Master thesis

Master of Science in Engineering, Computer Science and Engineering, 300 Credits

Generation of patient trajectories

Improving Time-series Generative Adversarial Networks for Generating Electronic Health Records

Master Thesis Project, 30 Credits

Halmstad, July 22, 2024
Joar Hjerpe

HALMSTAD UNIVERSITY

ABSTRACT

The integration of Electronic Health Records has transformed patient data management in healthcare, offering comprehensive insights into patient journeys through diverse data types. Despite the value of these records for developing diagnostic models, their use is heavily regulated by strict regulations to ensure patient privacy and confidentiality. Generative deep learning models can generate realistic synthetic data to circumvent these restrictions. Patient trajectories, marked by variable lengths and multi-modality, include irregularly sampled medical time series, which is often challenging for models to handle. In this thesis, we aim to create a GAN-based architecture that is able to handle as well as mimic the variable length and multi-modality of patient trajectories. To evaluate the synthetic dataset, we used three types of evaluation metrics based on fidelity, utility and privacy. Our results indicate that while the proposed model outperforms the baselines in some evaluation tests, it performs subpar in others, concluding that the fundamental ideas work and further improvements are possible.

CONTENTS

1 INTRODUCTION 1
- 1.1 Problem formulation 1
- 1.2 Research questions and goals 2
- 1.3 Novelty 3

2 RELATED WORKS 5

3 BACKGROUND 7
- 3.1 The MIMIC-III dataset 7
- 3.2 Generative Adversarial Networks 7
  - 3.2.1 Wasserstein GAN 8
- 3.3 RNN 8
- 3.4 Autoencoders 8

4 METHODOLOGY 11
- 4.1 Overview of the Thesis 11
- 4.2 Proposed model: TOP-GAN 11
- 4.3 Processing of the data 14
- 4.4 Evaluation techniques 15
  - 4.4.1 Fidelity Evaluation 15
  - 4.4.2 Utility Evaluation 17
  - 4.4.3 Privacy Evaluation 18

5 RESULTS 21
- 5.1 Fidelity Evaluation Results 21
  - 5.1.1 Number of Occurrences of Specific Elements 21
  - 5.1.2 Statistical Analysis of Continuous Feature Generation 23
- 5.2 Utility Evaluation Results 26
  - 5.2.1 Predictive model Results 27
  - 5.2.2 Discriminative model Results 27
- 5.3 Privacy Evaluation Results 28

6 DISCUSSION 29
- 6.1 Analysis of the results 29
- 6.2 Conclusion 30
- 6.3 Future work 31

BIBLIOGRAPHY 33

ACRONYMS

EHR Electronic Health Record

HIPAA Health Insurance Portability and Accountability Act

GDPR General Data Protection Regulation

ISMTS Irregularly Sampled Medical Time Series

GAN Generative Adversarial Networks

DAAE Dual Adversarial Auto Encoder

RNN Recurrent Neural Network

TOP-GAN Time Oriented Patient trajectory GAN

MIA Membership Inference Attack

1 INTRODUCTION

The integration of Electronic Health Records (EHRs) in healthcare has revolutionized the way patient data is managed and utilized. EHRs encompass a wide range of data, including age, sex, demographics, medical history, and patient encounters, that contain details of the interaction between patients and healthcare providers to receive care and treatments. Combining such information provides a comprehensive view of the events and experiences that a patient encounters throughout their journey through the healthcare system, i.e. their patient trajectory. This thorough collection of medical information is helpful in order to develop models to help diagnose patients, but accessing and using this trove of data is highly regulated [4]. The Health Insurance Portability and Accountability Act (HIPAA) [2] and the General Data Protection Regulation (GDPR) [1], in the U.S. and EU, respectively, contain a detailed list of rules and regulations on what one can and can not do in order to maintain privacy and confidentiality of patients. To get around this, one can use generative deep learning models to generate realistic synthetic data as a replacement.

A patient trajectory is characterized by variable trajectory length and multi-modality. This variability reflects the diverse and dynamic nature of patient interactions with healthcare providers. As such, a large part of the data contained within these trajectories is what is called irregularly sampled medical time series (ISMTS) [16]. While the irregularity of ISMTS can be challenging for many models that require uniformly sampled visits, it also provides valuable insights. Additionally, the multi-modality of patient data, encompassing clinical notes, lab results, and imaging, provides a rich and diverse set of information that can offer deeper insights into patient health beyond the timing of visits.

1.1 PROBLEM FORMULATION

The objective of this Master thesis is to develop a model that is capable of generating real patient health trajectories by focusing on the irregular length of both the trajectory itself and the diagnosis codes sequence found in patients' EHRs. This involves extracting and utilizing the multi-dimensional data that makes up the EHR, such as age, sex, medical history, and visit records.

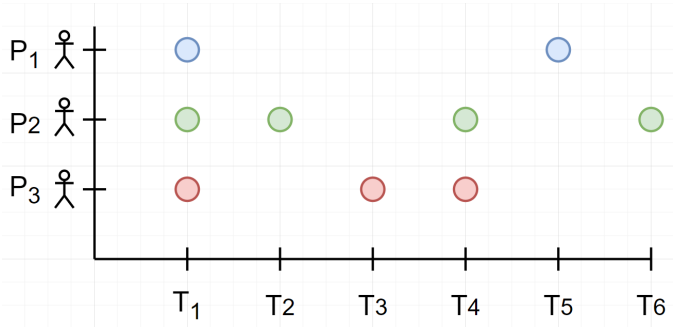

Figure 1: Overview of EHRs showing show the number of visits change between patients. This irregularity in lengths of visits in a trajectory can be difficult to handle

Mathematically, the problem could be formulated as follows: Consider a dataset, $\mathcal{D} = \{T_1, T_2, \ldots, T_n\}$. Each element T in this dataset represents a sequence of visits for a patient, i.e. their trajectory. Each visit within a trajectory, $v_i \in T_n$, contains five types of data, where one of the types is itself a sequence of varying length. Let $v_i = \{age_i, days_i, sex_i, hr_i, icd_i\}$ represent a visit in the dataset, where $\{age_i, days_i, sex_i, hr_i, icd_i\}$ represents the patient's age, the days since last visits, the sex, the heart rates and the diagnoses in the form of ICD codes for a visit i. Additionally, icd_i is a sequence of diagnosis codes $\{icd_{i1}, icd_{i2}, \ldots, icd_{iL}\}$ with its length L varying between each visit v_i.

Our goal is to create a model capable of generating a synthetic dataset $\hat{\mathcal{D}} = \{\hat{T}_1, \hat{T}_2, \ldots, \hat{T}_k\}$ which, when tested, looks and performs similarly to the real dataset $\mathcal{D}$.

To accomplish this, a comprehensive literature review of the current state-of-the-art techniques for synthetic EHR generation must first be done. The main consideration in this review is Recurrent neural networks (RNN) for their ability to handle the temporal aspect of the task.

The MIMIC-III dataset was used as the primary source of data for the duration of this thesis. The MIMIC-III dataset contains thousands of patients' data together with their admissions. The dataset uses a relational structured format where predefined relationships are stored in tables and columns. The data, as per the regulations set by HIPAA, has also been de-identified as this type of information is highly personal for the patients.

1.2 RESEARCH QUESTIONS AND GOALS

The aim of this thesis is to answer these two main research questions:

- Can the proposed system generate new, authentic-looking synthetic patient trajectories?
- Given EHR data that is highly irregular both in the number of visits given an interval but also in the number of medical events reported at encounter level, e.g., lab measurements, vital signs, medications, etc. Is this irregularity possible to capture?

Using the questions set above, the main goals of this thesis project is to:

- Propose a novel generative framework for generating synthetic EHR data.
- The model should be able to handle the varying lengths of the sequences as well as the multi-modality found with EHRs.

The criteria *authentic-looking* in this thesis will mainly be based on two different types of metrics: fidelity and utility. These metrics represent the similarity between datasets and the usability of the dataset in hypothetical downstream tasks. As such, an authentic-looking synthetic dataset would not only be similar from a statistical point of view but would also perform similarly to the original dataset when used for downstream tasks such as prediction.

