SynEHRgy: Synthesizing Mixed-Type Structured Electronic Health Records using Decoder-Only Transformers

Hojjat Karami
EPFL
hojjat.karami@epfl.ch

David Atienza
EPFL
david.atienza@epfl.ch

Anisoara Ionescu
EPFL
anisoara.ionescu@epfl.ch

Abstract

Generating synthetic Electronic Health Records (EHRs) offers significant potential for data augmentation, privacy-preserving data sharing, and improving machine learning model training. We propose a novel tokenization strategy tailored for structured EHR data, which encompasses diverse data types such as covariates, ICD codes, and irregularly sampled time series. Using a GPT-like decoder-only transformer model, we demonstrate the generation of high-quality synthetic EHRs. Our approach is evaluated using the MIMIC-III dataset, and we benchmark the fidelity, utility, and privacy of the generated data against state-of-the-art models. The code is available at `https://github.com/hojjatkarami/SynEHRgy`.

1 Introduction

Electronic Health Records (EHRs) are comprehensive digital repositories of patient health information, encompassing a wide range of data types. In in-patient settings, EHRs include structured data such as demographics and International Classification of Diseases (ICD) codes, time series data (e.g., vital signs and lab results), and unstructured data (e.g., clinical notes and radiology images). These records provide an invaluable resource for developing machine learning models, offering rich datasets that can be harnessed to enhance patient care, predict outcomes, and support clinical decision-making.

Access to real-world EHRs is often limited due to privacy regulations such as HIPAA and GDPR, as well as technical challenges and a lack of incentives for data sharing. Even pseudo-anonymized data remains susceptible to re-identification by malicious actors. Techniques like federated learning [1] and differential privacy [2] are being explored to mitigate these risks. Synthetic data provides a promising alternative by replicating the statistical properties of real data without being tied to actual individuals, thereby circumventing privacy regulations. It can be securely shared among stakeholders, facilitating hypothesis generation, AI model pre-training, and data exploration without exposing sensitive information. This approach not only alleviates privacy concerns but also reduces the effort involved in accessing real data, allowing researchers to focus on validating more probable hypotheses before turning to original datasets for confirmation.

Generating synthetic EHR data presents several challenges. EHRs comprise multiple data types with distinct characteristics. For instance, clinical events like ICD codes are high-dimensional and sequential, while time series data are relatively low-dimensional but feature irregular sampling and significant missingness, which is often informative and not random [3, 4]. Additionally, patients may have multiple visits or admissions, with data across visits being correlated. While many studies focus on generating single data types, such as discrete ICD codes ([5, 6, 7, 8]) or time series data ([9, 10, 11]), few addresses the generation of multiple data types across multiple visits simultaneously ([12]).

In the era of Large Language Models (LLMs), many have been adapted to the clinical domain for tasks such as diagnosis prediction [13], text embeddings [14], and medical reasoning [15, 16]. While

GenAI for Health Workshop @ NeurIPS 2024, Vancouver.

these models can interpret structured data and numerical values presented in text format, they are not directly applicable for generating synthetic structured EHR data. This limitation arises because LLMs are typically not trained on large structured EHR datasets, particularly those containing complex numerical data.

In this work, we present **SynEHRgy**, a framework designed to **Syn**thesize mixed-type structured **EHR**s, including covariates, ICD codes, and irregularly-sampled time series across multiple patient visits. We introduce a novel tokenization strategy for structured EHR data, particularly for numerical variables, allowing decoder-only transformer models to capture underlying data patterns within a causal language modeling framework and generate high-quality synthetic EHRs. Using a small GPT model, we demonstrate the efficacy of our approach on the MIMIC-III dataset [17], comparing the generated data's quality against state-of-the-art models in terms of fidelity, utility, and privacy. The key contributions of this work are as follows:

- We propose a novel tokenization strategy for mixed-type structured EHR data (covariates, ICD codes, irregularly-sampled time series) across multiple visits.
- We empirically demonstrate that a GPT-like decoder-only transformer model can generate high-quality structured EHR data.
- We conduct a comprehensive evaluation of the proposed methodology on the MIMIC-III dataset, comparing the generated data's quality to state-of-the-art models in terms of fidelity, utility, and privacy.
- Our framework shows exceptional performance in generating irregularly-sampled time series data, a challenging task due to high missingness and irregular time points.

2 Related Works

Generating clinical events, particularly ICD codes, has garnered significant research interest in recent years. In contrast, generating clinical time series presents a greater challenge due to irregularly-sampled time points and substantial missingness.

