PromptEHR: Prompt-based Language Models for Multi-modal Electronic Healthcare Records Generation

Anonymous Author(s)

ABSTRACT

Accessing longitudinal multi-modal Electronic Healthcare Records (EHRs) has long been challenging due to privacy concerns, which hinders the use of ML for healthcare applications. Synthetic EHRs generation is a promising direction bypassing the need to share sensitive real patient records. However, existing methods are limited to single-modal EHRs by making unconditional generation or longitudinal inference, which fall short in low flexibility and making unrealistic EHRs. In this work, we propose to formulate EHRs generation as a text-to-text translation task by language models (LMs) based on prompt-based learning, namely PromptEHR. This paradigm not only leverages the power of LMs but also allows ultimate flexibility in conditional imputation for multi-modal EHRs, which yield more realistic synthetic EHRs. Besides, we propose to evaluate EHRs quality by two perplexity measures accounting for their longitudinal pattern (longitudinal imputation perplexity, lpl) and the connections cross modalities (cross-modality imputation perplexity, mpl). Moreover, we utilize two adversaries: membership and attribute inference attacks for privacy-preserving evaluation. Experiments on MIMIC-III data demonstrate the superiority of our methods on realistic EHRs generation (53.1% decrease of lpl and 45.3% decrease of mpl on average compared to the best baselines) with low privacy concerns.

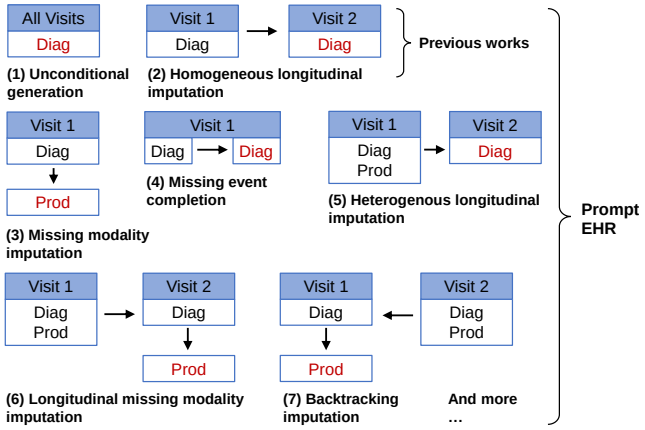


Figure 1: A demonstration of event imputation ways where black codes indicate known events and red codes are the targets to infer. Diag and Prod are short for diagnosis and procedure codes, respectively. Previous works generate EHRs by unconditional generation w/o time order, or by longitudinal predictions for homogeneous events; By contrast, crediting to its text-to-text translation formulation by prompt, PromptEHR allows ultimate flexibility, e.g., it supports additional (3) to (7) imputation ways and more.

1 INTRODUCTION

Electronic healthcare records (EHRs) fuel the development of machine learning models for healthcare applications [6, 7, 36, 37]. However, medical institutions are often reluctant to share EHRs to the research community due to the concern of privacy and legal risk. Therefore, the share of EHRs usually undergoes strict and expensive de-identification and administration processes. Although there have been attempts on perturbing potentially identifiable attributes as the de-identification step [12], they were argued not immune to the hack for re-identification [8, 11]. Alternatively, generating synthetic but realistic EHRs can circumvent data leakage while preserving the patterns of real EHRs for further research and development [4].

Deep generative models like GANs [14] and VAEs [23] have become popular for unconditional EHRs generation [8] and longitudinal EHRs generation [4, 40] for a single type of events like diagnosis codes, as illustrated by (1) and (2) in Fig. 1. However, EHRs are often multi-modal with different types of events, including diagnoses, procedures, medications, lab tests, and more [20]. Due to the limited representation capacity and flexibility, it is challenging to leverage GANs and VAEs for complex multi-modal data generation. A promising alternative is the transformer-based language model (LM) which has been proved powerful for learning from multi-modal data [27, 31]. However, unlike texts and images, EHRs contain structured and multi-modal sequences in time order, which render the direct applications of LMs infeasible.

In this work, we propose to leverage prompt-based learning to adapt the EHRs generation task to text generation task based on Bidirectional and Auto-Regressive Transformers (BART) [25], i.e., **Prompt**-based learning for **EHR**s generation (PromptEHR). Our method allows ultimate flexibility in data generation attributing to the prompt-based learning, as shown by Fig. 1. Besides (1) & (2), PromptEHR realizes the missing modality imputation (3) that infers the occurred procedures given the diagnoses in this visit; and the missing event completion (4). It can also infer the diagnoses in the next visit conditioned on the previous heterogeneous events (5) or the diagnoses in this visit (6). Reverse imputation is also feasible where the missing procedures in the first visit are inferred based on all other events (7). Moreover, PromptEHR is amenable to more imputation tasks if we develop new prompts for the generation. We argue this flexibility allows us to fully utilize the real EHRs for synthetic EHRs generation: we can apply arbitrary corruption to

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

the raw EHRs, then execute appropriate imputation methods to generate diverse and realistic synthetic EHRs.

We summarize our main contributions as:

- We propose EHR-BART that enables BART [25] for both longitudinal and missing modality conditional generation of heterogeneous EHRs by prompt-based learning.
- We design a systematic evaluation framework for quality and privacy of the generated synthetic records by LMs.
- We conduct extensive experiments that demonstrate the usefulness and safety of the synthetic EHRs for DL-based predictive healthcare applications.

