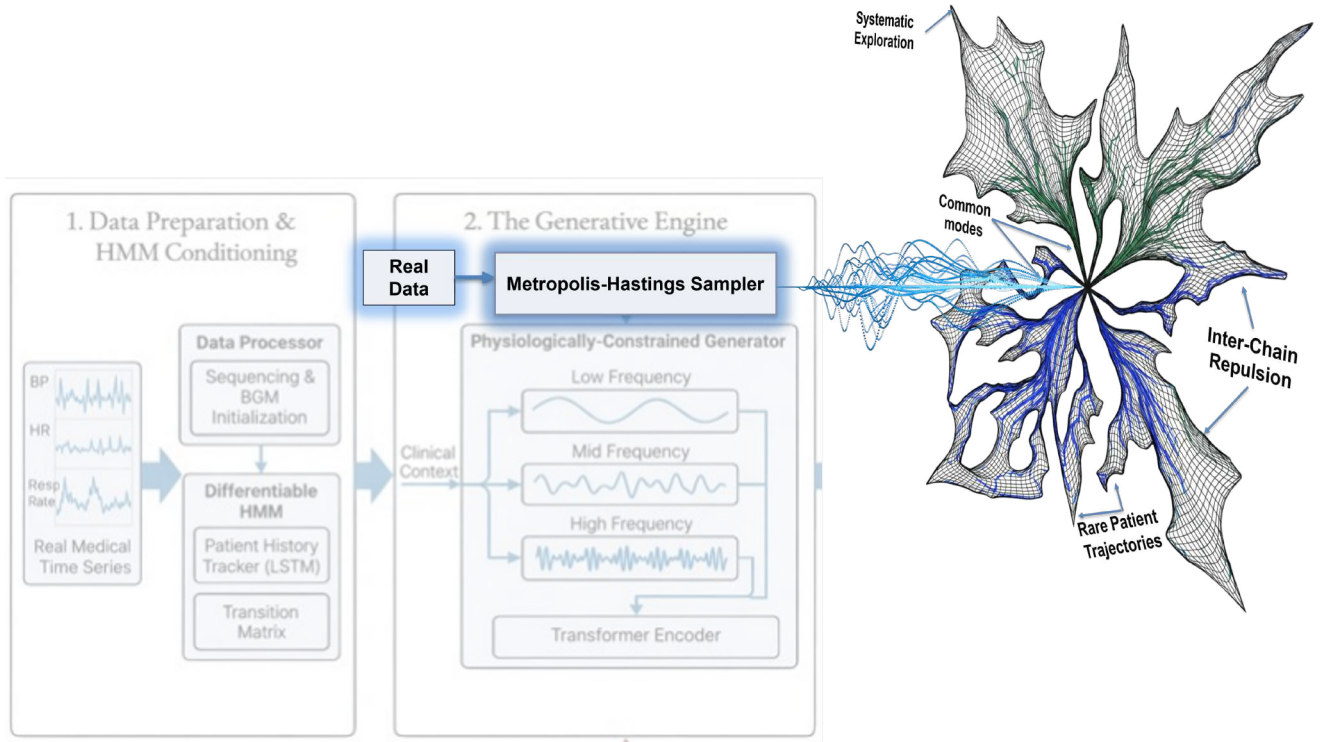
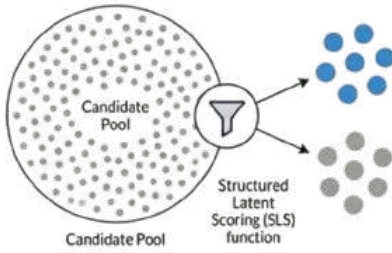
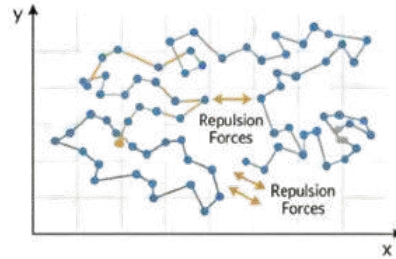


Figure 4.8: Metropolis-Hastings Sampling

Stage 1: Evaluate and Select Diverse Starting Points



Stage 2: Run Parallel Chains with Diversity Enforcement



Stage 3: Select the Final Diverse Subset

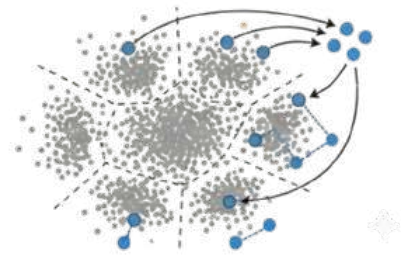


Figure 4.9: Metropolis-Hastings stages

(75 %top-scoring, 25 % random) initializes up to 40 parallel chains

B.2 Stage 2: Run Parallel Chains with Diversity Enforcement At inference we run multiple MCMC chains in parallel [Ramos et al., 2025], so the latent space is explored more comprehensively. Each chain is seeded differently during preprocessing and carries its own adaptive temperature, proposal scale, and momentum term [Qian, 1999]; sampling proceeds in fixed-length rounds, which makes periodic parameter updates and diversity checks practi-

cal. Proposals are formed by adding Gaussian noise whose scale adapts to the chain’s radial distance from the origin: we use a larger base scale near the centre to encourage outward probing, reduce the scale and add an inward tendency for chains that have drifted far out, and adopt a balanced scale in between. A fraction of the previous update (momentum) is included in each proposal so chains can keep a consistent trajectory rather than jumping aimlessly. Accepted proposals chosen via the tempered MH criterion replace the chain state, update momentum, and are appended to a global buffer. Every fixed interval, we compute pairwise distances between chains, identify the closest quarter of pairs, and apply mild repulsive adjustments (strongest near the centre) to preserve spatial spread[Hamo et al., 2016]. Simultaneously, we tune each chain’s temperature and proposal scale from its acceptance history and radial position: low acceptance increases temperature and scale to boost exploration, while high acceptance tightens them to favour exploitation. We continuously monitor the mean pairwise distance among newly accepted samples and nudge parameters if diversity drops below a threshold; clinically, this helps the sampler reveal distinct patient trajectory modes (for example, rapid decline versus slow chronic progression) so the generator captures both common patterns and rarer outcomes.

B.3 Stage 3: Select the Final Diverse Subset Once accepted samples accumulate in the global buffer we form the final batch for generation with diversity and representativeness in mind. If the buffer is undersized we pad it with scaled random vectors; if it is oversized we normalize candidates and cluster them with K-Means (up to 30 clusters)[Hamerly and Elkan, 2003]. From each cluster we select representatives using a hybrid of uniform sampling and farthest-point sampling (FPS)[Moenning and Dodgson, 2003] so both typical and extreme clinical examples are kept. The selected batch of latent vectors thus balances common cases and outliers and is ready for generation into synthetic EHR traces.

Algorithm 1 Tempered Metropolis–Hastings

Require: SLS, z_0 , N , σ **Ensure:** Z^*

```
// Training-Time Sampling
1: if  $z_0.requires\_grad$  then
2:   split  $z_0$  into three equal parts
3:   add small / medium / large Gaussian noise
4:   return perturbed batch
5: end if
// Inference-Time Sampling
// Stage 1: Initialization & Seeding
6: SLS( $\mathbf{z}$ ) for all  $\mathbf{z} \in \mathcal{P}$ 
7: Select the top 75% by score and 25% uniformly at random to form  $\mathcal{S}_0$ 
// Stage 2: Parallel MH Sampling
8: Initialize 40 chains from  $\mathcal{S}_0$  with stratified  $(T_c, \sigma_c)$ 
9: Initialize momentum  $\mathbf{m}_c = \mathbf{0}$  for each chain
10: for  $t = 1$  to  $T$  do
11:   for each chain  $c$  do
12:     propose  $z_{\text{prop}}$  via adaptive Gaussian +  $m_c$ 
13:      $\alpha \leftarrow \exp((\text{SLS}(z_{\text{prop}}) - \text{SLS}(z_c))/T_c)$ 
14:     if Uniform(0, 1) <  $\alpha$  then
15:       chain[ $c$ ]  $\leftarrow z_{\text{prop}}$ , update  $m_c$ , collect  $z_{\text{prop}}$ 
16:     end if
17:   end for
18:   if  $N = 50$  then
19:     apply inter-chain repulsion; adapt  $T_c, \sigma_c$ 
20:   end if
21: end for
// Stage 3: Final Selection
22: Apply KMeans clustering
23: select from each cluster half random, half farthest-point
24:  $Z^* \leftarrow A$ 
25: return  $Z^*$ 
```

C. Structured Latent Scoring Function

Latent proposals are evaluated by a Structured Latent Scoring function $S(z)$ that informs acceptance decisions and adaptive chain behavior. $S(z)$ is a weighted combination of components that promote fidelity, smoothness, sparsity, attraction to predefined regions, and inter-sample repulsion:

1. **Standard normal prior** (weight = 0.5): regularizes values toward the origin via

$$-\frac{1}{2} \sum_i z_i^2.$$

2. **Smoothness reward** (weight = 0.2): promotes correlation between adjacent latent dimensions by

$$\sum_{i=1}^{d-1} (\hat{z}_i \hat{z}_{i+1}),$$

where $\hat{z} = z / \|z\|_2$.