ICD Code Generation A substantial body of work focuses on generating ICD codes using various frameworks. Generative Adversarial Networks (GANs) have been employed for this purpose ([18, 6, 19, 20]), as have Variational Autoencoders (VAEs) ([21]) and diffusion models ([22, 23, 24, 25]). Additionally, Large Language Models (LLMs) have been explored due to the sequential nature of ICD codes and the feasibility of straightforward tokenization. For instance, PromptEHR [8] employs an encoder-decoder transformer model with prompt learning to generate ICD codes conditioned on numerical and categorical demographic features. HALO [12] utilizes a transformer encoder coupled with a linear autoregressive module for generating ICD codes at both the visit and code levels. CEHR-GPT [26] leverages GPT to produce sequences of visits based on demographic prompts and visit time intervals.

Time Series Generation Research on time series generation has predominantly focused on regularly-sampled time series, which are more common across various domains. Techniques such as Generative Adversarial Networks (GANs) have been widely applied ([27, 28, 29, 30, 31]), as well as diffusion models ([32, 33, 34]). However, generating irregularly-sampled time series is less explored. Notable efforts in this area include RTSGAN [9], EHR-Safe [31], and TimEHR [11] using GANs, and TS-Diffusion [35] and TimeDiff [10] using diffusion models. Additionally, research on leveraging language models for time series generation remains limited. While pre-trained LLMs have demonstrated effectiveness with numerical values in text format—e.g., LIFT [36] and GReaT [37] excel in tabular data generation—they primarily treat numerical values as text. Some approaches propose defining specific numerical tokens and training LLMs from scratch. For example, CodeAR [38] utilizes a Vector Quantized Autoencoder (VQ-VAE) for codebook creation, followed by an LSTM with next-token prediction loss. Chronos [39] discretizes numerical values into bins for time series forecasting using encoder-decoder transformers, a strategy also employed by HALO [12] for clinical time series and by [40] for financial time series. To the best of our knowledge, no existing work, other than HALO [12], explores the generation of irregularly-sampled time series using LLMs.

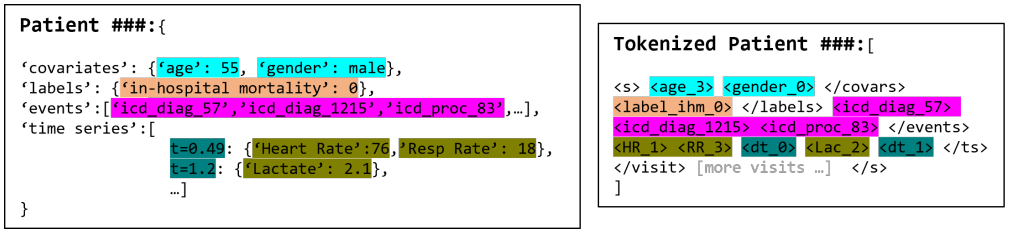

Figure 1: Left: Example of patient's data. Right: Tokenized patient's data

3 Methodology

3.1 Problem Formulation

An EHR dataset can be characterized as $\mathcal{D} = \{P_i\}_{i=1}^{N}$, where N denotes the number of patients and P_i represents the data for the i-th patient. Each patient may have one or more visits, denoted as $P_i = \{V_{ij}\}_{j=1}^{|V_i|}$, with each visit represented as $V_j = \{c_j, y_j, E_j, T_j\}$. Here, c_j denotes the set of covariates (e.g., demographics), y_j represents the set of clinical outcomes (e.g., in-hospital mortality), $E_j = \{e_m\}_{m=1}^{|E_j|}$ is a sequence of clinical events such as ICD codes, and T_j is the time series data, including vital signs and laboratory variables. The time series data for each visit is represented as $T_j = \{t_k, \{(n_{kp}, x_{kp})\}_{p=1}^{|L_k|}\}_{k=1}^{|T_j|}$, where $|T_j|$ denotes the total number of measurement times, and $|L_k|$ indicates the number of available data points per measurement. In this representation, $t_k \in \mathbb{R} \geq 0$ is the timestamp of the k-th measurement, while n_{kp} and x_{kp} are the variable name and value pairs. An example of patient data is illustrated on the left side of Figure 1.

3.2 Tokenization

For numerical values, we employ uniform quantization to discretize numerical variables into equal-sized bins, similar to Chronos [39]. Each bin is assigned a unique token. For instance, the token <HR_0> represents heart rate values in the range of 55-66 beats per minute. For timestamps, we compute the time intervals between consecutive measurements and discretize these intervals into bins. For non-numerical values such as ICD codes and categorical variables, each code or category is assigned a unique token.