1.3 NOVELTY

The aim of this thesis is to generate the patient trajectory where each visit does not only contain multi-modal data but also is of varying lengths. This will be done by employing multiple encoder-decoder pairs for each of the features in the multi-modal dataset. However, instead of the usual pipeline from one encoder to its corresponding decoder, the encoders will together create a shared latent representation, which is copied and used by all decoders.

Combining this with the GAN architecture tasked with generating new, synthetic latent representations should allow for the generation of new multi-modal patient trajectories.

2

RELATED WORKS

Generative Adversarial Networks (GANs) have advanced the field of generative modelling with its two networks model consisting of the generator and the discriminator [7]. These two models are each trained to accomplish two opposing goals: The generator creates synthetic data samples, while the discriminator evaluates them against real data. This process refines the generator's output, leading to increasingly realistic synthetic data. Their paper laid the groundwork for this technology, highlighting its potential across various domains, including image generation, natural language processing, and healthcare.

Advancements in GANs have led to specialized models for EHR generation [8][9]. One such model is the MedGAN [6]. MedGAN is tailored for EHR data and can accurately represent events found in EHRs by integrating both feed-forward networks and an autoencoder. While showing impressive results regarding the quality of the synthetic data when leveraging the help of a human doctor acting as a discriminator, the data generated by MedGAN contained certain impossible combinations of medical codes that were easy to spot.

To get rid of these impossible combinations that plagued the MedGAN model, Yan et al. [18] developed hierarchical GAN (HGAN), which introduces a penalizing system that forces the model to learn the spatial constraints between features in EHRs. By considering both the hierarchical and multidimensional nature of EHR data and applying appropriate penalties, it ensures that the generation is clinically relevant and realistic. This approach is particularly important in maintaining the integrity of the data, especially in terms of feature interactions that are crucial in clinical settings.

In the context of healthcare, specifically EHRs, irregular sampling and non-uniform time intervals between data samples are abundant [16]. This irregularity can be classed as either intra-visit or inter-visit, with intra-visit irregularity depicting the irregular sampling within each visit and inter-visit irregularity representing the patient's non-uniform visiting schedule. This irregularly sampled data complicates data analysis and interpretation, as traditional data processing techniques and models often assume regularly spaced intervals and complete sequences.

Both MedGAN and HGAN do not take into account this temporal irregularity for their models. However, a lot of work has been done to develop methods to capture this temporal irregularity.

TimeGAN [20] is made up of 4 main components: 1) The embedding component, 2) the recovery component, 3) the generate component and 4) the discriminate component. By training these components together as pairs and introducing a unique loss function between the two pairs, they aim to capture the conditional probability distributions within the data while taking into account the temporal aspect at the same time.

T-CGAN, proposed by Ramponi *et al.* [14], employs a conditional GAN network conditioning both the generator and the discriminator on an additional vector representing timestamps. While it is able to learn and mimic the temporal properties of the data it is only for a single sample and is unable to keep it for a sequence of samples.

From the same researcher that proposed TimeGAN, Yoon *et al.* proposed EHR-Safe[21], a framework able to generate multi-modal data with varying lengths. The framework is made up of encoder-decoder networks coupled with GANs. By dividing up the raw dataset into multiple types of feature-mask pairs, encoding them into a joint latent space, and training the generator in this space, they were able to produce synthetic data with fidelity that rivaled that of the original data. However, like T-CGAN, the data generated is visit-level and ignores the temporal correlation between visits.

The dual adversarial autoencoder (DAAE) proposed by Lee *et al.* [12] uses both a recurrent autoencoder with an outer and an inner GAN to generate chronological sequences of patient visits containing a set of diagnosis codes. Both the number of visits and the number of codes per visit can vary in length, making it more able to capture the irregular nature of hospital visits.

MT-GAN [13], with the goal of accurately and fairly representing uncommon diagnoses, is able to generate sequences of visits with each visit containing multi-label data in the form of diagnosis codes. This is done using a WGAN with a gradient penalty together with attention scores.

While MT-GAN, DAAE and EHR-safe are the closest out of the previously mentioned models that is able to generate sequences of patient trajectories, any of them on their own does not fulfil our goal of having a model that is able to generate patient sequences of varying lengths with each visit containing data that is both varying in lengths and multi-modal.

3

BACKGROUND

3.1 THE MIMIC-III DATASET

The MIMIC-III (Medical Information Mart for Intensive Care III) [10][11] is a dataset containing EHR patient data for tens of thousands of deidentified patients. The dataset is divided up into two parts: *hosp* and *icu*, with *hosp* containing general information about the hospital stay and patient and *icu*'s information is gathered from lab equipment and medical systems.

3.2 GENERATIVE ADVERSARIAL NETWORKS

First proposed in 2014 by Ian Goodfellow *et al.* [7], generative adversarial networks (or GANs) are characterized by their pair of networks locked in a min-max game against each other. Each pair consists of a generator $\mathbf{G}$ and a discriminator $\mathbf{D}$ whose goals are, respectively, to generate realistic samples and to distinguish between real and generated samples.
The generator takes in a noise z as input and creates a synthetic sample $\tilde{x} = \mathbf{G}(z)$. The discriminator's goal is then to correctly classify real samples as real and synthetic ones as synthetic, thus creating the objective function:

$$\min_{G} \max_{D} \mathop{\mathbb{E}}_{\mathbf{x}\sim p_x} [\log \mathbf{D}(\mathbf{x})] + \mathop{\mathbb{E}}_{\mathbf{z}\sim p_z} [\log(1 - \mathbf{D}(\mathbf{G}(\mathbf{x}))] \tag{1}$$

Proven in its original paper, for an optimal discriminator $\mathbf{D}$, the generator $\mathbf{G}$ has its global optimum only when the two distributions are equal to each other. In order to achieve this optimum, it is necessary to quantifiably measure how similar two probability distributions are to each other. To achieve this, the Jensen-Shannon divergence is used.

Built upon the KL (Kullback-Leibler) divergence (Eq 2), the Jensen-Shannon divergence (Eq 3) is a method that returns a similarity score that is, unlike the KL divergence, bounded within $[0, 1]$. It is also symmetrical, meaning $D_{JS}(P\|Q) = D_{JS}(Q\|P)$ which does not always hold true with KL divergence.

$$D_{KL}(P\|Q) = \int_x P(x) \log\left(\frac{P(x)}{Q(x)}\right) dx \tag{2}$$

$$D_{JS}(P\|Q) = \frac{1}{2}D_{KL}(P\|\frac{P+Q}{2}) + \frac{1}{2}D_{KL}(Q\|\frac{P+Q}{2}) \tag{3}$$

Unfortunately, the vanilla objective function 1 for GAN makes it difficult to train effectively [17]. Problems such as mode collapse, where the generator gets stuck to always produce outputs with minimal variety, and vanishing gradients plague the training process. Changing the metric in the loss function from JS divergence to the Wasserstein Distance aims to solve these issues.

3.2.1 *Wasserstein GAN*

Instead of using the Jenson-Shannon divergence as the metric in the loss function, Wasserstein GAN (WGAN) proposed by Arjovsky *et al.* [3] uses the Wasserstein-1 metric, otherwise known as the Earth Mover's distance, to measure the similarity between two distributions.
The Earth Mover's distance, defined in 4, can be thought of as a measurement of how much energy is required to transform one distribution to the other if they were piles of dirt, hence the name.

$$W(\mathbf{P}_r, \mathbf{P}_g) = \inf_{\gamma \in \Pi(\mathbf{P}_r, \mathbf{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|] \tag{4}$$

with $\mathbf{P}_r$,$\mathbf{P}_g$ being two probability distributions that act as marginals for the joint distributions $\gamma(x, y)$.