3. **Sparsity reward** (weight = 0.2): encourages distinct and sparse patterns via randomly masking 20% of dimensions and counting the entries in $\hat{z}$ with $|\hat{z}_i| < 0.1$.

4. **Attractor reward** (weight = 0.5): draws samples toward one of eight predetermined attraction points $\{a_j\}_{j=1}^8$ using

$$\exp(-\min_j \|z - a_j\|).$$

5. **Repulsion term** (weight = -0.5): computes the squared pairwise correlations across the normalized batch $\{\hat{z}^{(k)}\}$, removes self-similarities, and penalizes high inter-sample similarity to prevent mode collapse.

The aggregate score

$$\begin{aligned} S(z) = & 0.5 S_{\text{prior}}(z) + 0.2 S_{\text{smooth}}(z) \\ & + 0.2 S_{\text{sparse}}(z) + 0.5 S_{\text{attr}}(z) \\ & - 0.5 S_{\text{repel}}(z) \end{aligned} \tag{4.5}$$

thus drives both acceptance decisions and adaptive chain dynamics in our enhanced Metropolis–Hastings framework.

The weights of the Metropolis–Hastings (MH) scoring function were selected using a theoretically motivated initialization strategy. The resulting configuration is summarized in Table [4.1](#).

Table 4.1: Theoretically motivated initialization of MH scoring function weights.

Weight	Component	Rationale
0.5	Prior	Moderate prior regularization that prevents latent collapse while preserving exploration.
0.2	Smoothness	Lower weight to avoid excessive smoothing of temporal physiological signals.
0.2	Sparsity	Balanced with smoothness to encourage meaningful latent representations.
0.5	Attractor	Equal to the prior weight to ensure that multi-modal coverage is as important as prior conformity.
-0.5	Repulsion	Chosen symmetrically with the attractor term to balance attractive and repulsive forces, preventing both mode collapse and excessive dispersion.

The symmetry between the attractor (0.5) and repulsion (-0.5) terms is intentional. This design creates a balanced energy landscape in which latent samples are simultaneously attracted toward distinct high-probability modes while being repelled from one another, thereby maintaining diversity. This behavior is analogous to the attractive-repulsive dynamics employed in physics-inspired sampling methods, enabling effective exploration of the multimodal latent space without sacrificing stability.

4.2.3.2 Conditioning Integration

Generator outputs are tightly coupled to HMM-derived state information to ensure medical plausibility:

- **State-Guided Generation:** The generator is conditioned on HMM emission probabilities, ensuring the synthesized signals accurately reflect the inferred clinical state.

- **KL-Divergence for State Alignment:** A Kullback–Leibler divergence penalty is applied to discrepancies between the HMM state distributions of real and synthetic data. This encourages the strict alignment of state proportions and transition dynamics across both datasets.

4.2.3.3 Sequence Logic & Pattern Generation

Multi-Frequency Paths (Low, Mid, High): To capture the heterogeneous temporal scales inherent in physiological signals, the generator employs parallel 1D convolutional branches optimized for multi-frequency paths:

- **Low-Frequency:** Utilizes a kernel size $k = 7$ with padding $p = 3$: captures slow trends and long-term drift.
- **Mid-Frequency:** Utilizes a kernel size $k = 3$ with padding $p = 1$: models standard temporal dependencies and common rhythms.
- **High-Frequency:** Utilizes a kernel size $k = 1$: captures point-wise spikes and rapid fluctuations (e.g., transient heart-rate spikes).

The choice of kernels are grounded in ICU clinical practice where vital signs exhibit altered circadian periodicity [Davidson et al., 2020], medication effects follow rapid response-and-decay pharmacokinetic profiles [De Bels et al., 2023, University of California, San Francisco, nd], and acute clinical events manifest within single hourly observational intervals [Johnson et al., 2016].

Transformer Encoder: A Transformer encoder subsequently fuses these multi-scale features to model long-range dependencies and to produce coherent, non-fragmented sequences across the entire time window.

4.2.4 Step 4: The Discriminator

The Discriminator evaluates both local signal fidelity and global sequence coherence, using the HMMConditioner’s contextual signals. Inputs are the raw time series together with

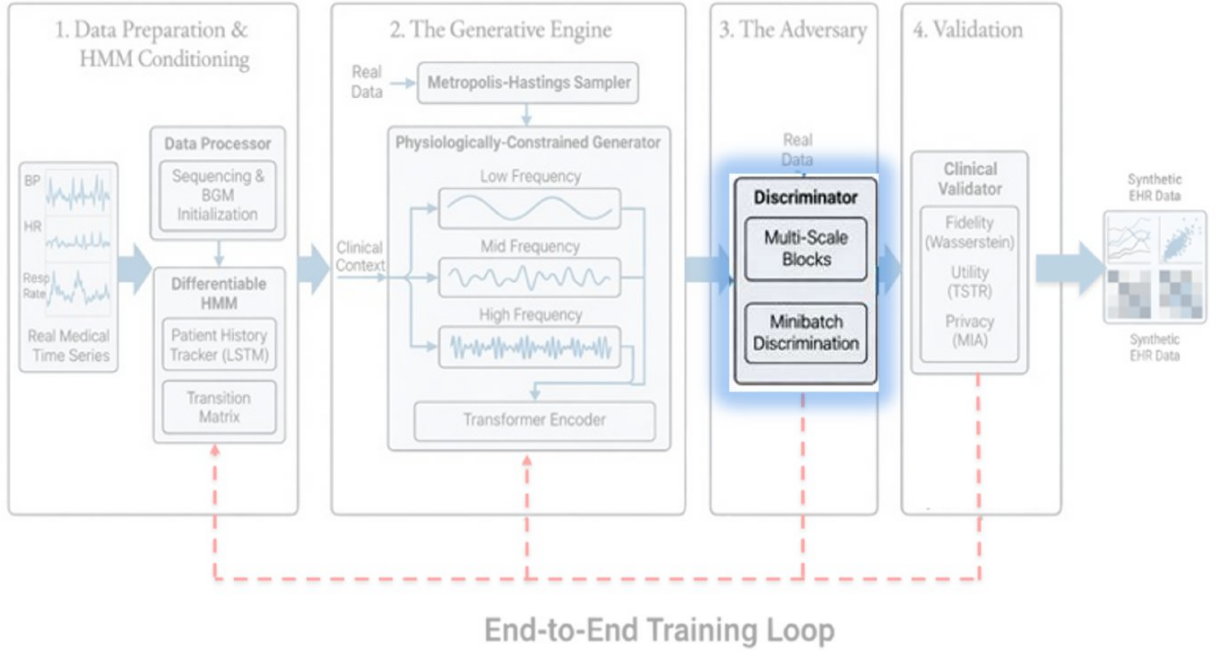


Figure 4.10: Discriminator Step

a conditioning vector containing HMM emission probabilities and patient-history features. Architectural elements include multi-scale temporal blocks to extract features at different temporal resolutions, a minibatch discrimination module to detect and penalize lack of diversity, and a final classifier head that outputs a scalar adversarial score. Stability and robustness are promoted through a WGAN-GP objective (gradient penalty) and spectral normalization applied to convolutional and linear layers.