We also use special tokens to delineate different sections of the data: <s> and </s> for the start and end of a sequence, respectively; </covars>, </labels>, </ts>, and </visit> to signify the end of covariates, labels, time series, and admissions. The padding token (<PAD>) is employed to ensure that all tokenized sequences are of uniform length. If a patient has multiple admissions, the data for new admissions is appended after the token </visit>. An example of tokenized patient data is illustrated on the right side of Figure 1.

This tokenization strategy offers several advantages: First, it is scalable with respect to the number of variables. For instance, when integrating a new dataset with additional variables, we simply update the tokenization dictionary by incorporating new tokens. Second, numerical tokenization is more efficient than treating numerical values as text. For example, the GPT-2 tokenizer would generate seven tokens for 'Lactate:2.5', whereas our method generates only one token.

3.3 Model Architecture

Decoder-only Transformer models, such as the ones used in language modeling (e.g., GPT [41]), typically employ a causal language modeling (CLM) objective as their loss function. This objective is also known as autoregressive language modeling. The core idea is to predict the next token in a sequence given all previous tokens. The model generates a probability distribution over the vocabulary for the next token, and the loss function measures the discrepancy between the predicted distribution and the actual next token. For an tokenized patient sequence $x = [x_1, x_2, \ldots, x_T]$, the model predicts the probability distribution over the vocabulary for each token in the sequence, conditioned on all previous tokens:

$$p(x_t \mid x_1, x_2, \ldots, x_{t-1})$$

The loss for a single token prediction is typically computed using cross-entropy between the predicted distribution and the one-hot encoded true next token. For the entire sequence, the loss function is:

$$\mathcal{L} = -\sum_{t=1}^{T} \log p(x_t \mid x_1, x_2, \ldots, x_{t-1})$$

3.4 EHR Data Generation

The generation of EHR data follows a process similar to text generation. We initiate the generation with the start token (<s>) and continue generating tokens sequentially until the end token (</s>) is produced. For the de-tokenization of numerical tokens, we apply uniform sampling within the specified bin range to recover the original numerical values. This approach ensures that the generated values are representative of the underlying distribution of the data. The de-tokenization of ICD codes and categorical variables is straightforward, as each token directly maps to a specific code or category.

This method offers several advantages: First, it leverages the well-established text generation framework, which allows for seamless integration with existing models and techniques. Second, uniform sampling for numerical values maintains the integrity of the data distribution, providing a realistic representation of the original values. Lastly, the straightforward de-tokenization of categorical variables ensures simplicity and efficiency in generating accurate categorical data.

4 Experiments

4.1 Dataset

We utilize the MIMIC-III dataset for our experiments [17]. Specifically, we use the preprocessing pipeline provided by [42] to extract data from approximately 42,000 patients with multiple hospital visits. The dataset is divided into training, validation, and test sets using a 70-15-15 split. For each patient, we include age and gender as covariates. Additionally, for each visit, we select 25 phenotype labels along with an in-hospital mortality label. We include ICD diagnosis and procedure codes with a frequency greater than five across the entire dataset, resulting in 4,656 unique ICD codes. We also select 41 time series from vital signs and laboratory variables, including five categorical time series. After tokenization, we have a total of 5,127 unique tokens, with the training split consisting of 29.6 million tokens. Each patient's sequence starts with the <s> token and ends with the </s> token, comprising demographics, ICD codes, and time series tokens, as explained in Section 3.2.

4.2 Evaluation Metrics

We evaluate the synthetic data based on **Fidelity**, **Utility**, and **Privacy**. Since all of our baselines, except HALO, do not generate multiple data types, we assess our metrics separately for each data type. Additionally, we use the test split as a reference for high-quality synthetic data to evaluate these metrics.

Fidelity For ICD codes, similar to [12], we assess fidelity using unigram, bigram, and trigram probabilities within each visit, as well as sequential bigram probabilities between consecutive visits. For instance, the bigram probability of the sequence $[icd_{1053}, icd_{610}]$ is calculated as its frequency divided by the total number of patients. We report Pearson's correlation between the top-1000 n-gram probabilities of the train and test/synthetic data pairs.

For time series data, we create manual embeddings for each patient by computing common statistics (min, max, mean, std) from the first 48 hours of the patient's stay. We then report the precision, recall, density, and coverage (PRDC) metrics [43] to evaluate the fidelity between the embeddings of the train split and the test/synthetic split. Additionally, we calculate the pairwise temporal correlation between all time series variables by concatenating the time series data from all patients and report the mean squared error between the correlation matrices of the train split and the test/synthetic split (MSE_{corr}). We also assess correlation accuracy [30] by discretizing the correlation coefficients into five levels: High negative ($[-1, -0.5)$), Medium negative ($[-0.5, -0.2)$), Low ($[-0.2, 0.2)$), Medium positive ($[0.2, 0.5)$), and High positive ($[0.5, 1)$). The confusion matrix of these correlation levels is then reported. To assess the fidelity of the missingness patterns, we visualize the co-occurrence of

measurements for each pair of time series variables and normalize this by the sum of occurrences for each variable.