3.3 RNN

Recurrent Neural Networks (RNNs) [5] are a type of neural network designed for processing sequential data, such as time series, speech, text, or any other form of data where the order matters. Unlike traditional neural networks, which assume all inputs (and outputs) are independent of each other, RNNs possess a form of memory (or hidden state) that captures and stores information about what has been calculated so far. The information stored in this hidden state is then used in later time steps, allowing them to make decisions based on past information.

3.4 AUTOENCODERS

Autoencoders [15] are a type of neural network used primarily for dimensionality reduction and feature learning. An autoencoder aims to

learn a compact representation (encoding) of input data by passing the data through a bottleneck architecture, then reconstructing the original input as closely as possible (decoding).

The network is divided into two main parts: the encoder, which compresses the input into a latent-space representation, and the decoder, which reconstructs the input from this representation. By training the autoencoder to minimize reconstruction errors, it learns to capture the most relevant features of the data.

4

METHODOLOGY

4.1 OVERVIEW OF THE THESIS

This thesis aims to investigate the feasibility of developing a generative deep learning model architecture that does not use any aggregation or imputation techniques to handle the irregular nature of EHR data but instead aims to learn from it. It should then be able to replicate the irregularity in its generation, resulting in a synthetic dataset of patient trajectories that is both varying in length in the number of visits in their trajectory and also in the number of diagnoses measured.

Chapter 4 is divided up into 3 main parts:

1. Presenting the proposed model TOP-GAN.
2. Preprocessing the MIMIC-III dataset and describing the types of data used.
3. Introducing the evaluation techniques

4.2 PROPOSED MODEL: TOP-GAN

The proposed network, called TOP-GAN (**T**ime **O**riented **P**atient trajectory GAN), builds upon the dual adversarial autoencoder (DAAE) [12] by making it able to generate multi-modal data. The main motivation for choosing DAAE as a base is its proven performance at generating multi-visit sequences of medicinal diagnoses. By extending the capabilities of its autoencoder to accommodate the new additional heterogeneous features as well as modifying the loss objective, we achieve a model capable of generating both continuous and set-valued data for multiple visits per patient.

As stated in Section 2, the dual adversarial autoencoder, shortened DAAE, consists of 3 main components: The sequence-to-sequence autoencoder, an inner GAN, and an outer GAN. Each of the components has its own unique loss function which together creates the objective function:

$$\min_{\phi,\psi,\theta} \max_{\omega_z,\omega_x \in W} L_{rec}(\phi,\psi) + L_z(\theta,\omega_Z) + L_x(\psi,\omega_x) \tag{5}$$

The autoencoder, inner and outer GAN each have their own set of trainable parameters (ϕ and ψ for the autoencoder, θ and ω_z for the

inner GAN, and ψ and ω_x for the outer GAN). The subscript x and z represent if the sample is obtained from the original dataset distribution or from the latent space. W is the set of 1-Lipschitz function set.

$$L_{rec}(\phi, \psi) = \mathbb{E}_{x \sim \mathbb{P}_x} \left[d_{BCE}(x, Dec_\psi(Enc_\phi(x))) \right]$$

$$L_z(\theta, \omega_Z) = \mathbb{E}_{x \sim \mathbb{P}_x} \left[f_z(Enc_\psi(x)) - \mathbb{E}_{s \sim \mathbb{P}_s}[f_z(G_\theta(s)] \right]$$

$$L_x(\psi, \omega_x) = \mathbb{E}_{x \sim \mathbb{P}_x} \left[f_x(x)] - \mathbb{E}_{z \sim \mathbb{P}_Q, \mathbb{P}_z}[f_x(Dec_\psi(z)] \right]$$

However, what makes TOP-GAN different is instead of only using one channel in its autoencoder, it jointly employs multiple encoder-decoder pairs, one for each feature. Since each decoder is suited and trained on one specific feature, the reconstruction loss function used for the autoencoder has to be updated to accommodate for this. The total reconstruction loss for the autoencoder is therefore:

$$L_{rec} = \beta_0 L_{age} + \beta_1 L_{gender} + \beta_2 L_{hr} + \beta_3 L_{codes} + \beta_4 L_{days}$$

With

$$\beta_0, \beta_1, \beta_2, \beta_3, \beta_4$$

being additional weights in order to balance out the losses from the different sources.

Fig 3 shows the general architecture of the TOP-GAN model. The main part, the autoencoder, is made up of multiple encoder-decoder pairs ($\mathbf{E}_k$ and $\mathbf{D}_k$), one for each feature k. The encoders each take their respective feature and encodes it into a lower dimensional representation. This lower dimensional representation output of each encoder channel is then used together with the other channels in order to create a joint latent space, z_E. The decoders' objective is then to recover the original data by using the information from the shared latent space. The decoders each samples a copy of z_E to produce sequences, which are then mapped to their respective feature space.

The generator $\mathbf{G}$'s objective is to generate synthetic vectors, but instead of directly synthesizing the continuous and discrete-valued time series data, it generates samples z_G mimicking that of z_R, which are then later passed through the decoders to be used to generate the final dataset during inference. The two critics f_z and f_x each try to discern real samples from fake ones, with the difference between them being that f_z is trained on samples from the shared latent space made out of the encoder with real data and the latent vector generated from the generator. Similarly, f_x is trained on samples from the decoders that

use either the latent space from the encoders or from the generator.

The difference between the inner models of the decoder is how it handles the list-type features compared to the other types. These decoders insert two additional tokens into the sequence to represent the start and the end, e_{sos} and e_{eos}, respectively. By dynamically stopping the inference of a sequence when it generates an e_{eos} token, we can achieve varying lengths in our synthetic data.

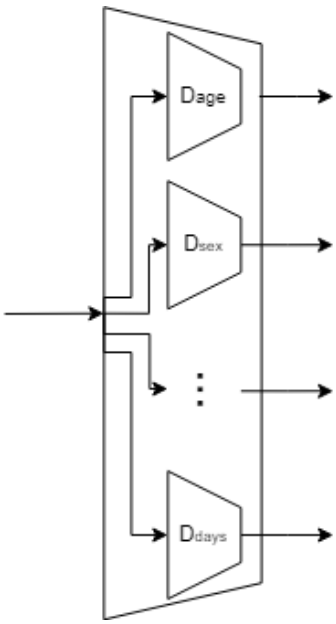

Figure 2: The decoder is made up of multiple smaller decoders, one for each feature