4.2.5 Step 5: WGAN-GP Trainer

The WGANGPTrainer orchestrates adversarial training and coordinates joint optimization of the GAN and embedded HMM modules. Its primary objective is to jointly optimize the Generator to produce high-fidelity synthetic medical time-series and the Discriminator to reliably distinguish real from generated sequences while leveraging patient-specific context

provided by the HMMConditioner.

Upon instantiation, the trainer initializes the following components: the Generator and Discriminator networks; the HMMConditioner (wrapping a DifferentiableHMM) for extracting patient history and real-data state probabilities; the training dataset; and a configuration dictionary specifying hyperparameters such as loss weights (e.g., gradient-penalty weight, KL-divergence weight), and Metropolis–Hastings sampling settings (temperature, sample factor). It also constructs separate optimizers for the Generator, Discriminator, and HMM, and employs a dynamic learning-rate. A ClinicalValidator periodically assesses synthetic-data quality and privacy metrics, while a gradient scaler enables Automatic Mixed Precision for computational efficiency.

At each train step, WGANPTrainer executes a two-phase update. In the Discriminator phase, real and context-conditioned data are forwarded through the HMMConditioner to obtain patient history and state probabilities; the Discriminator then scores both real samples and Generator produced fakes, and a Wasserstein gradient penalty enforces the Lipschitz constraint. The combined Discriminator loss is backpropagated with gradient clipping, followed by optimizer and scheduler steps. In the Generator phase, the Generator produces synthetic sequences along with a KL-divergence term aligning its latent distribution to the HMM’s state distribution. A feature-wise Wasserstein distance further measures distributional similarity, and the Generator loss combines this distance, the negative Discriminator score, and the KL term. These gradients are applied to both the Generator and HMM parameters via their respective optimizers, with schedulers advanced thereafter.

Throughout training, WGANPTrainer maintains a loss log for each component and dynamically adapts the HMM learning rate, scaling it up when certain latent states are underutilized to prevent mode collapse.

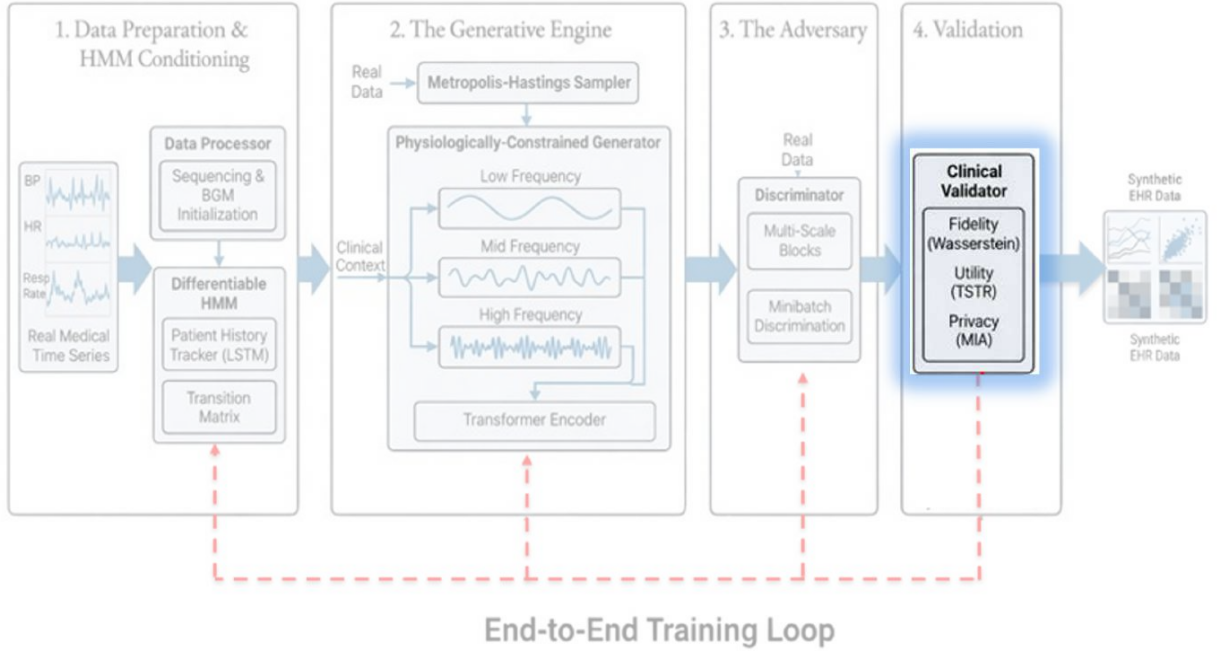


Figure 4.11: Clinical Validation

4.2.6 Step 6: Clinical Validation Step

The Clinical Validator acts as the final quality-control gate for model training, ensuring the integrity of generated EHR data by evaluating its fidelity, utility, and privacy. Fidelity assessment quantifies how closely the synthetic distribution adheres to the real data manifold via a PRDC suite (Precision, Recall, Density, Coverage), feature-wise distributional distances (Wasserstein) and information-theoretic divergence (Jensen–Shannon), and temporal fidelity checks using autocorrelation (ACF) matching to ensure clinically realistic trajectories over time. Utility is evaluated under a Train-on-Synthetic, Test-on-Real (TSTR) protocol using predictive metrics such as AUROC and AUPRC on clinically relevant tasks. Privacy assessment combines global similarity tests and adversarial diagnostics to quantify disclosure risk, including a JSD-based membership-inference assessment (JSD-MIA) and a Nearest-Neighbour Adversarial Accuracy (NNAA) risk score. The validation results act as

a feedback loop for the WGANGPTrainer. Training is terminated by an early-stopping rule once specific thresholds are met. This ensures the final selected model provides the optimal balance between statistical fidelity and diversity.

4.3 Conclusion

In this chapter, we presented a physiologically-informed framework for synthesizing medical time series that integrates a differentiable hidden Markov model (dHMM) within a WGAN-GP, complemented by a tempered Metropolis-Hastings sampler, a multi-scale generator, and a discriminator. We began by outlining the overall architecture and detailing the principal components, demonstrating how they together enhance clinical realism. We then described the WGANGPTrainer training protocol and the Clinical Validator. To validate our contribution, the following chapter details the experiments conducted to evaluate fidelity, utility, and privacy.

Chapter 5

Implementation and Evaluation

5.1 Introduction

After presenting the model architecture and the conditioning strategy, this chapter details the experimental procedure we adopt to validate our framework. We describe the different implementation steps and evaluation scenarios used to assess both the quality of the generated datasets and the privacy properties offered by our framework. The concrete steps of our experimental protocol are as follows:

- We demonstrate the contribution of the proposed framework to data quality by comparing multiple fidelity and utility metrics against centralized baselines (TimeGAN and TimEHR). This comparison highlights how well the synthetic sequences reproduce real patient dynamics and support downstream prediction tasks.
- To characterize performance and privacy trade-offs, we compare several implementation scenarios and ablation variants. For each scenario, we report fidelity (PRDC), utility (TSTR AUROC/AUPRC) and privacy metrics (JSD_{MIA} and NNAA), thus quantifying the practical cost or benefit of each design choice.

This experimental design ensures a systematic, reproducible assessment of how architectural components and training choices affect the realism, usefulness, and privacy risk of the

synthetic EHRs.