Utility We assess the utility of the synthetic data by evaluating its performance in two tasks: *in-hospital mortality* prediction (binary classification) and *phenotypes* classification (multi-label classification). Specifically, we use time series embeddings from the first 48 hours of patient admission, as described earlier, and train a LightGBM model [44]. To examine the impact of data augmentation with synthetic data, we incorporate the entire synthetic dataset into the training data at various ratios (0, 0.1, 0.2, 0.5, 1.0) and evaluate the model on the test split.

Privacy We adopt the Membership Inference Attack (MIA) approach as described in EHR-Safe [31], where the goal is to determine whether specific data points were included in the training dataset. To accomplish this, we fit a K-Nearest Neighbors (KNN) model on the synthetic data and calculate the nearest distances for each patient in both the training and test splits. A significant disparity between the distance distributions in the synthetic-train and synthetic-test sets indicates lower privacy. For ICD code sequences, we use the Hamming distance, whereas for time series embeddings, we employ the Euclidean distance. We then fit Gaussian distributions to these distances and assess the differences between the two distributions using the Wasserstein Distance (WD), Jensen-Shannon Divergence (JSD), and Area Under the Receiver Operating Characteristic (AUROC) metrics.

4.3 Baselines

For ICD code generation, we compare our method with HALO [12], a hierarchical autoregressive language model, and PromptEHR [8], which utilizes an encoder-decoder transformer model. For time series generation, we compare our method with RTSGAN [9] and TimEHR [11], both of which are GAN-based models designed for generating irregularly-sampled time series data. Additionally, we include HALO in the comparison for time series generation, as it employs a similar tokenization strategy for numerical variables.

Ablation study. We further demonstrate the effectiveness of numerical tokenization by removing the discretization step and treating the numerical values as raw text. We refer to this model as SynEHRgy0.

4.4 Training Details

We employ a small GPT-2 model from the Transformers library[1]. Our model consists of 4 layers, 4 attention heads, and 384 embedding dimensions. After testing various context lengths, we found that 1024 tokens offer a good balance between the quality of generated data and training time (notably, context lengths of 256, 512, 1024, and 2048 correspond to 20%, 34%, 51%, and 68% of the original data, respectively). For the ablated model (SynEHRgy0), we need to use a larger context length of size 4098 due to its inefficient tokenization.

We use the Adam optimizer with a learning rate of 3×10^{-4} and train the model for 20 epochs on a system equipped with an *NVIDIA Tesla V100-SXM2-32GB* GPU. The batch size is set to 128, and we employ gradient accumulation with a step of two. All other hyperparameters are set to their default values as per the Transformers library. For generation, we use *top-k sampling* with a temperature of 0.7 and a top-k value of 50. We generate 30,000 synthetic patients which is the same size of training split. Due to computational limitation, we have performed minimal hyperparameter tuning for our model and for the baselines to ensure a fair comparison.

5 Results

5.1 Fidelity

Table 1 shows the correlation values between n-gram probabilities of the training and test/synthetic splits. While HALO achieves the highest performance for unigram probabilities, our method consistently surpasses all baselines, particularly in bigram, trigram, and sequential bigram probabilities. This underscores the ability of our method to generate high-quality synthetic ICD codes.

[1] https://huggingface.co/docs/transformers

Table 1: Fidelity metrics for ICD codes

	Single-visit			Sequential Bigram
	Unigram	Bigram	Trigram	
Test split	*0.998*	*0.722*	*0.742*	*0.634*
SynEHRgy	**0.987**	**0.729**	**0.786**	**0.568**
HALO	0.885	0.265	0.302	*0.524*
PromptEHR	0.890	0.682	0.669	NA

Table 2: Fidelity metrics for Time Series. **Bold** and underline values indicate the best and second best results, respectively.