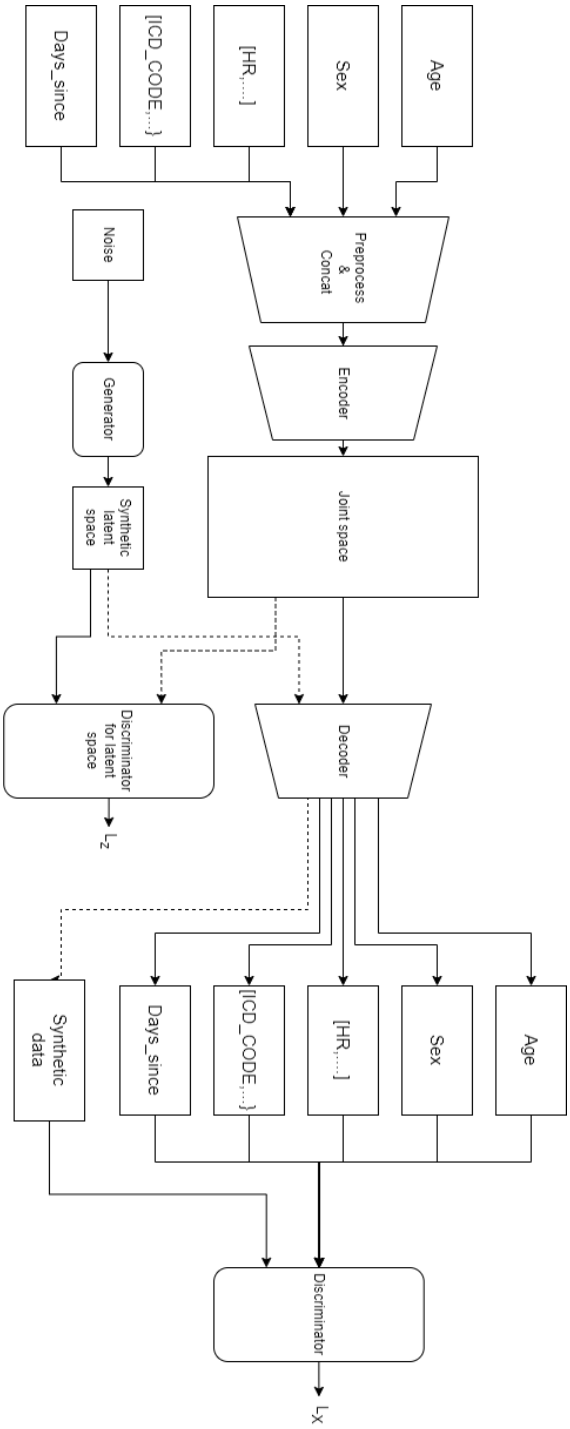

Figure 3: The structure of TOP-GAN

4.3 PROCESSING OF THE DATA

The data used is multi-modal, both categorical and numerical, as well as static and temporal. It currently consists of 5 fields, of which four are temporal and one is static. These are *days_since*, *sex*, *icd*, *heart_rate*

	Type	Info
days_since	Temporal and numeric	Amount of time since the last visit.
sex	Static and categorical	The sex of the patient.
icd	Temporal and categorical	A list of diagnosis codes.
heart_rate	Temporal and numerical	Heart rate values during a visit.
age	Temporal and numerical	The age of the patient.

Table 1: The dataset.

and *age*. Table 1 is a description of the type of data stored in each feature.

The features *days_since* and *age* were chosen to give an explicit dependency on the time between visits while being co-dependent on each other. The goal is that an optimally trained generative model should be able to understand the co-dependency between these two temporal features and generate an output where the amount of change in one of the features is mirrored in the other. For example: A patient with a *age*= 53, *days_since*= 0 during the first visit and *days_since*= 365 during his second visit. The 365 days should be reflected in the second visit's *age* and be equal to 54.

4.4 EVALUATION TECHNIQUES

Many different evaluation metrics exist to measure the performance of GANs quantifiably, and they differ based on the type of criteria in focus. To create *authentic-looking* data based on our definition in chapter 1, three different types of tests will performed to evaluate the effectiveness of the proposed model properly. These are Fidelity, Utility and Privacy, each meant to evaluate TOP-GAN from different perspectives using different methods. The last metric, privacy, is not part of our definition of authentic-looking, but it is essential due to the nature of data and the potential risks associated with it. All of the results listed below were the average across multiple runs to give as accurate of a result as possible.

4.4.1 *Fidelity Evaluation*

The model's fidelity will be evaluated using statistical similarity between the generated synthetic data and the original data. For this, two main types of tests were performed in order to evaluate the generation of both diagnosis codes and the age, days since the last visit, sex, and heart rate across visits.

4.4.1.1 *Diagnosis codes generation evaluation*

For the diagnosis code generation, similarity measures using cosine similarity and JS-divergence will be calculated for the top 30 most frequent diagnosis codes from both the original dataset and the synthetic datasets. These values were gathered and then compared across all the tests to compare it with the original data and the data using different hyperparameters setups.

Cosine similarity measures the cosine of the angle between two non-zero vectors. The cosine similarity between two vectors A and B is given by:

$$\text{Cosine Similarity}(A, B) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (6)$$

where $A \cdot B$ is the dot product of the vectors and $\|A\|$ and $\|B\|$ are the magnitudes of the vectors. In the context of testing TOP-GAN, cosine similarity was employed to compare the feature representations of the real and generated data, with values closer to 1 indicating higher similarity.

Like cosine similarity, Jensen-Shannon Divergence (JS-Divergence) is also a method to measure the similarity but instead between two probability distributions P and Q. It is defined as:

$$JS(P\|Q) = \frac{1}{2}\left(KL(P\|M) + KL(Q\|M)\right) \quad (7)$$

where $M = \frac{1}{2}(P + Q)$ is the mixture distribution of the two distributions and KL denotes the Kullback-Leibler divergence:

$$KL(P\|M) = \sum_{i} P(i) \log \frac{P(i)}{M(i)} \quad (8)$$

JS-Divergence is symmetric and bounded between 0 and 1. In our testing, lower JS-Divergence values indicate that the generated data distribution closely matches the real data distribution.

4.4.1.2 *Continuous features generation evaluation*

To evaluate the generation of continuous features, the mean and standard deviation for both the real and generated data were computed

across different latent space sizes. The difference in these values between the real and generated data would then indicate how similar the central tendency and spread are between the datasets. In addition, the pairplots are also calculated and shown to visualize and compare the whole distributions of the features and the correlation between them.

Similarly, the mean and standard deviation of the number of visits for each patient and the mean and standard deviation of the number of diagnoses per visit. The goal with this is to evaluate how well TOP-GAN accurately capture these distributions in frequency and length.

4.4.2 *Utility Evaluation*

Different downstream models will be trained on both the synthetic and the original dataset to measure the usability of the proposed model. The goal of this metric is to make sure that the proposed model is able to generate data that accurately captures the underlying characteristics of the original data to such an extent that it can be used as a replacement.

To this end, tests for two types of downstream tasks were performed: one for predicting the next time step of a time series and another for discriminating between real and generated synthetic data. Since TimeGAN [20] and DAAE [12] generates continuous values and set-valued data, respectively, the downstream model used had to be split into two models for each of the tasks, one making use of the continuous data from TOP-GAN and TimeGAN, and another using the set-valued codes from TOP-GAN and DAAE. This results in a total of four models trained for utility evaluation, two for prediction and two for discrimination.

4.4.2.1 *Predictive task evaluation*

In the predictive task we setup RNN-based models trained to predict the next time step of the sequence of visits. The sequences are split into inputs and labels where the input is every element of the sequence except for the latest one while the last one is saved as the true label. For example: Given a patient with three sequences symbolizing three visits $\{v_1, v_2, v_3\}$, the first two visits $\{v_1, v_2\}$ are used as input to the model. The model's output sequence is then evaluated against v_3, the true label.

As stated earlier, due to the limitations of the baseline models, the training metric for the two models had do be different. Hence, the

metric for evaluating continuous data is the mean average error between the predicted and true values, and accuracy of the cross entropy between predicted and true labels.

4.4.2.2 *Discriminative task evaluation*

We measured the classification accuracy of a discriminator trained to classify the real and fake samples. For this, we used CNN models, each with two convolutional layers coupled with an MLP, which were trained using datasets made up of an equal number of synthetic samples as actual samples, together with a set of labels containing 1s for fake samples and 0s for real ones. The models were then tasked to classify fake samples from real ones, and the accuracy of this classification was then measured.

In an ideal world where a properly trained GAN network managed to capture the original data's distribution precisely, a classification model should yield an accuracy score of 0.5, meaning that the synthetic dataset is so similar to the real dataset that the model is essentially guessing.

4.4.3 *Privacy Evaluation*

The data in an EHR is inherently susceptible and confidential and could lead to real-world consequences if not handled correctly. Probing the model with attacks and scoring how well it does is an important metric to understand how susceptible TOP-GAN is to probing attacks. One type of attack is called a membership inference attack.

4.4.3.1 *Membership inference attacks*

A membership inference attack (MIA) is a privacy threat in machine learning where an adversary tries to determine if a specific data record from one dataset is a member of another. This type of attack, if successful, can reveal sensitive information and potentially lead to privacy breaches, mainly when the data includes personal or confidential information. Not only that, but a successful attack can also create serious mistrust between the data handler and the affected individual [22].

We applied the membership inference attack evaluation technique developed by Yan Chao et al. [19]. In their open-source script, they make some assumptions, such as that said adversary is already in possession of the synthetic data and accompanying features. The resulting output takes the form of a risk score, represented by the F1 score. This evaluation provides a metric to gauge how closely the synthetic data resembles the real data.