In the following, we review the related works in §2, then present the main framework of PromptEHR in §3. Experimental details and results are shown in §4.

2 RELATED WORKS

2.1 EHRs Generation

Early works on generating EHRs [5, 28, 29] tried rule-based methods. However, they were argued that not capable of providing sufficiently realistic data for machine learning tasks and were still vulnerable to re-identification [8]. On the other hand, deep generative models advanced by the power of deep learning, e.g., variational auto-encoders (VAE) [23] and generative adversarial network (GAN) [14], gained attention from researchers recently. Choi et al. [8] pioneered in adapting GAN for discrete patient records generation, namely MedGAN, which was followed by a series of works on improving GANs for EHRs generation [3, 15, 40]. Besides, there were also methods based on VAE [4] and hybrid GANs [9, 24]. However, most of them only work on generating homogeneous EHRs and fall short in only being capable of longitudinal conditional generation.

2.2 LMs & Prompt-based Learning

LMs thrived in the natural language processing (NLP) field with the emergence of BERT [10], GPT-2 [32], and so on. They encouraged a shift from the *fully supervised learning* to the *pre-train and fine-tune* paradigm in NLP practice. Left-to-right (L2R) LMs, as one of the major types of LMs, were commonly adopted for text generation tasks attributed to their *auto-regressive* nature, e.g., T5 [33] and BART [25]. Nonetheless, they cannot be directly applied to EHRs generation since EHRs consist of not only plain clinical notes but also other forms of longitudinal sequences of events like lab tests, diagnosis codes, visit sequences, etc. Moreover, LMs learned from general corpus like Wikipedia are unable to provide sufficient representation of clinical data [19], it is imperative to include extra domain-specific corpus. As far as we know, although there were works on generating medical texts by LMs [1, 21, 26], none has been done for synthetic EHRs generation.

3 METHODS

In this section, we elaborate on the main framework of PromptEHR, including the problem setting, workflow, and training tasks formulation. Next, we discuss the strategies for generating diverse synthetic EHRs with minor loss of quality. Then, we present the recipe proposed for the evaluation for both quality and privacy-preserving ability of the EHR generation models.

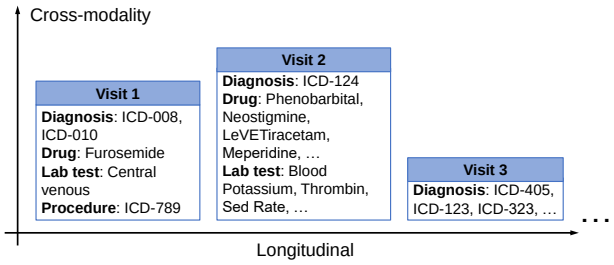


Figure 2: A mini example of a patient's record. Each patient may have more than one visit in time order. And real EHRs are multi-modal, i.e., in each visit, there are multiple types of events.

3.1 Problem Formulation

An example of patient record is demonstrated by Fig. 2 which are longitudinal and multi-modal. To formalize the problem of synthesizing patient records, we assume there are N patients in total, where the n-th patient records are represented by $X_{n,1:T_n} = \{\boldsymbol{x}_{n,1}, \boldsymbol{x}_{n,2}, \ldots, \boldsymbol{x}_{n,T_n}\}$. Here, T_n indicates the number of admissions, hence $\boldsymbol{x}_{n,t}$ signifies the records of his/her t-th visit. Consider there are K modalities, we have $\boldsymbol{x}_{n,t} = \{\boldsymbol{x}^1_{n,t}, \boldsymbol{x}^2_{n,t}, \ldots, \boldsymbol{x}^K_{n,t}\}$ where $\boldsymbol{x}^k_{n,t}$ indicates the codes from k-th modality which appear in this visit. Hereinafter, $\boldsymbol{x}^k_{n,t} = \{v_1, v_2, \ldots, v_l\}$ as $v \in \mathcal{V}_k$ is the code in the k-th modality.

Without loss of generality, here we formalize the **longitudinal imputation** ((2) & (5) shown in Fig. 1) and the **missing modality imputation** ((3) & (6) in Fig. 1). Other imputation tasks can be formulated similar to the above four. For the longitudinal prediction, we are given a list of historical records of the patient, as $X_{n,1:t} = \{\boldsymbol{x}_{n,1}, \ldots, \boldsymbol{x}_{n,t}\}$, the model should predict what will happen in the next admission, as $\boldsymbol{x}_{n,t+1}$; for the cross-modality prediction, the model is ought to fill the missing modality k, i.e., $\boldsymbol{x}^k_{n,t}$, conditioned on all the remaining modalities $\boldsymbol{x}_{n,t} \setminus \{\boldsymbol{x}^k_{n,t}\}$. Specifically, there can be more than one missing modalities.

We can leverage these two functions to synthesize EHRs conditioned on patient records or from scratch. For instance, given $\boldsymbol{x}^1_{n,1}$, we can let the model first do missing modality imputation to fill all modalities in this admission to get $\boldsymbol{x}_{n,1}$. Then, the model makes longitudinal prediction to get the following $\boldsymbol{x}_{n,2}$ and so on. Or we can randomly remove several modalities in each visit and iteratively do missing modality imputation and longitudinal imputation to generate diverse synthetic and realistic EHRs.