5.2 Dataset Description

Our framework is evaluated on the publicly available MIMIC-III clinical database [Johnson et al., 2016]. This database contains de-identified records of over 40,000 ICU patients admitted between 2001 and 2012, and it has become a standard benchmark for time-series modeling in healthcare [Karami et al., 2025, Yoon et al., 2023]. From this extensive corpus, a specific cohort of 100 patients with multiple ICU stays is extracted. The study focuses on six routinely monitored physiologic signals: Diastolic Blood Pressure (DiastolicBP), Heart Rate (HeartRate), Platelet Count (Platelet), Respiratory Rate (RespRate), Sodium Concentration (Sodium), and Oxygen Saturation (SpO2). These features are chosen because they represent common vital signs crucial for patient state assessment, enabling the model to learn and reproduce realistic physiological patterns and trajectories. Each patient’s trajectory is represented as a fixed-length sequence of 20 time steps. To benchmark our approach, we compare against two state-of-the-art generative models for medical time-series data TimeGAN [Yoon et al., 2019] and TimEHR [Karami et al., 2025]. We adapt their open-source implementations by constraining inputs to our six clinical features and 20-step window, thereby facilitating a one-to-one comparison of generated patient trajectories. Our hyperparameters were selected to balance three priorities: clinical interpretability, training stability, and sufficient model capacity. In particular, we set the DHMM to three hidden states, following the clinically grounded abstraction introduced in [Schmidt and Wiil, 2015], which models patient trajectories using a compact set of interpretable physiological states. The Bayesian Gaussian Mixture (BGM) employs a Dirichlet Process prior, which automatically performs model selection by pruning unused components. the hidden dimension of 32 provides enough capacity to capture temporal structure without overfitting noisy medical signals. We used 400 epochs and batch size 32 because adversarial training on limited healthcare time series

typically requires a longer optimization horizon, while a smaller batch size adds regularization and helps reduce memorization. The learning rates were chosen under a two-time-scale update rule to stabilize GAN training, with a more conservative generator update to avoid mode collapse and a slightly larger HMM rate to let the latent dynamics adapt more quickly. Likewise, the scheduler choices were made to support stable convergence, the loss weights were tuned to preserve adversarial learning while keeping the latent space structured, and the temperature and gradient clipping were selected to favour realistic sampling and numerical stability throughout training. Summary of key preprocessing hyperparameters is provided in Table 5.1. All experiments were conducted on an 11th Gen Intel(R) Core(TM) i7-1165G7 CPU

Table 5.1: Summary of Key Hyperparameters

Category	Parameter	Value
Differentiable HMM	$n_{\text{components}}$	3 (stable, deteriorating, recovering)
	hidden_dim	32
Training Configuration	epochs	400
	batch_size	32
Learning Rates	Generator (lr_G)	5×10^{-8}
	Discriminator (lr_D)	1×10^{-6}
	HMM (RMSprop)	1×10^{-5}
Schedulers	Generator & HMM	OneCycleLR (warm-up + cosine decay)
	Discriminator	CyclicLR
Loss Weights	Gradient penalty (λ_{gp})	1×10^{-4}
	Wasserstein (W)	0.5
	KL divergence (KL)	0.02
Latent-Space Exploration	MH temperature	0.4
Gradient Clipping	global_grad_clip	0.5

5.3 Evaluation Metrics

To comprehensively assess the quality, utility, and privacy of the synthetic time-series data, three categories of metrics are employed: Fidelity, Utility, and Privacy. Each category is vital for justifying the use of synthetic EHRs in health informatics:

5.3.1 Fidelity

We measure how well the synthetic data distribution aligns with the real data manifold using the PRDC suite of evaluation measures (precision, recall, density and coverage). [Naeem et al., 2020]

5.3.2 Utility

Adhering to the TSTR evaluation paradigm (training on synthetic data and testing on real data) [Esteban et al., 2017], we evaluate downstream task performance as a proxy for data utility. A LightGBM classifier is trained solely on synthetic data and then tested on held-out real data. We report: Area Under the Receiver Operating Characteristic Curve (AUROC) [Fawcett, 2006] and Area Under the Precision–Recall Curve (AUPRC) [McDermott et al., 2024]

5.3.3 Privacy

To quantify privacy risks, we adopt two complementary measures:

1. **Jensen–Shannon Divergence (JSD_{MIA})** between the empirical distributions of real and synthetic data, summarizing their global similarity. [Menéndez et al., 1997] [Hu et al., 2022]
2. **Nearest Neighbor Adversarial Accuracy Risk (NNAA)**, defined as the difference between two adversarial accuracies under a nearest-neighbor attack: [Yale et al., 2019]

$$\text{NNAA} = \text{AA}_{\text{test}}(D_{\text{gen}}, D_{\text{test}}) - \text{AA}_{\text{train}}(D_{\text{gen}}, D_{\text{train}}), \quad (5.1)$$

where AA_{test} is the adversarial accuracy measured on held-out real data D_{test} , and AA_{train} is measured on training data D_{train} . A large positive NNAA indicates overfitting to the training set, raising privacy-related concerns.

5.4 Results

5.4.1 Fidelity Evaluation

Table 5.2: Fidelity metrics

Model	Precision ($\uparrow$)	Recall ($\uparrow$)	Density ($\uparrow$)	Coverage ($\uparrow$)
HMM-GAN	1.00	1.00	0.465	0.520
TimEHR	1.00	0.067	0.564	0.313
TimeGAN	1.00	0.362	0.523	0.308

Table 5.2 presents Precision, Recall, Density, and Coverage for our HMM-GAN, TimEHR, and TimeGAN. All three models achieve perfect Precision (1.00), indicating none produce outliers outside the real data support. However, Recall reveals stark differences: our model attains Recall = 1.00, fully covering the real data manifold; TimEHR lags with Recall = 0.067; and TimeGAN obtains 0.362. In terms of Density (proportion of synthetic points near real points) and Coverage (fraction of real modes captured), our model scores 0.465 and 0.520, respectively substantially higher Coverage than TimEHR (0.313) and TimeGAN (0.308), and competitive Density. These results demonstrate that our method both respects the real data distribution and spans its diversity more completely than existing baselines. This perfect Precision and Recall for our model can be attributed to the patient-level generation strategy.

5.4.2 Utility Evaluation

Using the TSTR protocol (Table 5.3), we train a LightGBM mortality predictor on each synthetic dataset. Our model achieves AUROC = 0.5710 and AUPRC = 0.5341, outperforming

Table 5.3: Utility metrics

Model	AUROC	AUPRC
HMM-GAN	0.5710	0.5341
TimEHR	0.5183	0.5338
TimeGAN	0.479	0.479

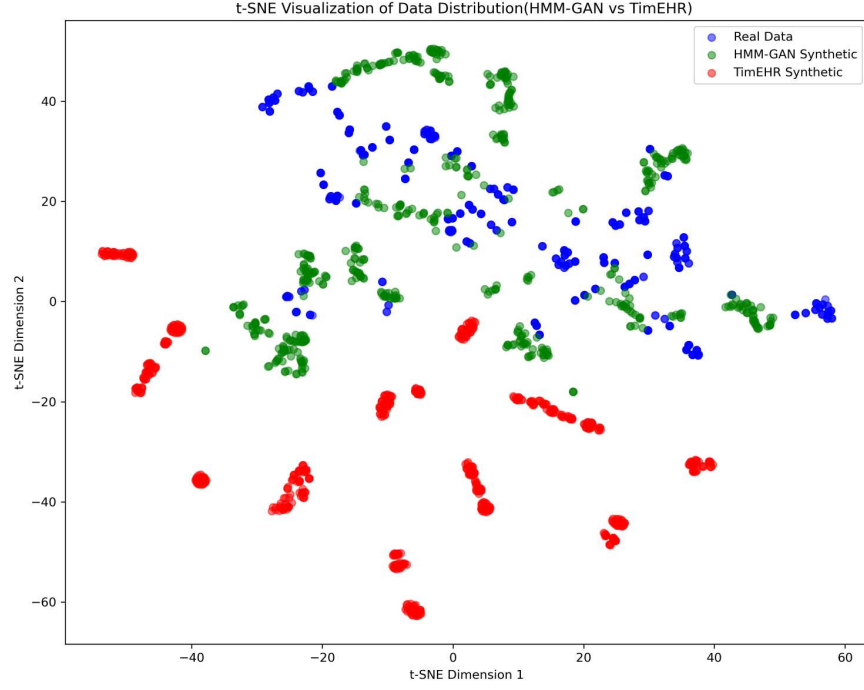
TimEHR (0.5183/0.5338) and TimeGAN (0.4790/0.4790). This indicates that classifiers trained on our synthetic EHRs better capture the predictive signals present in real patient data. This directly translates to the clinical value of the synthetic data: classifiers trained on the HMM-GAN synthetic data better capture the predictive signals present in real patient data, thereby directly enabling more accurate clinical prediction models. This demonstrates the synthetic data’s ability to support crucial real clinical tasks, allowing researchers to develop and validate predictive models without direct access to sensitive patient data.