The attack procedure revolves around the Euclidean distance between the targeted and synthetic records and involves the following steps:

- The distances between the sets of records is calculated.
- A distance threshold is set depending on the sensitivity of the test.
- By using this threshold, the adversary can assert that a targeted record is part of the real dataset if at least one other record exist with a Euclidean distance lower than the threshold.
- With this set of targeted records and their inferred membership to the real dataset, the F1 score is calculated. The lower the score, the lower the risk measure.

5 RESULTS

During the training phase of TOP-GAN, each encoder and decoder incorporated a single-layer Recurrent Neural Network (RNN) architecture. To ensure efficient learning and model stability, we adopted a batch size of 32 and a learning rate of 0.001. The loss weights

$$\beta_0, \beta_1, \beta_2, \beta_3, \beta_4$$

were set to

$$0.7, 0.7, 0.7, 1.0, 0.7$$

respectively, to put more weight on the code generation. No specific illness cohort was chosen from the MIMIC-III dataset, with the only two criteria being patients with at least one visit and one diagnosis per visit and patients younger than 89.

The main focus during hyper-parameter tuning was the size of the shared latent space, which had the most significant impact on the performance of the model. 4 different sizes were trained and evaluated; these are $S = \{32, 64, 128, 256\}$ and are shown in the results as TOP-GAN$_{s \in S}$.

5.1 FIDELITY EVALUATION RESULTS

5.1.1 *Number of Occurrences of Specific Elements*

To evaluate the frequency distribution of elements in the real and generated data, we counted the occurrences of specific elements and compared these counts. If N_i represents the number of occurrences of an element i in the dataset, we plot N_i for both real and generated data. Plotting this histogram to visualize these frequencies allows us to gain an intuitive feel of how well the model replicates the distribution of specific codes.

In addition to visual inspection, measuring the similarity between the real and synthetic frequency distributions with JS-divergence will provide a quantifiable measurement of how well TOP-GAN's generation of medical codes performs.

Table 2 summarizes the results of the cosine similarity and Jensen-Shannon divergence scores for the real data and four different sets of generated data:

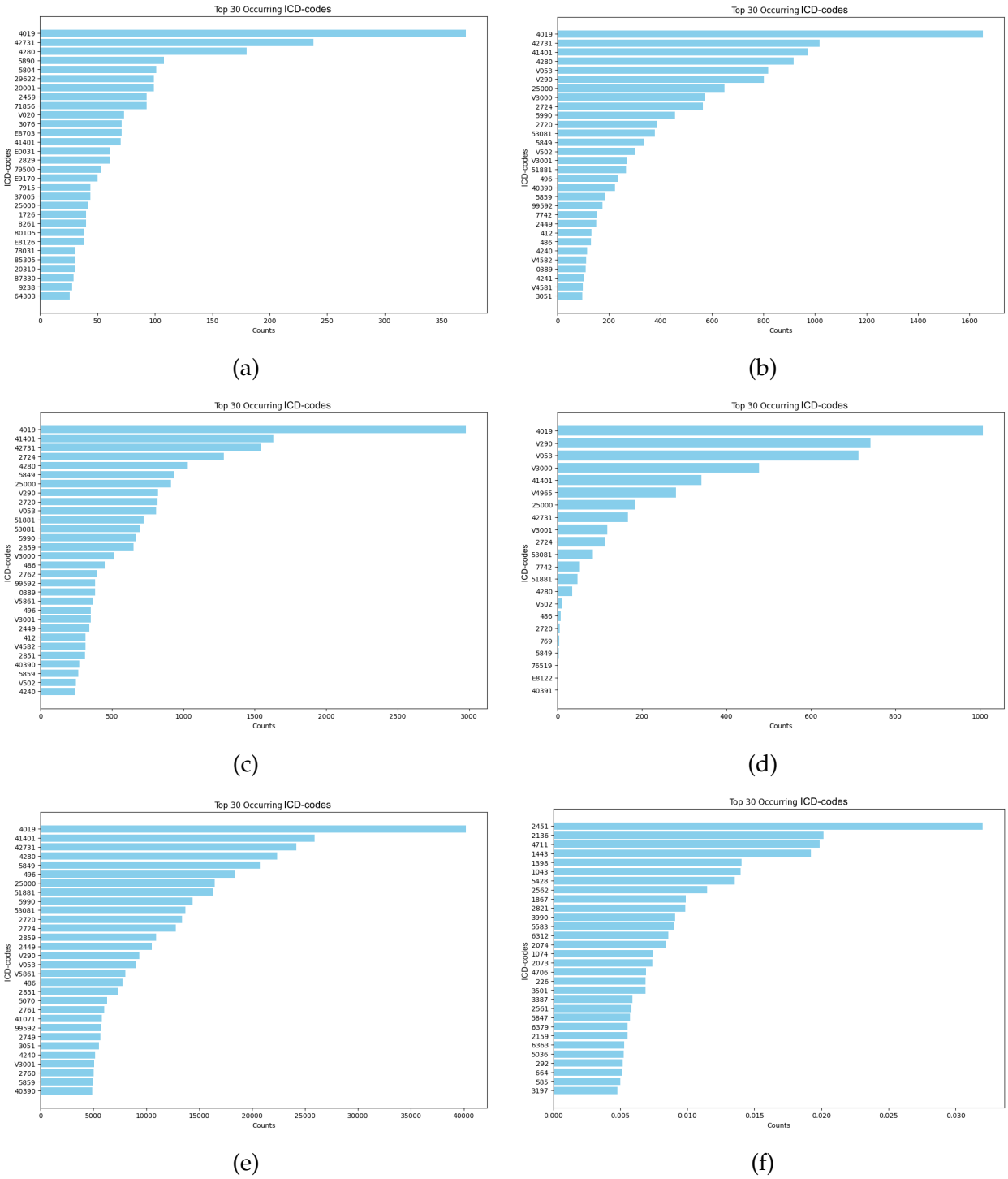

Figure 4: The frequency distributions of the top 30 codes generated from TOP-GAN using 4 different sizes on the latent space as well as the distributions from DAAE and the MIMIC dataset. (a): 32, (b): 64, (c): 128, (d): 256, (e): DAAE, (f): MIMIC

We can see that DAAE performs best compared to the different TOP-GAN configurations, with TOP-GAN$_{128}$ being close behind, sug-

	Cosine similarity	Jensen-Shannon divergence
TOP-GAN$_{32}$	0.55285	0.53346
TOP-GAN$_{64}$	0.86352	0.25817
TOP-GAN$_{128}$	0.93219	0.12396
TOP-GAN$_{256}$	0.57287	0.48241
DAAE	**0.96618**	**0.04774**

Table 2: Cosine similarity and JS-divergence analysis of the frequency distributions between real and generated datasets.

gesting that while the additional information that gets encoded into the shared latent space has some impact on the performance, the impact is not significant enough to hinder code generation.

5.1.2 *Statistical Analysis of Continuous Feature Generation*

As stated in 4, the difference between mean and variance of the real and generated datasets where calculated. This was repeated multiple times for each of the three numerical features and then averaged. Table 3 below summarizes these statistical values. Additionally, the pairplots in figure 5 show the distribution of the numeric features as well as the correlation between features for each of the TOP-GAN configurations.

Feature	**Statistic**	**TOP-GAN32**	**TOP-GAN$_{64}$**	**TOP-GAN$_{128}$**	**TOP-GAN$_{256}$**
Age	Mean	0.80198	0.093192	3.39428	3.39627
	Standard deviation	20.27060	16.98430	10.06218	10.08875
Days	Mean	41.9580	43.95078	42.35176	42.35212
	Standard deviation	170.29931	138.99701	160.4915	160.49146
Heart rate	Mean	3.27705	1.53207	10.21289	10.29305
	Standard deviation	9.35040	7.53252	2.75613	2.70890

Table 3: Mean and standard deviation difference for age, days, and heart rate.