3.2 Model Architecture

Language models based on transformers take a sequence of tokens as their inputs. To build the inputs based on multi-modal codes, we make use of *prompts*. In detail, special tokens are introduced to specify the input modality and the predicted modality. Without the loss of generality, we assume there are two modalities in the data: diagnosis and drug. We use two special tokens <diag> and <drug> to cover them. Denote [X] and [Z] as the input and answer slots, respectively, we can formulate the missing modality prediction for the diagnosis codes as a *cloze prompt* problem: [X] <diag> [Z] [X].

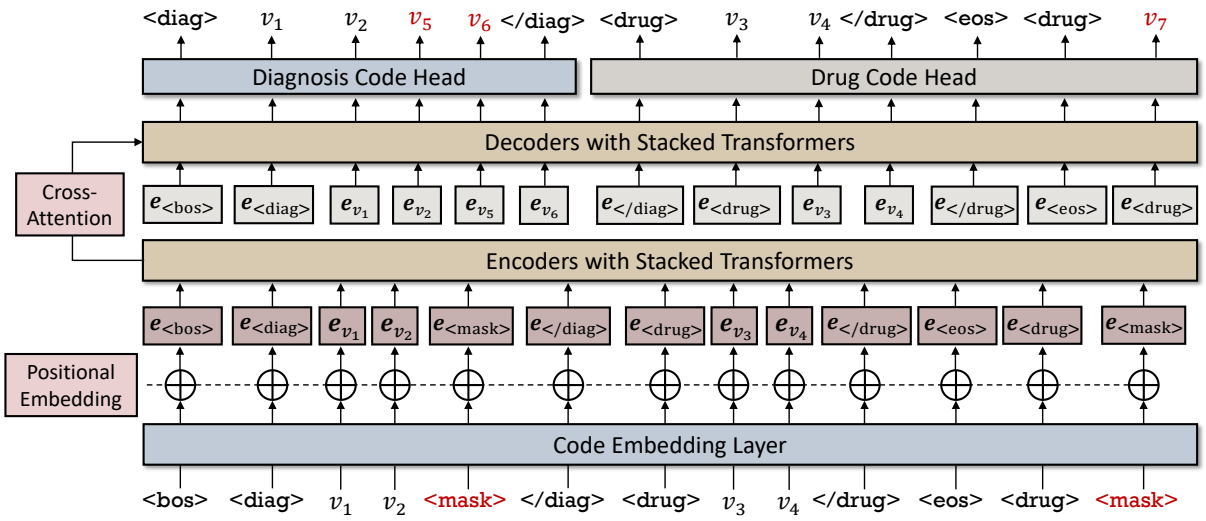


Figure 3: The workflow of the proposed PromptEHR method. The masked target codes are in red. The input codes covered by prompts that indicate the modality are mapped to dense vectors by the embedding layer. Their embeddings are added by the positional embedding indicating the time steps. After processing by the encoder stacked with transformers, the representations then become the inputs for the decoder to reconstruct the input sequence. In detail, different heads are responsible for different modalities.

The input slot [X] can contain the codes in historical visits and the drug codes in this admission. On the other hand, we use <eos> to divide codes from different admissions. This makes the longitudinal prediction a *prefix prompt* problem: [X] <eos> [Z]. The answer slot [Z] here can be further started by <diag> and <drug> for making generation separately. All the answer slots [Z] are covered by a special mask token <mask> during training.

Similarly, to do missing event completion, we can remove a part of diagnosis codes to build the answer for the prompt, as <diag> [X] [Z] </diag>; to do backtracking imputation, we can put the answer slot before <eos> as [Z] <eos> [X]. We can see that prompt-based learning with BART offers ultimate flexibility to build imputation tasks for synthesizing EHRs.

Fig. 3 plots the overview flowchart of PromptEHR. The inputs represent a patient's admission where there are two kinds of codes: diagnosis (<diag>) and drug (<drug>). The tasks are to fill the <mask> inside this admission and in the next admission. Raw inputs are encoded by the general code embedding layer then added with positional embeddings. The obtained input embeddings then go into the encoders, which are used for building the cross-attention towards the decoders later. The decoders try to recover the original inputs by a left-to-right paradigm. Specifically, two heads are responsible for generating two codes, respectively. When met with the prompts of modality, e.g., <drug>, the decoders will switch to the specific head for codes generation.

3.3 Training

PromptEHR is trained to recover the visit sequences given the corrupted inputs, supervised by the cross-entropy between the decoded sequence of codes and the groundtruth visits. Since we leverage a bidirectional encoder, we can apply any corruption techniques to the inputs. In detail, we design the supervision based on the combination of the following corruptions:

Code Mask, Infill, and Deletion. We follow how BART [25] did for token-level transformations. In our case, we randomly sample codes and replace them with <mask> or deleted. For infilling, a span of codes with length sampled as length ~ Poisson(3) are replaced with a single <mask>.

Span Shuffle and Permutation. Unlike natural language, the codes of EHRs inside a span and different modalities inside a visit are concurrent thus not ordered. We hereby shuffle the codes within the same span to rid the model's dependency on their orders. Similarly, we shuffle the span's order within the same admission to remove the modalities' orders in the inputs.