5.4.3 Privacy Evaluation

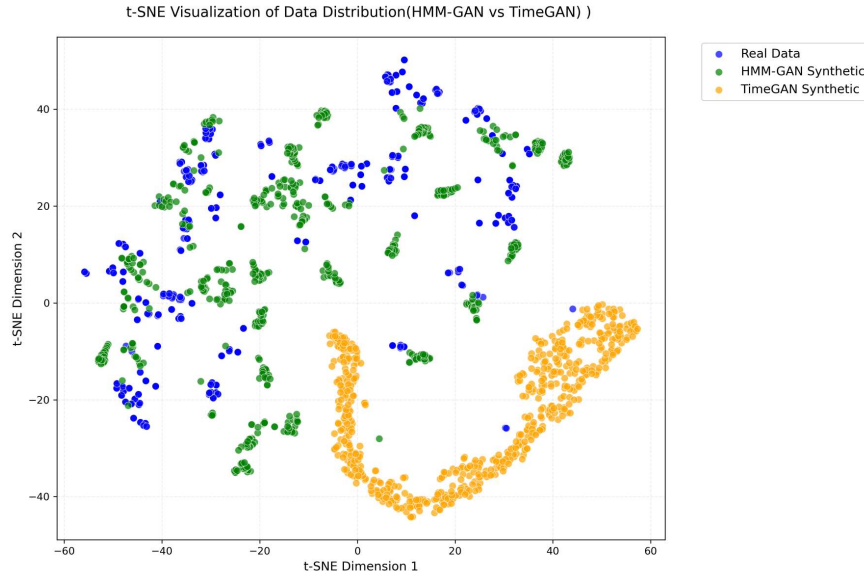
Table 5.4: Privacy and adversarial accuracy metrics

Model	JSD_{MIA}	AA_{test}	AA_{train}	NNAA Risk
HMM-GAN	0.2881	0.4089	0.4298	0.0209
TimEHR	0.7491	0.5084	0.5084	0.0000
TimeGAN	0.6555	0.4328	0.4787	0.0459

Table 5.4 reports two privacy metrics: JSD_{MIA} , and adversarial accuracy risk (NNAA). Our model yields $JSD_{MIA} = 0.2881$ and NNAA Risk = 0.0209, indicating low risk of training-set membership leakage. TimEHR and TimeGAN exhibit higher JSD_{MIA} (0.7491 and 0.6555) and comparable NNAA values, showing that our approach better balances fidelity with privacy.



(a) t-SNE: HMM-Gan vs TimEHR

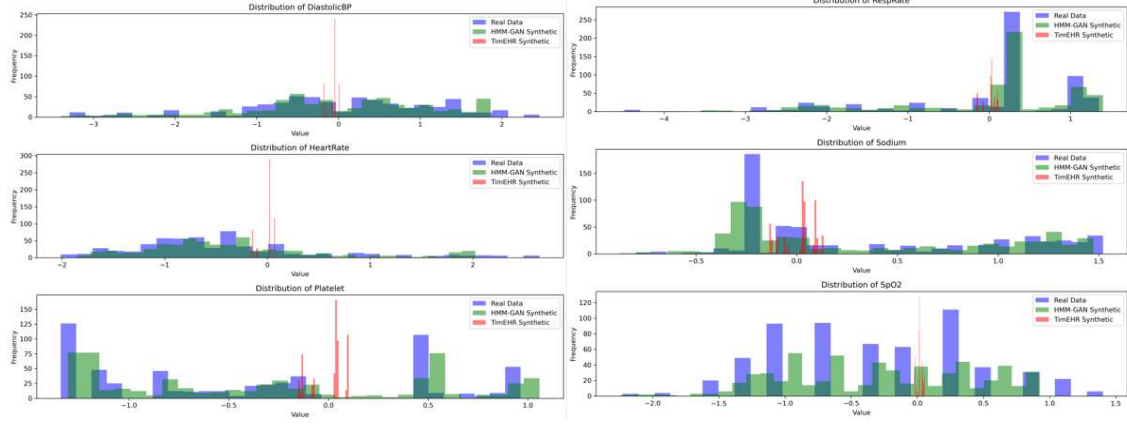


(b) t-SNE: HMM-Gan vs TimeGAN

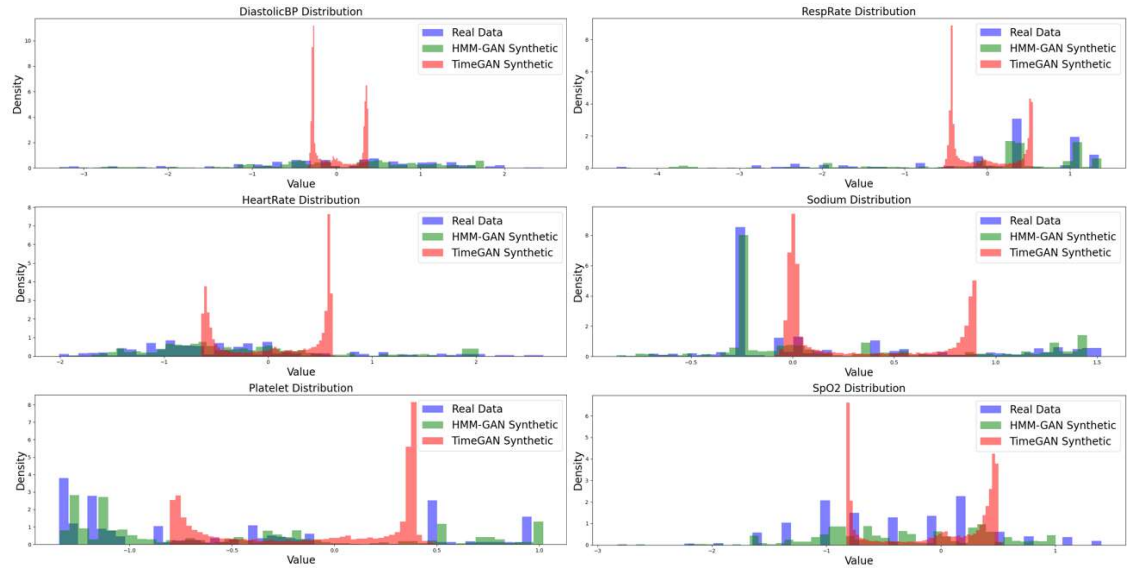
Figure 5.1: Comparison of t-SNE embeddings for HMM-Gan against (a) TimEHR and (b) TimeGAN.

5.4.4 $t-SNE$ Visualization

Figure 5.1 shows a t-SNE embedding of real versus synthetic sequences. In subfigure 5.1a, our model's samples (green) are well interleaved with real data (blue), indicating strong fidelity



(a) HMM-Gan vs TimEHR



(b) HMM-Gan vs TimeGAN

Figure 5.2: Feature-level distribution comparison. (a) HMM-Gan vs TimEHR. (b) HMM-Gan vs TimeGAN.

and broad mode coverage. In contrast, TimEHR’s points (red) form a distinct cluster, revealing limited generalization. Similarly, subfigure 5.1b shows that TimeGAN (orange) also fails to align with the real data manifold, suggesting mode collapse. Together, these visualizations confirm that our model better captures the underlying structure of patient trajectories.

5.4.5 Feature-Level Distribution Comparison

Figure 5.2 overlays histograms for six normalized clinical features. For each variable (DiastolicBP, HeartRate, Platelet, RespRate, Sodium, and SpO_2) our synthetic distributions (green) closely match the real data (blue). TimEHR’s outputs (red), however, concentrate sharply near zero, missing both central modes and distributional tails 5.2a. TimeGAN shows intermediate behavior but still under-represents rare values, consistent with its lower Coverage 5.2b.