	Precision(↑)	Recall(↑)	Density(↑)	Coverage(↑)	MSE_{corr}(↓)
Test split	*0.863 (0.004)*	*0.878 (0.003)*	*0.984 (0.005)*	*0.966 (0.001)*	*0.02*
SynEHRgy	**0.805 (0.006)**	**0.86 (0.003)**	0.705 (0.008)	**0.898 (0.005)**	**0.029**
SynEHRgy0	0.452(.018)	0.227(0.020)	0.345(0.009)	0.150(0.014)	0.133
TimEHR	0.593 (0.008)	0.669 (0.011)	0.385 (0.011)	0.613 (0.003)	0.059
HALO	0.521 (0.021)	0.469 (0.002)	0.387 (0.022)	0.19 (0.003)	0.072
RTSGAN	0.725 (0.026)	0.113 (0.019)	**0.744 (0.054)**	0.108 (0.001)	0.097

Table 2 presents the precision, recall, density, and coverage (PRDC) metrics, as well as the temporal correlation difference (MSE_{corr}) for evaluating the fidelity of time series embeddings. It is important to consider these PRDC metrics collectively. For instance, while RTSGAN achieves the highest density, it performs poorly on recall and coverage metrics. Our method excels in precision, recall, and coverage, and ranks second in density. Additionally, our model demonstrates the lowest MSE_{corr}, indicating strong performance in preserving structural correlations within the data. This is further supported by the confusion matrix of correlation levels shown in Figure 2, where SynEHRgy displays superior performance in capturing correlation levels. Moreover, the co-occurrence analysis in Figure 3 reveals that our method exhibits the highest fidelity in preserving missingness patterns, highlighting its capability to generate realistic irregularly-sampled time series data. The ablated model (SynEHRgy0) exhibits poor fidelity performance, demonstrating the importance of numerical tokenization.

5.2 Utility

Table 3 and Table 4 present the AUROC scores for the 25-phenotype and in-hospital mortality prediction tasks, respectively, across different data augmentation settings. When training exclusively on synthetic data and testing on real data (TSTR), our method surpasses all baseline models and achieves performance closest to that of the Train on Real and Test on Real setting (TRTR). Additionally, our method demonstrates the highest AUROC when synthetic data is incorporated into the training set, although the performance gain diminishes as the proportion of real training data increases. This suggests that while the models leverage the synthetic data effectively, they do not extend beyond the real data distribution to generate new patient profiles that significantly enhance test performance.

Table 3: AUROC for phenotypes prediction. *NA*: Not Applicable.

	Train ratio				
	0 (TSTR)	0.1	0.2	0.5	1
No synthetic	*NA*	*0.769*	*0.78*	*0.793*	*0.802(TRTR)*
Val split	*0.781*	*0.788*	*0.791*	*0.799*	*0.805*
SynEHRgy	**0.787**	**0.789**	**0.793**	**0.798**	**0.804**
TimEHR	0.736	0.762	0.77	0.778	0.784
HALO	0.665	0.746	0.761	0.78	0.787
RTSGAN	0.546	0.764	0.774	0.788	0.799

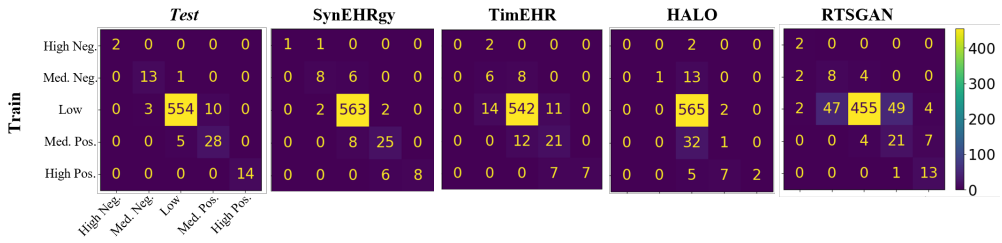

Figure 2: Confusion matrices of correlation levels for time series data

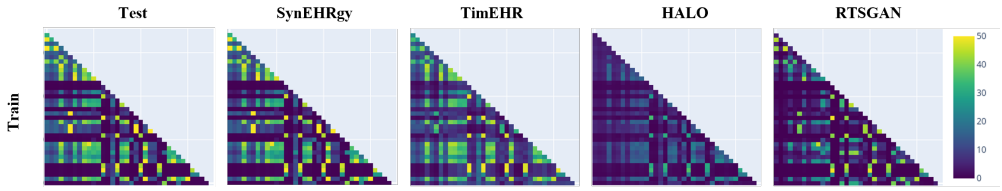

Figure 3: Co-occurrence of measurement for time series data

Furthermore, the validation split shows marginally better performance compared to other synthetic data settings. These utility results highlight the potential of our method for data anonymization (with no exposure of training data) and for data augmentation in scenarios with limited training data.