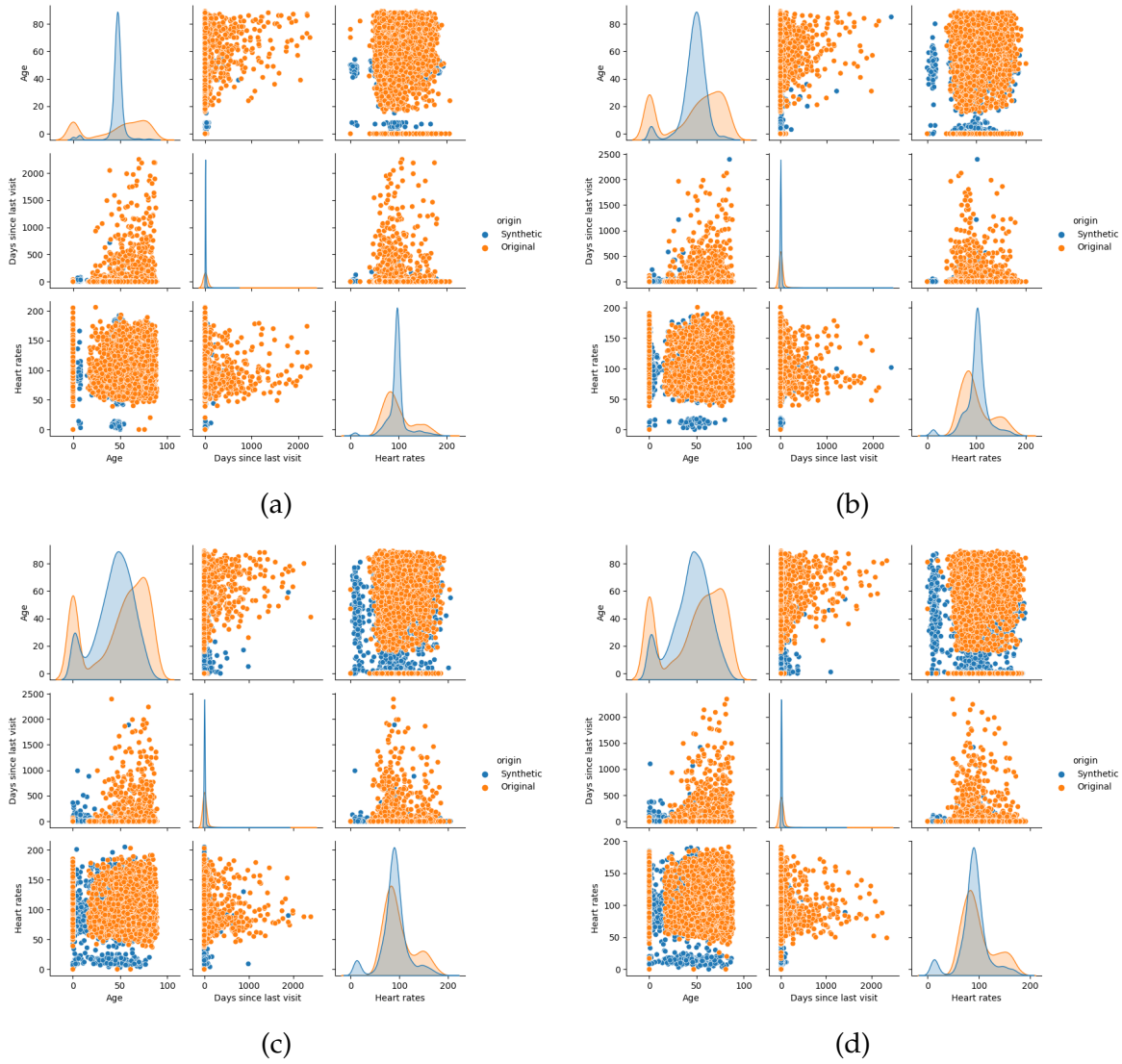

Figure 5: The pairplots of age, days_since and heart rates generated from TOP-GAN using 4 different sizes on the latent space. (a): 32, (b): 64, (c): 128, (d): 256

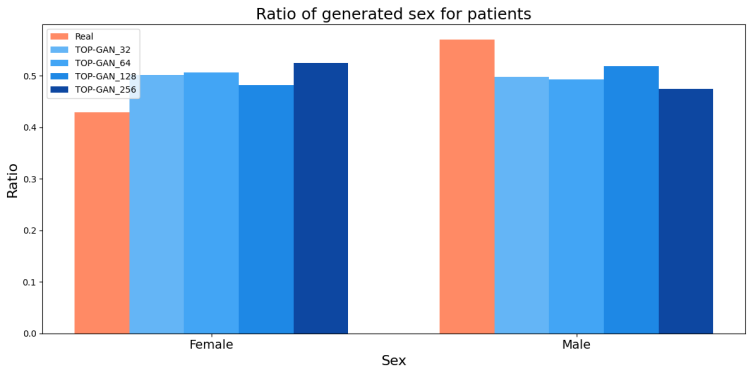

Figure 6: The ratio of the generated sex

The lengths of both the inter- and the intra-visit sequences, i.e. the number of visits and the number of diagnosis codes, respectively, can be found being represented in both figure 7 & 8 as well as in table 4 & 5.

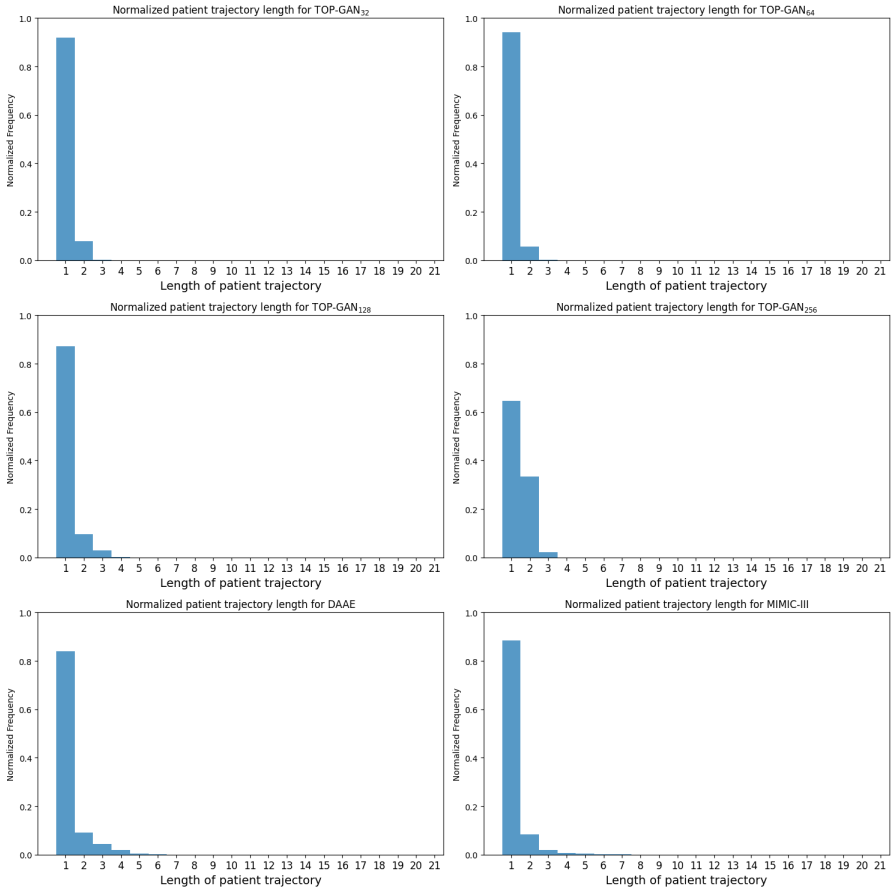

Figure 7: The distributions of patent trajectory lengths

Statistic	32	64	128	256	DAAE	Real data
Mean	1.08274	1.06418	1.16117	1.37589	1.26369	1.18428
Standard deviation	0.28131	0.30199	0.45533	0.52678	0.71328	0.77098

Table 4: The mean and standard deviation of the inter-visit sequence lengths.