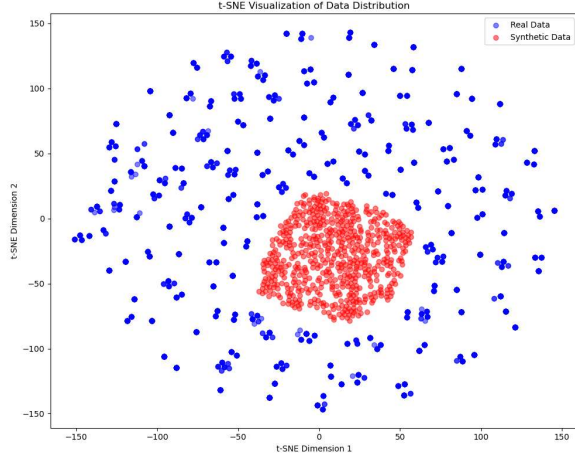
5.5 Ablation Study Results

To assess the individual contributions of each component within the proposed framework, we conducted a comprehensive ablation study. This analysis involved systematically removing key architectural elements and evaluating the impact on the model’s performance in generating realistic patient-level Electronic Health Records (EHRs). Figure 5.3 presents t-SNE embeddings of real versus synthetic sequences for each variant, while Table 5.5 reports their Frechét Inception Distance (FID) scores.

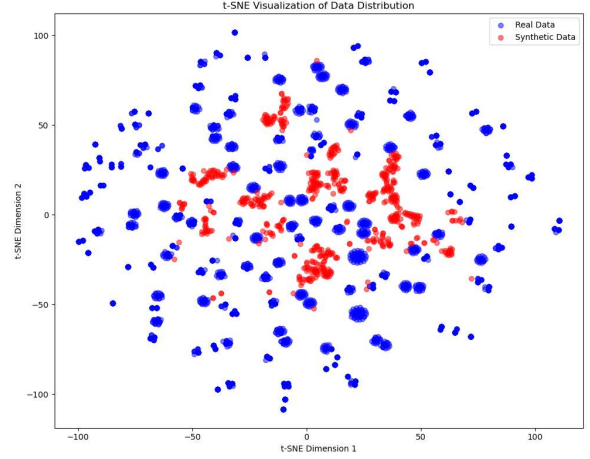
5.5.1 Ablation Configurations

We evaluated the following model variants:

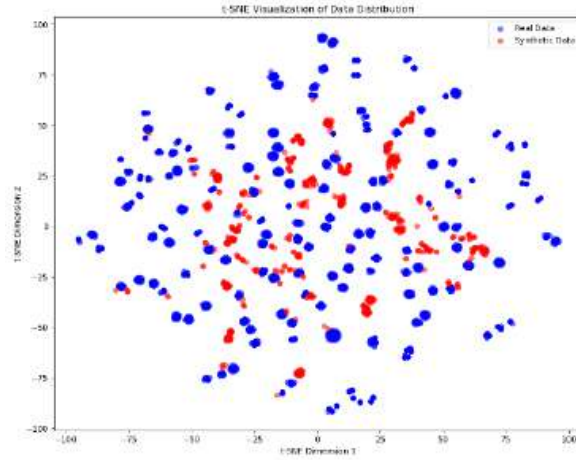
1. **Full HMM-GAN** The complete model integrating the Differentiable Hidden Markov Model (HMM) for capturing temporal dependencies, the Generative Adversarial Network (GAN) for generating realistic data sequences, and the Metropolis-Hastings (MH) sampling mechanism for refining sample quality.
2. **Without HMM Component** This variant excludes the DifferentiableHMM module, relying solely on the GAN and MH components to model temporal dynamics, thereby assessing the importance of explicit temporal modeling.



(a) Without HMM Component



(b) Without MH Sampling



(c) Full HMM-GAN

Figure 5.3: t-SNE Embeddings of Real vs. Synthetic Data for Ablation Variants.

3. **Without MH Sampling** This version omits the Metropolis-Hastings sampling step, allowing us to evaluate its impact on the quality and diversity of the generated sequences.

5.5.2 Ablation Results

Without the differentiable HMM

Removing the differentiable HMM causes the generator to collapse into a narrow region of latent space (see Figure 5.3a), failing to cover the manifold of real patient trajectories.

Table 5.5: Ablation Study Results

Model Variant	FID Score ($\downarrow$)
Without HMM Component	6.22
Without MH Sampling	1.06
Full Model	0.58

For EHR synthesis, this failure implies that important latent clinical states such as “stable,” “deteriorating,” and “recovering” are no longer represented in the outputs. Consequently, the synthetic cohort lacks the variety of patient pathways needed for meaningful research or model training. This severe mode collapse is reflected in a high Fréchet Inception Distance (FID) of 6.22, indicating poor fidelity and limited diversity in table 5.5

Without MH Sampling

When the differentiable HMM is retained, but the Metropolis–Hastings (MH) latent refinement is omitted, the generated points show partial overlap with the real data (see Figure 5.3b) but still miss multiple modes of the true distribution. The observed FID of 1.06 signals a marked improvement relative to the HMM-free variant, yet it remains notably distant from the full model’s optimal performance. Practically, this deficit results in a failure to represent rare but clinically critical trajectories sparse events that are crucial for downstream safety evaluation and predictive modeling.

Only the complete architecture produces synthetic sequences that intermix closely with real data clusters (see Figure 5.3c) and attains the lowest FID (0.58). These results indicate that both components are necessary to reproduce the full complexity of patient trajectories: the HMM supplies explicit, interpretable state structure while MH sampling promotes coverage of diverse latent modes. Together they yield synthetic EHRs that are statistically faithful and clinically plausible.

5.6 Discussion of Results

The empirical evaluation demonstrates three main conclusions. First, explicit, state-conditioned sequential modeling is essential for realistic EHR synthesis. Conditioning on clinician-interpretable state probabilities and encoded patient history preserves latent clinical trajectories and fine-grained temporal context, and is a core contribution of the proposed framework. Second, while prior state-of-the-art methods capture aspects of temporal structure, they exhibit critical limitations in clinical settings. For example, TimeGAN attempts to preserve dynamics but shows pronounced mode concentration and fails to reproduce long-term disease courses, an important shortcoming for chronic-disease modeling and prognostic applications. Similarly, image-based approaches such as TimEHR capture local correlations but sacrifice explicit sequential order; the resultant loss of temporal fidelity undermines clinical interpretability because event timing and ordering (e.g., vital-sign excursions, medication delivery) are central to diagnosis and outcome prediction. In both cases, the reproduced feature distributions concentrate narrowly and omit clinically important modes. Third, the combined use of a differentiable HMM, an LSTM encoder for cumulative history, and tempered Metropolis–Hastings–refined latent noise within a WGAN–GP training regime produces synthetic trajectories that both overlap the real data manifold and cover major clinical modes. This integrated approach preserves short- and long-term temporal coherence, mitigates mode collapse, and delivers synthetic data of sufficient fidelity to support downstream predictive modeling. Collectively, these properties position the proposed HMM–GAN framework as a robust solution for privacy-preserving EHR generation with practical utility for collaborative medical research.

5.7 Conclusion

This chapter outlined the experimental protocol used to validate the HMM–GAN framework, detailing the ablation studies that isolate individual component effects and the metrics

employed to quantify fidelity, utility, and privacy.

In the first phase, synthetic data quality was assessed against centralized baselines (TimeGAN and TimEHR) using a comprehensive suite of metrics, including PRDC (precision, recall, density, coverage), TSTR performance (AUROC, AUPRC), and distributional divergence. These metrics confirmed our method’s superiority in reproducing patient dynamics for downstream prediction tasks.

In the second phase, operational robustness and privacy trade-offs were analyzed through targeted ablations (such as removing the dHMM or MH refinement) alongside privacy diagnostics, including membership-inference risk measures like JSD_{MIA} and NNAA. Scenario analyses further demonstrated how specific design choices reliably balance high clinical utility with stringent privacy protections.