Longitudinal and Missing Modality Imputation. In each training iteration, we randomly mask one of the modalities and let the model to recover it based on all the remaining modalities. The longitudinal imputation requires the model to recover a modality in the next admission by all the patient's historical visits.

Denote the context by X and the target event by $\boldsymbol{x}$. We can denote the true distribution over the context X in the EHRs by $p(\boldsymbol{x}|X)$. For instance, for the longitudinal inference task, the context is the historical record of the patient $X_{1:t}$ and the target is the events in the next visit $\boldsymbol{x}_{t+1}$, as described in §3.1. Correspondingly, $p(\boldsymbol{x}|X;\theta)$ is the prediction made by the model. We use $\tilde{X} \sim q(X)$ to represent the stochastic perturbations added to the context, as mentioned above. The training objective is hence to minimize the negative log-likelihood as

$$\mathcal{L} = \mathbb{E}_{X\sim p(X)}\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|X)}\mathbb{E}_{\tilde{X}\sim q(X)}[-\log p(\boldsymbol{x}|\tilde{X};\theta)]. \quad (1)$$

3.4 Introduce Harmless Randomness to EHRs Generation

Apart from preciseness, the *diversity* of the generated data is also of great importance. Previous GAN and VAE based methods try to introduce randomness by sampling from a noise vector which is combined with another state vector as the inputs. This strategy follows the practice in deep generative models for image and video generation. However, images are insensitive to perturbation, e.g., we can inject high adversarial noises to many pixels in an image without changing the determination of human eyes. By contrast, it is challenging to adjust the degree of noises injected into the EHRs representations to balance the randomness and the quality: inappropriate noises may significantly vary the model predictions thus changing the generated events dramatically, which often causes low-quality synthetic EHRs.

Fortunately, with the `PromptEHR` framework, we can introduce rather a harmless randomness during the generation process. One major advantage of `PromptEHR` is that we can leverage the randomly corrupted real EHRs for synthesizing by a series of imputations discussed before (Fig. 1). On the other hand, we can take stochastic sampling for single code generation. Recall that LMs do event prediction by maximizing the conditional distribution: $\arg\max_{\boldsymbol{x}} P(\boldsymbol{x}_t|\boldsymbol{x}_{1:t-1};\theta)$. For generating more records, we can instead make sampling as

$$\boldsymbol{x} \sim P(\boldsymbol{x}_t|\boldsymbol{x}_{1:t-1};\theta). \tag{2}$$

Therefore, the sampled events still have high probability to be correct. Moreover, to prevent the generation of low probability events, we can apply *top-k* sampling to only sample from the k mostly likely next event [13]. Besides, temperature can be used to making the softmax distribution $P(\boldsymbol{x}_t|\boldsymbol{x}_{1:t-1};\theta)$ flatter or sharper to adjust the degree of randomness. More advanced strategies from the text generation literature, e.g., beam search [38] and nucleus sampling [18], are all available for exploitation in `PromptEHR`, which brings great potential for `PromptEHR` to achieve a better trade-off between generation quality and diversity.

3.5 Quality Evaluation

We here provide a recipe to evaluate LMs on two dimensions: **accuracy** and **privacy**. For accuracy, we propose to adopt perplexity which is usually used in the text generation task, defined by the exponent of the average negative log-likelihood (NLL) per word [30]:

$$\texttt{ppl} = e^{-(\log P(v_1,\ldots,v_L;\theta))/L} = e^{-(\log \prod_{l=1}^{L} P(v_l|v_{1:l-1};\theta))/L}, \tag{3}$$

where $P(v_l|v_{1:l-1})$ indicates how the model predicts the next word using all previous words as the context; L is the length of the document; θ is the model parameter. Intuitively, a random predictor will produce `ppl` that is equal to the cardinality of vocabulary $|\mathcal{V}|$. However, the EHR records have a different structure from the natural language. Codes are multi-modal and those within the same admission are not ordered. We hereby adapt it to the **longitudinal imputation perplexity** and **cross-modality imputation perplexity** taking the structure of EHR into account.

Longitudinal Imputation Perplexity. For accurate generation, the model should capture the temporal coherence of the patient conditions. For instance, some chronic diseases like diabetes can cause complications (e.g., heart disease and kidney failure) in the future. Following Eq. (3), we can write the longitudinal imputation perplexity `lpl` of a patient's records $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ as

$$\begin{aligned}\texttt{lpl} &= e^{-\sum_{t=1}^{T} \log P(\boldsymbol{x}_t|\boldsymbol{x}_{1:t-1};\theta)/(l_t * T)} \\ &= e^{-\sum_{t=1}^{T}\sum_{l=1}^{l_t} \log P(v_l|\boldsymbol{x}_{1:t-1};\theta)/(l_t * T)}.\end{aligned} \tag{4}$$

Here, $\boldsymbol{x}_t = \{v_1, \ldots, v_{l_t}\}$ are all codes during the t-th admission. Inside this admission, all these codes are conditionally independent, therefore we can decompose $P(\boldsymbol{x}_t|\boldsymbol{x}_{1:t-1};\theta) = \prod_{l=1}^{l_t} P(v_l|\boldsymbol{x}_{1:t-1};\theta)$ then come to the results.