5.3 Privacy

Membership Inference Attack (MIA) metrics are reported in Table 5. As can be seen, none of methods shows any potential privacy risk. It should be noted that these privacy metrics are computed based on hamming distance of ICD codes and Euclidean distance of time series data, and the results may vary for different distance metrics.

6 Conclusion

In this work, we introduced SynEHRgy, a novel methodology for generating synthetic structured EHR data that incorporates mixed-type data (ICD codes and time series) across multiple visits. A key innovation of our approach is the tokenization strategy for numerical variables, which enhances compatibility with decoder-only transformer models. Despite its advantages, the proposed method has limitations. Specifically, treating each data point (e.g., an ICD code or a heart rate measurement) as a single token may become inefficient for very large sequences due to context length constraints. Future work will explore more efficient tokenization strategies to mitigate this issue. Additionally, integrating high-frequency continuous signals, such as ECG data, is currently not feasible with our tokenization approach. We plan to address this limitation by incorporating patch-based tokenization strategies [45, 39, 46]. Furthermore, we aim to expand our methodology to include other data modalities, such as clinical notes and radiology images, to generate comprehensive multimodal EHR

Table 4: AUROC for in-hospital mortality prediction. *NA*: Not Applicable.

	Train ratio				
	0 (TSTR)	0.1	0.2	0.5	1
No synthetic	*NA*	*0.889*	*0.898*	*0.912*	*0.92 (TRTR)*
Val split	*0.902*	*0.91*	*0.914*	*0.915*	*0.924*
SynEHRgy	**0.901**	**0.906**	**0.905**	**0.913**	0.915
TimEHR	0.865	0.887	0.893	0.906	0.912
RTSGAN	0.58	0.878	0.891	0.906	**0.919**

Table 5: Privacy metrics for ICD codes and Time Series

ICD codes				Time Series			
Model	JSD	WD	AUROC	Model	WD	JSD	AUROC
SynEHRgy	0.014	0.001	0.479	SynEHRgy	0.002	0.001	0.493
HALO	0.013	0.000	0.511	TimEHR	0.003	0.001	0.495
PromptEHR	0.012	0.001	0.455	HALO	0.003	0.001	0.493
				RTSGAN	0.002	0.001	0.494

data. Another promising direction for future research is to enhance the capabilities of clinical LLMs, such as Meditron [16] and Med-PaLM [47], to better understand and generate structured data.

Acknowledgments

This work was supported by the RealCare and DigiPredict project, which has received funding from the European Union and by the Swiss State Secretariat for Education, Research and Innovation (SERI). Additionally, we would like to thank the EPFL IC Cluster for providing the computational resources used in this work.

References

[1] Nicola Rieke et al. "The Future of Digital Health with Federated Learning". In: *npj Digital Medicine* 3.1 (Sept. 2020), p. 119. ISSN: 2398-6352. DOI: 10.1038/s41746-020-00323-1. arXiv: 2003.08119 [cs].

[2] Nazish Khalid et al. "Privacy-Preserving Artificial Intelligence in Healthcare: Techniques and Applications". In: *Computers in Biology and Medicine* 158 (May 2023), p. 106848. ISSN: 0010-4825. DOI: 10.1016/j.compbiomed.2023.106848.

[3] Marzyeh Ghassemi et al. "A Review of Challenges and Opportunities in Machine Learning for Health". In: *AMIA Summits on Translational Science Proceedings* 2020 (May 2020), pp. 191–200. ISSN: 2153-4063.

[4] Amelia L. M. Tan et al. "Informative Missingness: What Can We Learn from Patterns in Missing Laboratory Data in the Electronic Health Record?" In: *Journal of Biomedical Informatics* 139 (Mar. 2023), p. 104306. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2023.104306.

[5] Edward Choi et al. "Generating Multi-label Discrete Patient Records Using Generative Adversarial Networks". In: *arXiv: Learning* (Mar. 2017).

[6] Amirsina Torfi and Edward A. Fox. *CorGAN: Correlation-Capturing Convolutional Generative Adversarial Networks for Generating Synthetic Healthcare Records*. Mar. 2020. DOI: 10.48550/arXiv.2001.09346. arXiv: 2001.09346 [cs, stat].

[7] Siddharth Biswal et al. *EVA: Generating Longitudinal Electronic Health Records Using Conditional Variational Autoencoders*. Dec. 2020. DOI: 10.48550/arXiv.2012.10020. arXiv: 2012.10020 [cs].

[8] Zifeng Wang and Jimeng Sun. *PromptEHR: Conditional Electronic Healthcare Records Generation with Prompt Learning*. Oct. 2022. DOI: 10.48550/arXiv.2211.01761. arXiv: 2211.01761 [cs].