We close the chapter with a concise synthesis of the main findings: the proposed HMM-GAN framework by integrating explicit state-conditioned sequential modeling with latent refinement consistently improves coverage of clinically relevant modes, yields higher downstream utility, and maintains competitive privacy characteristics relative to existing baselines. These results substantiate the principal contributions of this work and provide a clear experimental foundation for future extensions and broader clinical validation.

General Conclusion

This thesis proposed, implemented and empirically validated a physiologically informed generative framework (HMM-GAN) for producing realistic, privacy-aware synthetic electronic health record (EHR) time series. The main contributions and the structure of the manuscript can be summarized as follows.

In Chapter two we surveyed classical anonymization and Statistical Disclosure Control (SDC) techniques for microdata. The chapter presents non-perturbative models (k -anonymity, ℓ -diversity, t -closeness), perturbative techniques (swapping, re-sampling, noise

addition), their advantages and limitations, and the principal disclosure risk types (membership inference, attribute inference, reconstruction). This provides the privacy foundations used later.

in the third chapter, we reviewed core deep-learning building blocks and architectures used throughout the thesis: optimization and loss basics, MLPs, CNNs, RNNs, and GANs. The goal was to ensure the reader is familiar with the concepts used in subsequent chapters.

In Chapter four, we presented a state-of-the-art survey of generative modelling for EHRs: GAN-based methods for tabular and time-series data, differential-privacy variants, and Markovian/probabilistic sampling techniques. The chapter analyzes limitations of prior work (temporal modelling, diversity, privacy guarantees) and motivates our hybrid approach.

Chapter five contains the core methodological contribution: a detailed presentation of the physiologically informed framework that integrates a differentiable Hidden Markov Model with a WGAN-GP, an LSTM encoder for patient history, and tempered Metropolis–Hastings latent refinement. Each pipeline step is described, including preprocessing choices and the clinical rationale for state conditioning.

To validate our contributions, Chapter 6 details dataset selection and preprocessing, experimental protocol, baselines, hyperparameters and evaluation metrics. We report quantitative results and interpretation (PRDC fidelity, TSTR utility, privacy diagnostics), t-SNE visualizations, and feature-level distribution comparisons. An ablation study demonstrates why the combined dHMM with Metropolis–Hastings design yields a favorable utility/privacy trade-off. Finally, we discuss limitations and practical implications.

Future work will extend the framework to richer, heterogeneous multimodal EHRs (medications, procedures, diagnostic codes) and un-structured text (clinical notes), integrate formal privacy mechanisms (for example DP-SGD or federated learning) to obtain provable privacy budgets, and develop improved metrics for clinical plausibility and information loss. Involving domain experts in clinician-in-the-loop evaluations will be crucial to validate safety and practical utility for real research workflows.

By combining interpretable temporal conditioning with principled latent refinement, this work lays a foundation for trustworthy, privacy-preserving synthetic EHR generation that can support collaborative medical research at scale.

Bibliography

- [Al Zamal et al., 2012] Al Zamal, F., Liu, W., and Ruths, D. (2012). Homophily and latent attribute inference: Inferring latent attributes of twitter users from neighbors. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 6, pages 387–390.
- [Andoni et al., 2008] Andoni, A., Indyk, P., and Krauthgamer, R. (2008). Earth mover distance over high-dimensional spaces. In *Proceedings of SODA*, volume 8, pages 343–352.
- [Arjovsky et al., 2017] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70, pages 214–223, Sydney, Australia. PMLR.
- [Bahdanau et al., 2014] Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- [Bai, 2022] Bai, Y. (2022). Relu-function and derived function review. In *SHS web of conferences*, volume 144, page 02006. EDP Sciences.
- [Carlo, 2004] Carlo, C. M. (2004). Markov chain monte carlo and gibbs sampling. Lecture notes 540, EEB. Lecture notes for EEB.

- [Carvalho et al., 2023] Carvalho, T., Moniz, N., Faria, P., and Antunes, L. (2023). Survey on privacy-preserving techniques for microdata publication. *ACM Computing Surveys*, 55(14s):1–42.
- [Chakraborty and Tripathy, 2016] Chakraborty, S. and Tripathy, B. (2016). Privacy preservation in relational data through l-diversity and recursive (c, l) diversity anonymisation. *International Journal of Mathematical Modelling and Numerical Optimisation*, 7(3-4):338–362.
- [Cho et al., 2014] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder–decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*. arXiv:1406.1078.
- [Dai et al., 2017] Dai, H., Dai, B., Zhang, Y.-M., et al. (2017). Recurrent hidden semi-markov model. In *International Conference on Learning Representations (ICLR)*.
- [Dalenius and Reiss, 1982] Dalenius, T. and Reiss, S. P. (1982). Data-swapping: A technique for disclosure control. *Journal of Statistical Planning and Inference*, 6(1):73–85.
- [Davidson et al., 2020] Davidson, S., Villarroel, M., Harford, M., Finnegan, E., Jorge, J., Young, D., and Tarassenko, L. (2020). Vital-sign circadian rhythms in patients prior to discharge from an icu: A retrospective observational analysis of routinely recorded physiological data. *Critical Care*, 24(1):181.
- [De Bels et al., 2023] De Bels, D., Bousbiat, I., Perriens, E., Blackman, S., and Honoré, P. M. (2023). Sedation for adult icu patients: A narrative review including a retrospective study of our own data. *Saudi Journal of Anaesthesia*, 17(2):223–235.
- [Dinur and Nissim, 2003] Dinur, I. and Nissim, K. (2003). Revealing information while preserving privacy. In *Proceedings of the Twenty-Second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 202–210. ACM.

- [Domingo-Ferrer and Mateo-Sanz, 2002] Domingo-Ferrer, J. and Mateo-Sanz, J. M. (2002). Practical data-oriented microaggregation for statistical disclosure control. *IEEE Transactions on Knowledge & Data Engineering*, 14(01):189–201.
- [Douc et al., 2018] Douc, R., Moulines, E., Priouret, P., and Soulier, P. (2018). *Markov chains*, volume 4. Springer.
- [Dwork, 2006] Dwork, C. (2006). Differential privacy. In *International Colloquium on Automata, Languages and Programming*, pages 1–12, Berlin, Heidelberg. Springer.
- [Elman, 1990] Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, 14(2):179–211.
- [Esteban et al., 2017] Esteban, C., Hyland, S. L., and Rätsch, G. (2017). Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*.
- [Faisal et al., 2022] Faisal, F., Mohammed, N., Leung, C. K., and Wang, Y. (2022). Generating privacy-preserving synthetic medical data. In *Proceedings of the IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10.
- [Fawcett, 2006] Fawcett, T. (2006). An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874.
- [Foomani et al., 2022] Foomani, F. H., Anisuzzaman, D. M., Niezgoda, J., Guns, G. S., and Yu, Z. (2022). Synthesizing time-series wound prognosis factors from electronic medical records using generative adversarial networks. *Journal of Biomedical Informatics*, 125:103972.
- [Franke and Degen, 2023] Franke, M. and Degen, J. (2023). The softmax function: Properties, motivation, and interpretation. *PsyArXiv DOI*, 10.

- [Goodfellow et al., 2020] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ..., and Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, 63(11):139–144.
- [Hamerly and Elkan, 2003] Hamerly, G. and Elkan, C. (2003). Learning the k in k-means. *Advances in neural information processing systems*, 16.
- [Hamo et al., 2016] Hamo, A., Benyamini, A., Shapir, I., Khivrich, I., Waissman, J., Kaasbjerg, K., Oreg, Y., von Oppen, F., and Ilani, S. (2016). Electron attraction mediated by coulomb repulsion. *Nature*, 535(7612):395–400.
- [Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.
- [Hu et al., 2022] Hu, H., Salcic, Z., Sun, L., Dobbie, G., Yu, P. S., and Zhang, X. (2022). Membership inference attacks on machine learning: A survey. *ACM Computing Surveys*, 54(11s):235:1–235:37.
- [Hundepool et al., 2012] Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Nordholt, E. S., Spicer, K., and De Wolf, P.-P. (2012). *Statistical disclosure control*. John Wiley & Sons.
- [Johnson et al., 2016] Johnson, A. E. W., Pollard, T. J., Shen, L., Lehman, L.-W. H., Feng, M., and Ghassemi, M. e. a. (2016). Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3:160035.
- [Jordon et al., 2018] Jordon, J., Yoon, J., and Van Der Schaar, M. (2018). Pate-gan: Generating synthetic data with differential privacy guarantees. *International conference on learning representations*.