Cross-modality Imputation Perplexity. For example, if the patient is diagnosed with fever while the lab tests indicating her high body temperature, Acetaminophen is a probable medication given in this admission. In this case, we focus on the t-th admission where the joint distribution of all K modalities $P(\boldsymbol{x}_t^1, \ldots, \boldsymbol{x}_t^K|\boldsymbol{x}_{1:t-1};\theta)$. We hope to test the model's ability on cross-modality imputation, i.e., $P(\boldsymbol{x}_t^k|\boldsymbol{x}_t^{1:K\setminus k}, \boldsymbol{x}_{1:t-1};\theta)$. We can write the NLL here by

$$\begin{aligned}\text{NLL}_t &= -\frac{1}{K}\sum_{k=1}^{K} \log P(\boldsymbol{x}_t^k|\boldsymbol{x}_t^{1:K\setminus k}, \boldsymbol{x}_{1:t-1};\theta) \\ &= -\frac{1}{K}\sum_{k=1}^{K}\frac{1}{l_t^k}\sum_{l=1}^{l_t^k} \log P(v_l|\boldsymbol{x}_t^{1:K\setminus k}, \boldsymbol{x}_{1:t-1};\theta),\end{aligned} \tag{5}$$

where l_t^k indicates the number codes belonging the k-th modality. Next, we can track all admissions to obtain the final definition of `mpl` by

$$\texttt{mpl} = e^{\sum_{t=1}^{T} \text{NLL}_t/T}. \tag{6}$$

3.6 Privacy Evaluation

It is crucial for us to measure the privacy preserving when sharing the synthetic data generated by the model trained on the true data. We try to evaluate two privacy risks: **membership inference** and **attribute inference**. We split the data into the training data $\mathcal{D}_1 = \{X_{n,1:T_n}\}_{n=1}^{N}$ and testing data $\mathcal{D}_2$, and generate synthetic data $\mathcal{D}_S$ with the same length as $\mathcal{D}_1$.

Membership Inference. Attackers would try to infer the membership of the patient records based on the real records they own. Once this membership leaks, attackers can leverage it to infer more sensitive information from the training database. We design this adversary based on shadow training [34]. In the first stage, a shadow model M_{sd} is trained on $\mathcal{D}_S$. It tries to mimic the performance of the generation model in longitudinal inference.

In the second stage, a membership inference dataset is built based on $M_{\text{sd}}(X)$ where $X \in \widetilde{\mathcal{D}}_S \bigcup \mathcal{D}_2$. $\widetilde{\mathcal{D}}_S$ is a subset of $\mathcal{D}_S$ with the same number as $\mathcal{D}_2$. A model $M_{\text{mi}} : \mathbb{Y}_{\texttt{ppl}} \mapsto \{0, 1\}$ is trained to differentiate if X comes from $\mathcal{D}_S$ or $\mathcal{D}_2$. We will then evaluate the success rate of M_{mi} on identifying $X \in \mathcal{D}_1 \bigcup \mathcal{D}_2$. The better the adversary $M_{\text{sd}}(X)$ and M_{mi} perform on this evaluation, the higher the privacy risk caused by releasing the synthetic EHRs.

Attribute Inference. We build this adversary following [39]. In this case, attackers hold some incomplete real records where several sensitive attributes are missing. They would take advantage of the synthetic data to infer these attributes. Besides, attackers

Table 1: Statistics of the used MIMIC-III data.

Item	Number	Code Type	Number
Patients	46,520	Diagnosis	1,071
Total Visits	58,976	Drug	500
Total Codes	5,401,961	Procedure	668
Codes per Patient	116	Lab Test	185

also hold the prior knowledge of association between the attributes, i.e., given the incomplete individual records, how probable another code appears in expectation or $P_0(v_l|\{v_1,\ldots,v_{l_t}\}_{t=1}^T \setminus v_l)$. With the prior, the attacker will train an attribute imputation model on the synthetic data $\mathcal{D}_S$, i.e., $P(v_l|\{v_1,\ldots,v_{l_t}\}_{t=1}^T \setminus v_l;\theta_I)$. The attacker then believe the code v_l exists when

$$\log P(v_l|\{v_1,\ldots,v_{l_t}\}_{t=1}^T \setminus v_l;\theta_I) - \log P_0(v_l|\{v_1,\ldots,v_{l_t}\}_{t=1}^T \setminus v_l) \geq \delta. \tag{7}$$

δ is a pre-defined threshold. In experiments, we train an another attribute imputation model on $\mathcal{D}_1$ to approximate the prior knowledge. We evaluate the success rate of this attack. Besides, a imputation model trained on the testing set is leveraged for calibration.

4 EXPERIMENTS

In this section, we designed experiments to answer the following questions.

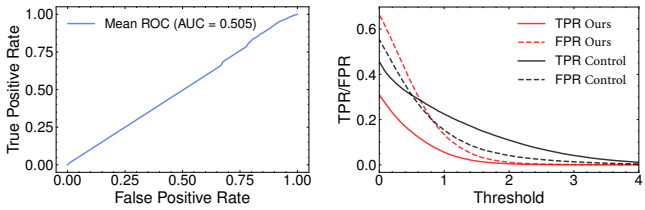
- **Q1.** How well does PromptEHR perform for EHRs generation compared with the state-of-the-art methods on generation quality?
- **Q2.** What is the level of privacy risk on membership inference and attribute inference of the generated EHRs by PromptEHR?
- **Q3.** Are the synthetic data useful for further predictive modeling in practice?
- **Q4.** How is the generation quality of PromptEHR influenced by the size of training records?