[9] Hengzhi Pei et al. "Towards Generating Real-World Time Series Data". In: (Dec. 2021). DOI: 10.1109/ICDM51629.2021.00058.

[10] Muhang Tian et al. *Fast and Reliable Generation of EHR Time Series via Diffusion Models*. Oct. 2023. DOI: 10.48550/arXiv.2310.15290. arXiv: 2310.15290 [cs].

[11] Hojjat Karami et al. *TimEHR: Image-based Time Series Generation for Electronic Health Records*. Feb. 2024. DOI: 10.48550/arXiv.2402.06318. arXiv: 2402.06318 [cs].

[12] Brandon Theodorou, Cao Xiao, and Jimeng Sun. "Synthesize High-Dimensional Longitudinal Electronic Health Records via Hierarchical Autoregressive Language Model". In: *Nature Communications* 14.1 (Aug. 2023), p. 5305. ISSN: 2041-1723. DOI: 10.1038/s41467-023-41093-0.

[13] Yikuan Li et al. *BEHRT: Transformer for Electronic Health Records*. July 2019. DOI: 10.48550/arXiv.1907.09538. arXiv: 1907.09538 [cs, stat].

[14] Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. *ClinicalBERT: Modeling Clinical Notes and Predicting Hospital Readmission*. Nov. 2020. DOI: 10.48550/arXiv.1904.05342. arXiv: 1904.05342 [cs].

[15] Karan Singhal et al. *Towards Expert-Level Medical Question Answering with Large Language Models*. May 2023. DOI: 10.48550/arXiv.2305.09617. arXiv: 2305.09617 [cs].

[16] Zeming Chen et al. *MEDITRON-70B: Scaling Medical Pretraining for Large Language Models*. Nov. 2023. DOI: 10.48550/arXiv.2311.16079. arXiv: 2311.16079 [cs].

[17] Alistair E. W. Johnson et al. "MIMIC-III, a Freely Accessible Critical Care Database". In: *Scientific Data* 3.1 (May 2016), p. 160035. ISSN: 2052-4463. DOI: 10.1038/sdata.2016.35.

[18] Mrinal Kanti Baowaly et al. "Synthesizing Electronic Health Records Using Improved Generative Adversarial Networks". In: *Journal of the American Medical Informatics Association* 26.3 (Mar. 2019), pp. 228–241. ISSN: 1067-5027, 1527-974X. DOI: 10.1093/jamia/ocy142.

[19] Ziqi Zhang et al. "SynTEG: A Framework for Temporal Structured Electronic Health Data Simulation". In: *Journal of the American Medical Informatics Association* 28.3 (Mar. 2021), pp. 596–604. ISSN: 1527-974X. DOI: 10.1093/jamia/ocaa262.

[20] Chang Lu et al. "Multi-Label Clinical Time-Series Generation via Conditional GAN". In: *IEEE Transactions on Knowledge and Data Engineering* (2023), pp. 1–13. ISSN: 1558-2191. DOI: 10.1109/TKDE.2023.3310909.

[21] Giannis Nikolentzos et al. "Synthetic Electronic Health Records Generated with Variational Graph Autoencoders". In: *npj Digital Medicine* 6.1 (Apr. 2023), pp. 1–12. ISSN: 2398-6352. DOI: 10.1038/s41746-023-00822-x.

[22] Huan He et al. *MedDiff: Generating Electronic Health Records Using Accelerated Denoising Diffusion Model*. Feb. 2023. DOI: 10.48550/arXiv.2302.04355. arXiv: 2302.04355 [cs].

[23] Taha Ceritli et al. *Synthesizing Mixed-type Electronic Health Records Using Diffusion Models*. Aug. 2023. DOI: 10.48550/arXiv.2302.14679. arXiv: 2302.14679 [cs].

[24] Hongyi Yuan, Songchi Zhou, and Sheng Yu. *EHRDiff: Exploring Realistic EHR Synthesis with Diffusion Models*. Mar. 2024. DOI: 10.48550/arXiv.2303.05656. arXiv: 2303.05656 [cs].

[25] Yuan Zhong et al. *Synthesizing Multimodal Electronic Health Records via Predictive Diffusion Models*. June 2024. DOI: 10.48550/arXiv.2406.13942. arXiv: 2406.13942 [cs].

[26] Chao Pang et al. *CEHR-GPT: Generating Electronic Health Records with Chronological Patient Timelines*. May 2024. DOI: 10.48550/arXiv.2402.04400. arXiv: 2402.04400 [cs].

[27] Cristóbal Esteban, Stephanie L. Hyland, and Gunnar Rätsch. "Real-Valued (Medical) Time Series Generation with Recurrent Conditional GANs". In: *arXiv: Machine Learning* (June 2017).