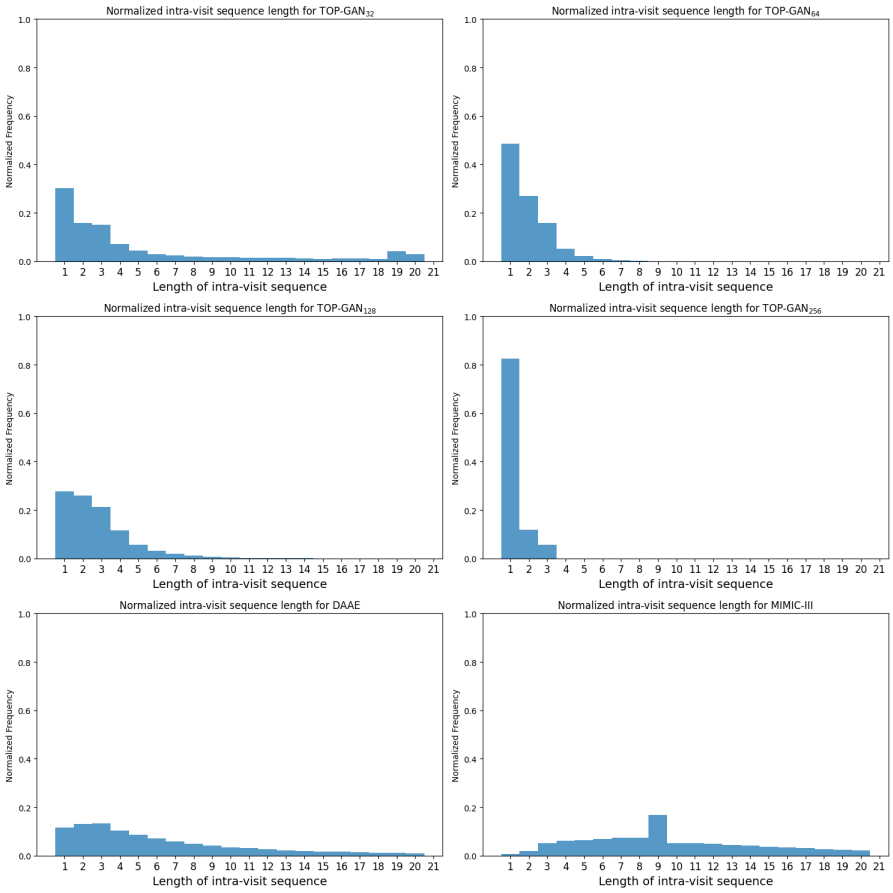

Figure 8: The distributions of length of diagnosis code sequences

Statistic	32	64	128	256	DAAE	Real data
Mean	2.67423	1.15453	2.29602	1.24464	7.46386	9.7960
Standard deviation	4.79091	1.29544	2.01484	1.14010	8.16346	5.8326

Table 5: The mean and standard deviation of the intra-visit sequence lengths.

5.2 UTILITY EVALUATION RESULTS

This chapter is divided up into three main subchapters for each of the three evaluation metrics mentioned in chapter 4. Be aware that, as was stated in 4.4.2, the evaluations had to be split in order accommodate for the different baselines.

5.2.1 *Predictive model Results*

The results of the predictive tests is shown in table 6. The training metric used in the continuous model is the MAE score and accuracy in multi-label classification for the set-valued model.

Evaluation metric	Dataset	Metric score
Accuracy	**Real data**	0.188 ±0.078
	TOP-GAN$_{32}$	0.232 ±0.064
	TOP-GAN$_{64}$	0.268 ±0.092
	TOP-GAN$_{128}$	**0.274 ±0.0687**
	TOP-GAN$_{256}$	0.1432 ±0.0854
	DAAE	0.191 ±0.082
MAE	**Real data**	0.1053 ±0.098
	TOP-GAN$_{32}$	0.0742 ±0.058
	TOP-GAN$_{64}$	**0.0584 ±0.045**
	TOP-GAN$_{128}$	0.1053 ±0.065
	TOP-GAN$_{256}$	0.2444 ±0.120
	TimeGAN	0.1562 ±0.087

Table 6: Performance metrics of the predictive network on real and generated data for set-valued and continuous data.

5.2.2 *Discriminative model Results*

Like the predictive test, the evaluation results for the discriminative evaluation are split between the two baselines. The measured accuracy for both the continuous values and the diagnosis codes can be found in table 7.

Data type	Model	Accuracy
Continuous Values	**TOP-GAN$_{32}$**	0.77465 ±0.1825
	TOP-GAN$_{64}$	0.73495 ±0.17677
	TOP-GAN$_{128}$	**0.7135 ±0.1854**
	TOP-GAN$_{256}$	0.7419 ±0.1717
	TimeGAN	0.79485 ±0.0967
Diagnosis Codes	**TOP-GAN$_{32}$**	0.63065 ±0.0354
	TOP-GAN$_{64}$	0.66949 ±0.1450
	TOP-GAN$_{128}$	0.5812 ±0.1359
	TOP-GAN$_{256}$	**0.53445 ±0.0814**
	DAAE	0.6780 ±0.0663

Table 7: Performance metrics of the discriminative network on real and generated continuous values and diagnosis codes.

5.3 PRIVACY EVALUATION RESULTS

The Membership inference script by Yan C et. al. [19] was setup using a distance threshold of 2, similar to their setup in their main experiment. The resulting scores in table 8 is the average risk score after 5 runs.

Data type	Model	Accuracy
Continuous Values	**TOP-GAN$_{32}$**	0.2370 ±0.0354
	TOP-GAN$_{64}$	0.2003 ±0.0293
	TOP-GAN$_{128}$	**0.1833 ±0.0111**
	TOP-GAN$_{256}$	0.2136 ±0.0215
	TimeGAN	0.1889 ±0.0342
Diagnosis Codes	**TOP-GAN$_{32}$**	0.3580 ±0.1307
	TOP-GAN$_{64}$	0.2955 ±0.1222
	TOP-GAN$_{128}$	**0.2535 ±0.1190**
	TOP-GAN$_{256}$	0.2823 ±0.1334
	DAAE	0.2698 ±0.1276

Table 8: Performance metrics of membership inference attacks on real and generated continuous values and diagnosis codes.

6 DISCUSSION

6.1 ANALYSIS OF THE RESULTS

The results show that TOP-GAN can perform comparably to the other baselines. Looking at the frequency distribution from the code generation and comparing it to the actual dataset, we can see that while TOP-GAN is not able to reach the cosine similarity completely and JS-divergence scores from DAAE [12], it was very close, which could be a strong indication that given further fine-tuning of TOP-GAN in order for it to yield better results is possible given more time.

This comparable performance from TOP-GAN continues into the utility evaluation, where in some of the tests, it even performs slightly better than the baseline models, depending on the latent size.

However, looking at the distribution of the intra- and inter-visit generation lengths and pair plot figures in Chapter 5, we can see some subpar performance from TOP-GAN. While the distributions for age and heart rate were captured relatively well for TOP-GAN$_{128}$ and TOP-GAN$_{256}$, It had difficulty capturing the variability in the *Days_ since* feature instead opting to over-representation thus leading to the massive spike. In the statistical analysis of the lengths of the visits, we can see that while it was able to match roughly the average amount of visits as well as the median compared to the actual data, it was not able to adequately capture the patients whose number of visits are much higher than the average patient, thus leading to a comparably narrow distribution. In addition, TOP-GAN also seemed to suffer from generating sequences during the intra-visit generation compared to the original data and DAAE [12]. The most likely reason for this is the multiple losses affecting the training of the shared latent space, but further testing, such as a proper ablation study to systematically remove and alter loss functions to observe their individual impact, would be required to ensure this is the case.