- [Kag et al., 2020] Kag, A., Zhang, Z., and Saligrama, V. (2020). Rnns incrementally evolving on an equilibrium manifold: A panacea for vanishing and exploding gradients? In *International Conference on Learning Representations*.
- [Kang et al., 2023] Kang, H. Y. J., Batbaatar, E., Choi, D.-W., Choi, K. S., Ko, M., and Ryu, K. S. (2023). Synthetic tabular data based on generative adversarial networks in health care: generation and validation using the divide-and-conquer strategy. *JMIR Medical Informatics*, 11.
- [Karami et al., 2025] Karami, H., Hartley, M. A., Atienza, D., and Ionescu, A. (2025). Timehr: Image-based time series generation for electronic health records. *IEEE Journal of Biomedical and Health Informatics*. to appear.
- [Kuili et al., 2024] Kuili, S., Dabbour, K., Hasan, I., Herscovich, A., Kantarci, B., Chenier, M., and Erol-Kantarci, M. (2024). Adversarial machine learning-enabled anonymization of openwifi data. *arXiv preprint arXiv:2401.01542*.
- [Kyurkchiev and Markov, 2015] Kyurkchiev, N. and Markov, S. (2015). Sigmoid functions: some approximation and modelling aspects. *LAP LAMBERT Academic Publishing, Saarbrücken*, 4:34.
- [LeCun et al., 2002] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (2002). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- [Li et al., 2007] Li, N., Li, T., and Venkatasubramanian, S. (2007). t-closeness: Privacy beyond k-anonymity and l-diversity. In *Proceedings of the IEEE 23rd International Conference on Data Engineering*, pages 106–115. IEEE.
- [Liao et al., 2024] Liao, S., Ni, H., Sabate-Vidales, M., Szpruch, L., Wiese, M., and Xiao, B. (2024). Sig-wasserstein gans for conditional time series generation. *Mathematical Finance*, 34(2):622–670.

- [Liu et al., 2023] Liu, Y., Zhang, Z., and Qin, S. (2023). Neuralhmm: A deep markov network for health risk prediction using electronic health records. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.
- [Lu et al., 2023] Lu, C., Reddy, C. K., Wang, P., Nie, D., and Ning, Y. (2023). Multi-label clinical time-series generation via conditional gan. *IEEE Transactions on Knowledge and Data Engineering*, 36(4):1728–1740.
- [Machanavajjhala et al., 2007] Machanavajjhala, A., Kifer, D., Gehrke, J., and Venkitasubramaniam, M. (2007). l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3–es.
- [Marchesi et al., 2022] Marchesi, R., Micheletti, N., Jurman, G., and Osmani, V. (2022). Mitigating health data poverty: Generative approaches versus resampling for time-series clinical data. *arXiv preprint arXiv:2210.13958*.
- [McDermott et al., 2024] McDermott, M. B., Zhang, H., Hansen, L. H., Angelotti, G., and Gallifant, J. (2024). A closer look at auroc and auprc under class imbalance. *Advances in Neural Information Processing Systems*, 37:44102–44163.
- [Menéndez et al., 1997] Menéndez, M. L., Pardo, J. A., Pardo, L., and Pardo, M. d. C. (1997). The jensen-shannon divergence. *Journal of the Franklin Institute*, 334(2):307–318.
- [Moenning and Dodgson, 2003] Moenning, C. and Dodgson, N. A. (2003). Fast marching farthest point sampling. Technical report, University of Cambridge, Computer Laboratory.
- [Moore, 1996] Moore, R. (1996). Controlled data-swapping techniques for masking public use microdata sets. Technical report, US Census Bureau. Technical report.

- [Naeem et al., 2020] Naeem, M. F., Oh, S. J., Uh, Y., Choi, Y., and Yoo, J. (2020). Reliable fidelity and diversity metrics for generative models. In *International conference on machine learning*, pages 7176–7185. PMLR.
- [Qian, 1999] Qian, N. (1999). On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151.
- [Rabiner and Juang, 2003] Rabiner, L. and Juang, B. (2003). An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16.
- [Ramos et al., 2025] Ramos, A., Pascual, J. A., Navaridas, J., and Coluzza, I. (2025). Acceleration of parallel tempering for markov chain monte carlo methods. *arXiv preprint arXiv:2512.03825*.
- [Ramponi et al., 2018] Ramponi, G., Protopapas, P., Brambilla, M., and Janssen, R. (2018). T-cgan: Conditional generative adversarial network for data augmentation in noisy time series with irregular sampling. *arXiv preprint arXiv:1811.08295*.
- [Schmidt and Wiil, 2015] Schmidt, T. and Wiil, U. K. (2015). Designing a 3-stage patient deterioration warning system for emergency departments. In *HEALTHINF conference*, pages 470–7.
- [Schuster and Paliwal, 1997] Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.
- [Shi et al., 2021] Shi, N., Li, D., Hong, M., and Sun, R. (2021). Rmsprop converges with proper hyper-parameter. In *International Conference on Learning Representations*.
- [Shokri et al., 2017] Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

- [Sweeney, 2000] Sweeney, L. (2000). Simple demographics often identify people uniquely. *Health (San Francisco)*, 671:1–34.
- [Sweeney, 2002] Sweeney, L. (2002). k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570.
- [Templ et al., 2015] Templ, M., Kowarik, A., and Meindl, B. (2015). Statistical disclosure control for micro-data using the r package sdcmicro. *Journal of Statistical Software*, 67:1–36.
- [Torra and Navarro-Arribas, 2023] Torra, V. and Navarro-Arribas, G. (2023). Attribute disclosure risk for k-anonymity: the case of numerical data. *International Journal of Information Security*, 22(6):2015–2024.
- [University of California, San Francisco, nd] University of California, San Francisco (n.d.). Protocol for septic shock. <https://hospitalhandbook.ucsf.edu/12-protocol-septic-shock/12-protocol-septic-shock>. UCSF Hospital Handbook, Accessed: 2026-06-18.
- [Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- [Xie et al., 2018] Xie, L., Lin, K., Wang, S., Wang, F., and Zhou, J. (2018). Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*.
- [Xu et al., 2019] Xu, L., Skoularidou, M., Cuesta-Infante, A., and Veeramachaneni, K. (2019). Modeling tabular data using conditional gan. *Advances in neural information processing systems*, 32.

- [Yale et al., 2019] Yale, A., Dash, S., Dutta, R., Guyon, I., Pavao, A., and Bennett, K. (2019). Privacy preserving synthetic health data, esann 2019-european symposium on artificial neural networks. *Computational Intelligence and Machine Learning*.
- [Yoon et al., 2019] Yoon, J., Jarrett, D., and van der Schaar, M. (2019). Time-series generative adversarial networks. In *Advances in Neural Information Processing Systems*, volume 32. NeurIPS 2019.
- [Yoon et al., 2023] Yoon, J., Mizrahi, M., Ghalaty, N. F., Jarvinen, T., Ravi, A. S., Brune, P., and Pfister, T. (2023). Ehr-safe: Generating high-fidelity and privacy preserving synthetic electronic health records. *NPJ Digital Medicine*, 6(1):141.
- [Yu et al., 2017] Yu, J., Xie, L., Xiao, X., and Chng, E. S. (2017). A hybrid neural network hidden markov model approach for automatic story segmentation. *Journal of Ambient Intelligence and Humanized Computing*, 8:925–936.
- [Zhang et al., 2020] Zhang, Z., Yan, C., Mesa, D. A., Sun, J., and Malin, B. A. (2020). Ensuring electronic medical record simulation through better training, modeling, and evaluation. *Journal of the American Medical Informatics Association*, 27(1):99–108.
- [Zhao et al., 2021] Zhao, Z., Kunar, A., Birke, R., and Chen, L. Y. (2021). Ctab-gan: Effective table data synthesizing. *Asian conference on machine learning*, pages 97–112.