4.1 Experimental Setup

4.1.1 MIMIC-III dataset. [20] We use MIMIC-III data for training and evaluation in this paper, which has 46k patients' records collected from the intensive care unit. We pick the diagnosis, procedure, drug, and lab test as the target events for generation. All events in the same admission are seen as contemporary. We randomly split the 46,520 patients records into 39,581, 2,301, 4,633 for the train/validation/test set. The data statistics are available in Table 1.

4.1.2 Baselines. We compare the performance of PromptEHR with several baselines:

- LSTM+MLP [17]. This is the baseline that leverages LSTM to learn the patient state thus extracting the temporal visit patterns. Based on the state embeddings, MLP layers are able to impute the probability of events within the visit or for the next visit.
- LSTM+MedGAN [8]. The original MedGAN is not able to do conditional generation and temporal inference. Similar to the first baseline, LSTM is used for capturing temporal patterns as the inputs for MedGAN. Then, the generator of MedGAN will try to make conditional generation for records as realistic as possible to fool its discriminator.



(a) The ROC curve of the membership inference attack by shadow training.

(b) The true positive rate (TPR) and false positive rate (FPR) of the attribute inference attack w.r.t. different thresholds δ.

Figure 4: Privacy-preserving evaluation on membership inference (left) and attribute inference (right) adversaries. On the right, the PromptEHR curves indicate the results of attribute inference model trained on the synthetic data $\mathcal{D}_S$ by PromptEHR; the Control curves indicate the one trained on test set $\mathcal{D}_2$.

- SynTEG [39]. This is one of the most recent EHRs generation methods. It also consists of a state embedding module and a imputation module. It utilizes transformers [35] for temporal dependency learning and conditional Wasserstein GAN with gradient penalty (WGAN-GP) [2, 16] for event inference.
- GPT-2 [32]. We pick GPT-2 as the LM baseline that only does causal language modeling on EHRs. Then, it is able to do event inference like texts generation.

4.1.3 Evaluation metrics. For evaluating generation quality, we make use of the proposed two perplexity measures: longitudinal and cross-modality imputation perplexity. Since perplexity of different patient records vary significantly, we take the median of perplexity across patients for the sake of stability of the performance estimate.

For evaluating the privacy of PromptEHR, we use two adversaries: membership inference (MI) and attribute inference (AI). In MI, we use LSTM+MLP as the shadow model to mimic the outputs of PromptEHR. A three-layer MLP is then for predicting the membership. ROC curve is plotted to evaluate the attack success rate; In AI, we train an LSTM+MLP on $\mathcal{D}_1$ to approximate the prior and another LSTM+MLP on $\mathcal{D}_S$ as the attribute imputation model. It is the same for the control set.

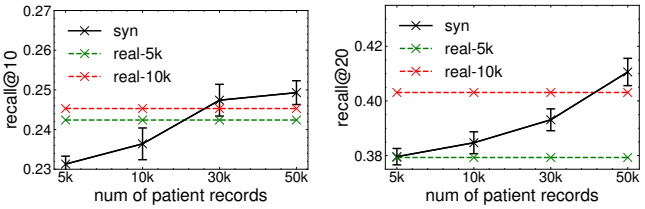
To test the utility of the synthetic data for downstream predictive healthcare applications, we train LSTM+MLP on $\mathcal{D}_S/\mathcal{D}_2$ and test it on $\mathcal{D}_2$ to compute the recall@20/30.

4.1.4 Implementation details. For all the used LSTM+MLP models, we take a three-layer bi-directional LSTM with 128 hidden dimensions with one 256-dim MLP layer. It is trained with 1e-4 learning rate by Adam optimizer [22]. The 12-layer transformer based pretrained GPT-2 is trained with 1e-5 learning rate and 1e-4 weight decay by Adam. We follow the architecture and training protocol from the original papers of MedGAN and SynTEG.

In PromptEHR, we utilize the pretrained BART-base model [25] to build our EHR-BART. We use Adam by setting learning rate as 1e-5, weight decay as 1e-4, batch size as 16. The total training epoch is 50 where the first 3 epochs are warm-up steps. During the training

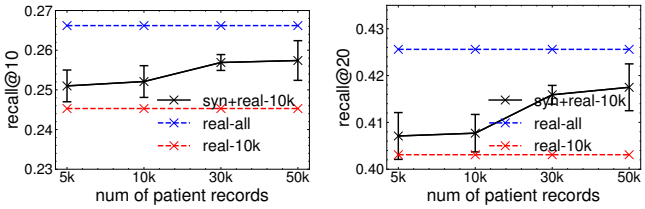
Table 2: Longitudinal imputation perplexity (lpl) & cross-modality imputation perplexity (mpl) of models on different kinds of codes. Best values are in bold. ± value indicates the 95% confidence interval.