One hyperparameter, in particular, had a tremendous impact on the model's performance and is the dimension of the shared latent space, as shown in the pairplots and the cosine similarity and JS-divergence. Having too small or too large of a latent space would result in subpar performance, as the model had difficulty converging during training. Finding a sweet spot for the shared latent space is difficult due to the many different losses used to train TOP-GAN, which is reflected in

how the best-performing model between the tests uses different sizes.

6.2 CONCLUSION

Thinking back to the research questions and goals in 1, were they achieved? For the goals:

- Propose a novel system of models for generating synthetic EHR data.
- The model should not only be able to handle irregularity in the temporal dimension but also be able to learn the temporal dependencies among heterogeneous features.

Both of these goals were accomplished. While there are other GAN models that 1.) can handle the multi-modal data found in EHRs, 2.) handle the irregularity aspect of the data within EHRs, to the best of our knowledge, there are none that can handle both while at the same time allowing the generation of an entire patient trajectory and instead being locked to only one visit.

Being able to generate not only an entire trajectory of visits for a patient but also allowing each patient to have varying amounts of visits and each visit having a varying amount of diagnosis codes allows TOP-GAN to fill a hole left by other models of today.

The questions:

- Can the proposed system generate new, authentic-looking synthetic patient trajectories?
- The data is highly irregular both in the number of visits given an interval but also in the number of medical events reported at encounter level, e.g., lab measurements, vital signs, medications, etc... Is this irregularity possible to capture?

The answer to the first question is yes. However, while TOP-GAN was able to generate authentic-looking data based on the metrics setup, it is not guaranteed that this will be the same given other evaluation metrics. More rigorous testing needs to be performed to properly evaluate that, which was not possible given the scope of this thesis.

The answer to the second question is also that it depends. As stated earlier, the distribution of the intra- and inter-visit generation lengths is very subpar compared to the original dataset. While it was able to match the mean of the inter-visit length, it's far narrower than the

original data, as well as the DAAE baseline and the intra-visit generation is comparably a lot shorter than the baselines.

6.3 FUTURE WORK

The most significant limitation of this thesis is the need for domain knowledge, and due to the scope of this thesis, recruiting external domain experts was not possible. While TOP-GAN is able to learn the underlying distribution of the data thanks to its GAN architecture, it is unable to rule out properly disregard generated samples that contain medicinal violations or other impossible combinations. One such combination that was observed was a generated male patient suffering from the diagnosis code 616.0, Cervicitis. Using domain expertise to create a learning objective that utilizes some degree of violation penalty could potentially yield better results, making the model generate more authentic-looking diagnoses. It could also be an exciting way to build upon this work.

Another path of improvement could be regarding TOP-GAN's inability to capture the outlier patients with large amounts of visits properly.

BIBLIOGRAPHY

[1] Regulation on the protection of natural persons with regard to the processing of personal data and on the free movement of such data., Apr 2016. URL https://eur-lex.europa.eu/eli/reg/2016/679/oj.

[2] Summary of the hipaa privacy rule, Oct 2022. URL https://www.hhs.gov/hipaa/for-professionals/privacy/laws-regulations/index.html.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

[4] Nduma N Basil, Solomon Ambe, Chukwuyem Ekhator, Ekokobe Fonkem, Basil N Nduma, and Chukwuyem Ekhator. Health records database and inherent security concerns: A review of the literature. *Cureus*, 14(10), 2022.

[5] Anthony L. Caterini and Dong Eui Chang. *Recurrent Neural Networks*, pages 59–79. Springer International Publishing, Cham, 2018. ISBN 978-3-319-75304-1. doi: 10.1007/978-3-319-75304-1_5. URL https://doi.org/10.1007/978-3-319-75304-1_5.

[6] Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F Stewart, and Jimeng Sun. Generating multi-label discrete patient records using generative adversarial networks. In *Machine learning for healthcare conference*, pages 286–305. PMLR, 2017.

[7] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[8] Atiye Sadat Hashemi, Kobra Etminani, Amira Soliman, Omar Hamed, and Jens Lundström. Time-series anonymization of tabular health data using generative adversarial network. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2023. doi: 10.1109/IJCNN54540.2023.10191367.

[9] Atiye Sadat Hashemi, Omar Hamed, Jens LundstrÃ¶m, Amira Soliman, and Kobra Etminani. Domain knowledge-driven generation of synthetic healthcare data, 2023. URL https://pubmed.ncbi.nlm.nih.gov/37203680/.

[10] Alistair Johnson, Tom Pollard, and Roger Mark. Mimic-iii clinical database, 2023. URL https://physionet.org/content/mimiciii/1.4/.

[11] Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3(1), May 2016. ISSN 2052-4463. doi: 10.1038/sdata.2016.35. URL http://dx.doi.org/10.1038/sdata.2016.35.

[12] Dongha Lee, Hwanjo Yu, Xiaoqian Jiang, Deevakar Rogith, Meghana Gudala, Mubeen Tejani, Qiuchen Zhang, and Li Xiong. Generating sequential electronic health records using dual adversarial autoencoder. *Journal of the American Medical Informatics Association*, 27(9):1411–1419, 09 2020. ISSN 1527-974X. doi: 10.1093/jamia/ocaa119. URL https://doi.org/10.1093/jamia/ocaa119.

[13] Chang Lu, Chandan K. Reddy, Ping Wang, Dong Nie, and Yue Ning. Multi-label clinical time-series generation via conditional gan. *IEEE Transactions on Knowledge and Data Engineering*, 36(4):1728â1740, April 2024. ISSN 2326-3865. doi: 10.1109/tkde.2023.3310909. URL http://dx.doi.org/10.1109/TKDE.2023.3310909.

[14] Giorgia Ramponi, Pavlos Protopapas, Marco Brambilla, and Ryan Janssen. T-CGAN: conditional generative adversarial network for data augmentation in noisy time series with irregular sampling. *CoRR*, abs/1811.08295, 2018. URL http://arxiv.org/abs/1811.08295.

[15] Lior Rokach, Oded Maimon, and Erez Shmueli. *Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook*. Springer, 2023.

[16] Chenxi Sun, Shenda Hong, Moxian Song, and Hongyan Li. A review of deep learning methods for irregularly sampled medical time series data. *arXiv preprint arXiv:2010.12493*, 2020.

[17] Lilian Weng. From gan to wgan, 2019. URL https://arxiv.org/abs/1904.08994.

[18] Chao Yan, Ziqi Zhang, Steve Nyemba, and Bradley A Malin. Generating electronic health records with multiple data types and constraints. In *AMIA annual symposium proceedings*, volume 2020, page 1335. American Medical Informatics Association, 2020.

[19] Chao Yan, Yao Yan, Zhiyu Wan, Ziqi Zhang, Larsson Omberg, Justin Guinney, Sean D. Mooney, and Bradley A. Malin. A multifaceted benchmarking of synthetic electronic health record generation models. *Nature Communications*, 13(1), December

2022. ISSN 2041-1723. doi: 10.1038/s41467-022-35295-1. URL http://dx.doi.org/10.1038/s41467-022-35295-1.

[20] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

[21] Jinsung Yoon, Michel Mizrahi, Nahid Farhady Ghalaty, Thomas Jarvinen, Ashwin S. Ravi, Peter Brune, Fanyu Kong, Dave Anderson, George Lee, Arie Meir, and et al. Ehr-safe: Generating high-fidelity and privacy-preserving synthetic electronic health records. *npj Digital Medicine*, 6(1), Aug 2023. doi: 10.1038/s41746-023-00888-7.

[22] Ziqi Zhang, Chao Yan, and Bradley A. Malin. Membership inference attacks against synthetic health data. *Journal of Biomedical Informatics*, 125:103977, January 2022. ISSN 1532-0464. doi: 10.1016/j.jbi.2021.103977. URL http://dx.doi.org/10.1016/j.jbi.2021.103977.

.

PO Box 823, SE-301 18 Halmstad
Phone: +35 46 16 71 00
E-mail: registrator@hh.se
www.hh.se