[28] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. "Time-Series Generative Adversarial Networks". In: (Sept. 2019).

[29] Zinan Lin et al. "Using GANs for Sharing Networked Time Series Data: Challenges, Initial Promise, and Open Questions". In: *Proceedings of the ACM Internet Measurement Conference*. Virtual Event USA: ACM, Oct. 2020, pp. 464–483. ISBN: 978-1-4503-8138-3. DOI: 10.1145/3419394.3423643.

[30] Jin Li et al. "Generating Synthetic Mixed-Type Longitudinal Electronic Health Records for Artificial Intelligent Applications". In: *npj Digital Medicine* 6.1 (May 2023), pp. 1–18. ISSN: 2398-6352. DOI: 10.1038/s41746-023-00834-7.

[31] Jinsung Yoon et al. "EHR-Safe: Generating High-Fidelity and Privacy-Preserving Synthetic Electronic Health Records". In: *npj Digital Medicine* 6.1 (Aug. 2023), pp. 1–11. ISSN: 2398-6352. DOI: 10.1038/s41746-023-00888-7.

[32] Xinyu Yuan and Yan Qiao. "Diffusion-TS: Interpretable Diffusion for General Time Series Generation". In: *The Twelfth International Conference on Learning Representations*. Oct. 2023.

[33] Jian Qian et al. *TimeLDM: Latent Diffusion Model for Unconditional Time Series Generation*. July 2024. arXiv: 2407.04211 [cs].

[34] Namjoon Suh et al. *TimeAutoDiff: Combining Autoencoder and Diffusion Model for Time Series Tabular Data Synthesizing*. https://arxiv.org/abs/2406.16028v2.

[35] Yangming Li. *TS-Diffusion: Generating Highly Complex Time Series with Diffusion Models*. Nov. 2023. DOI: 10.48550/arXiv.2311.03303. arXiv: 2311.03303 [cs].

[36] Tuan Dinh et al. *LIFT: Language-Interfaced Fine-Tuning for Non-Language Machine Learning Tasks*. Oct. 2022. DOI: 10.48550/arXiv.2206.06565. arXiv: 2206.06565 [cs].

[37] Vadim Borisov et al. *Language Models Are Realistic Tabular Data Generators*. Apr. 2023. DOI: 10.48550/arXiv.2210.06280. arXiv: 2210.06280 [cs].

[38] Yoonhyung Lee, Younhyung Chae, and Kyomin Jung. "Leveraging VQ-VAE Tokenization for Autoregressive Modeling of Medical Time Series". In: *Artificial Intelligence in Medicine* 154 (Aug. 2024), p. 102925. ISSN: 0933-3657. DOI: 10.1016/j.artmed.2024.102925.

[39] Abdul Fatir Ansari et al. *Chronos: Learning the Language of Time Series*. Mar. 2024. arXiv: 2403.07815 [cs].

[40] Alexandru Grigoraș and Florin Leon. "Synthetic Time Series Generation for Decision Intelligence Using Large Language Models". In: *Mathematics* 12.16 (Aug. 2024), p. 2494. ISSN: 2227-7390. DOI: 10.3390/math12162494.

[41] Alec Radford et al. "Language Models Are Unsupervised Multitask Learners". In: *OpenAI blog* 1.8 (2019), p. 9.

[42] Hrayr Harutyunyan et al. "Multitask Learning and Benchmarking with Clinical Time Series Data". In: *Scientific Data* 6.1 (June 2019), p. 96. ISSN: 2052-4463. DOI: 10.1038/s41597-019-0103-9. arXiv: 1703.07771 [cs, stat].

[43] Muhammad Ferjad Naeem et al. *Reliable Fidelity and Diversity Metrics for Generative Models*. June 2020. DOI: 10.48550/arXiv.2002.09797. arXiv: 2002.09797 [cs, stat].

[44] Guolin Ke et al. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017.

[45] Gerald Woo et al. *Unified Training of Universal Time Series Forecasting Transformers*. Feb. 2024. DOI: 10.48550/arXiv.2402.02592. arXiv: 2402.02592 [cs].

[46] Yong Liu et al. *Timer: Generative Pre-trained Transformers Are Large Time Series Models*. June 2024. DOI: 10.48550/arXiv.2402.02368. arXiv: 2402.02368 [cs, stat].

[47] Karan Singhal et al. *Large Language Models Encode Clinical Knowledge*. Dec. 2022. DOI: 10.48550/arXiv.2212.13138. arXiv: 2212.13138 [cs].