Method/Code perplexity	Diagnosis		Procedure		Drug		Lab Test	
	lpl	mpl	lpl	mpl	lpl	mpl	lpl	mpl
LSTM+MLP	125.1 ± 5.3	122.9 ± 2.0	40.3 ± 1.7	43.8 ± 0.9	173.3 ± 1.9	169.5 ± 0.5	68.9 ± 0.3	71.3 ± 0.5
LSTM+MedGAN	169.2 ± 6.0	109.8 ± 3.1	54.4 ± 2.5	40.1 ± 1.4	197.3 ± 2.5	166.7 ± 0.9	76.9 ± 0.3	66.2 ± 0.2
SynTEG	130.4 ± 4.6	130.0 ± 2.6	46.4 ± 1.8	46.2 ± 1.5	175.6 ± 2.0	175.4 ± 0.9	69.5 ± 0.2	69.6 ± 0.3
GPT-2	121.1 ± 1.8	134.2 ± 0.9	38.7 ± 0.9	48.2 ± 0.5	166.4 ± 1.8	169.6 ± 0.6	69.7 ± 0.1	69.6 ± 0.1
PromptEHR	**65.9 ± 2.0**	**67.7 ± 0.6**	**13.5 ± 0.8**	**10.1 ± 0.3**	**104.7 ± 1.8**	**93.7 ± 0.5**	**24.4 ± 0.1**	**50.1 ± 0.1**



(a) Recall@10. **(b) Recall@20.**

Figure 5: Recall@10/20 of the predictive model on the test set with varying input data size: *syn* indicates the model trained on *fully synthetic data*; *real-5k/10k* indicate trained on 5k/10k real data. Error bars show the 95% confidence interval.



(a) Recall@10. **(b) Recall@20.**

Figure 6: Recall@10/20 of the predictive model on the test set with varying input data size: *syn+real-10k* indicates the model trained on the *hybrid of synthetic & 10k real data*; *real-10k/all* indicate trained on 10k/all real data. Error bars show the 95% confidence interval.

stage, the perplexity computed on the validation set is used to pick the best checkpoint for the testing phase.

All experiments are conducted with an RTX-3090 GPU, 251 GB RAM, and AMD Ryzen Threadripper 3970X 32-core CPU.

4.2 Q1: Generation Quality Comparison

We compare the calculated mpl and lpl of all methods in Table 2, where it can be witnessed that PromptEHR obtains the best result among all methods. On the contrary, LSTM+MedGAN and SynTEG do not gain better test perplexity than the basic LSTM+MLP. The main reason is that their GAN part takes a noise input except for the learned temporal state embeddings to make conditional generation. Although this technique might enhance the diversity of the generated samples, it inevitably undermines the generation quality due to the varying noisy inputs. GPT-2 works better than LSTM+MLP on temporal perplexity crediting to its power in capturing series pattern through transformers.

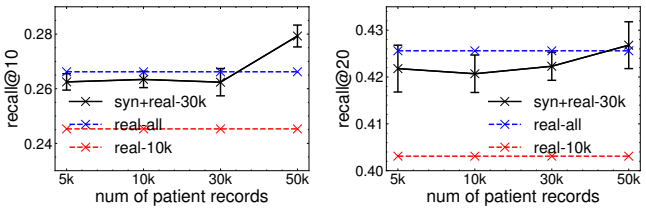
On the other hand, most methods obtain better mpl than lpl. It is intuitive because models know the additional in-visit information from the other modalities for the target modality imputation, thus making better predictions. However, GPT-2 performs worse in mpl than in lpl. The reason is that GPT-2 is trained with the so-called causal language modeling task where it models the sequence autoregressively. Though this manner works for language, it is sensitive to the order change of events within visits when modeling EHRs, which induces weak inference performance for contemporary events.

4.3 Q2: Privacy Preserving Evaluation

As aforementioned, we test the privacy preserving ability of the generated synthetic EHRs by applying membership and attribute inference attacks. Results are illustrated by Fig. 4.

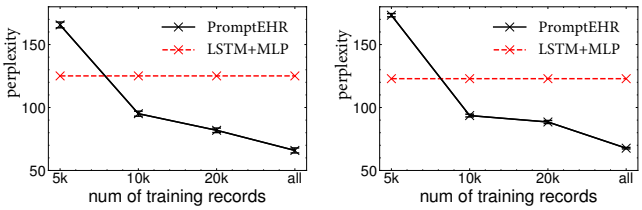
Fig. 4a demonstrates the ROC curve consisting of true positive rate (TPR) and false positive rate (FPR) of the membership inference on $\mathcal{D}_1 \bigcup \mathcal{D}_2$. It clearly shows the MI model has bad performance that is near random guess (AUC $\simeq$ 0.5), which means the MI attack gains no sensitive membership information when trained on the synthetic data $\mathcal{D}_S$.

Fig. 4b showcases the TPR/FPR of attribute inference attack based on shadow training with the varying threshold defined in Eq. (7). Here, we cut the curve where $\delta = 4$ because all the remaining curves are approaching zero on its right. The threshold δ adjusts to the confidence level of the attacker, i.e., the smaller δ is set, the higher probability that the AI is correct we believe. When $\delta = 0$, so long as the AI inference probability $P(v_l)$ is larger than the prior $P_0(v_l)$, the AI model will believe the attribute v_l exists. In this scenario, both two models have a high FPR of around 0.6, but the TPR of PromptEHR is only near half the control model. The TPR of PromptEHR then keeps a much lower level when δ increases, which implies the low attribute leakage risk of the synthetic data generated by PromptEHR. Although the FPR of PromptEHR becomes smaller than Control when $\delta > 0.8$, the TPR of PromptEHR is approaching zero after that. That means, being conservative for PromptEHR avoids inferring some wrong attributes but loses the ability to specify the right attributes at the same time. In a nutshell, the synthetic data by PromptEHR has a low risk to leak the attribute information.



(a) Recall@10. (b) Recall@20.

Figure 7: Recall@10/20 of the predictive model on the test set with varying input data size: *syn+real-30k* indicates the model trained on the *hybrid of synthetic & 30k real data*; *real-30k/all* indicate trained on 30k/all real data. Error bars show the 95% confidence interval.



(a) Longitudinal perplexity (lpl). (b) Cross-modality perplexity (mpl).

Figure 8: Black solid lines show the spatial and temporal perplexities of PromptEHR with regard to varying input training record sizes. Red dotted lines show the spl and tpl of baseline LSTM+MLP trained on all training records (~40k). Error bars show the 95% confidence interval.

4.4 Q3: Synthetic EHRs Utility

The ultimate goal of synthetic EHRs generation is to assist the downstream healthcare applications without sharing sensitive real patient records. Now, with the synthetic data at hand, we aim to measure the utility of $\mathcal{D}_S$ by PromptEHR for the sake of one common task in DL for healthcare: the clinical event prediction. We utilize an LSTM model to be trained on $\mathcal{D}_S$ and $\mathcal{D}_1$ then make multilabel predictions for diagnosis events similar to the setting in [7]. We evaluate the LSTM models by recall@10 and recall@20. In detail, we design two experiments: (1) train LSTM on fully synthetic data and compare its performance with the one trained on real data; (2) train LSTM on a mixture of synthetic data and part of real data where the synthetic data is regarded as a means of data augmentation.

4.4.1 Q3-(1): Fully synthetic data. This is the case when we share the synthetic data to those who have no access to any real EHRs while hoping to develop healthcare applications. We test the LSTM performance on 5k, 10k, 30k, and 50k synthetic patient records. For comparison, the model performance on 5k and 10k real records are also tested. Results are shown in Fig. 5. For recall@10 in Fig. 5a, we can observe that though 10k synthetic records are not comparable to 5k real records, 30k synthetic records can reach a better performance than 10k real records. On the other hand, for recall@20 in Fig. 5b, we can surprisingly see that 5k synthetic records achieve the same performance as the 5k real records. With more synthetic records involved, the 50k synthetic records-based LSTM outperforms its counterpart on 10k real records at last. This experiment demonstrates that synthetic EHRs by PromptEHR are sufficient to support healthcare applications. Users are expected to achieve comparable performance by synthetic records as the real data.

4.4.2 Q3-(2): Hybrid synthetic-real data. To further investigate that whether the synthetic records can be a beneficial complement to real records as a means of data augmentation, we try to train LSTM on the hybrid synthetic-real data. In Fig. 6, we randomly sample 10k real data from $\mathcal{D}_1$ and combine them with different sizes of synthetic data from $\mathcal{D}_S$. We find that the model trained on the augmented hybrid data has obvious advantages over its counterpart on the real data. With more synthetic records involved, the model gains better performance. This demonstrates the utility of synthetic data used as augmentation in low-resource cases. Besides, from Fig. 6 we identify this hybrid data is still inferior to the model trained on all real records. So we are curious about how many synthetic and real data we need to *outperform* this seemingly performance upper bound. In other words, can we beat the real data with the synthetic data?

We conduct the next experiment where 30k real data is combined with synthetic data. Note that we have around 40k real training records in total. Results are shown in Fig. 7. It can be seen that with 50k synthetic records plus 30k real records can outperform the model on all the real training records.

Table 3: A synthetic patient generated by PromptEHR. *ICD_abc* indicates the first three digits represented by ICD code of the event.

visit-1	diagnosis: Liveborn labtest: Hematocrit procedure: Prophylactic vaccination
visit-2	diagnosis: Streptococcus infection, Extreme immaturity, Perinatal infection, Neonatal jaundice, Liveborn labtest: Anion Gap, Bands, Base Excess, Bilirubin, Total, Chloride, Eosinophils, Hematocrit, Hemoglobin, Lymphocytes, MCH, MCHC, MCV, Monocytes, Platelet Count, Potassium, Red Blood Cells, Sodium, pCO2, pH, pO2 drug: Ampicillin Sodium, Heparin Sodium (Preservative Free), NEO*IV*Gentamicin, NEO*PO*Ferrous Sulfate Elixir, Send 500mg Vial, Syringe (Neonatal) *D5W* procedure: Biopsy of spinal cord

4.5 Q4: Generation Quality w.r.t. Training Data Size

In practice, the original data source to be shared might be in limited size, which elicits a question on how much the generation quality of PromptEHR is influenced by the size of the training cohort. To answer this question, we sampled 5k, 10k, and 20k patient records from the training set and testify the perplexity of the learned PromptEHR. Results are illustrated by Fig. 8. We plot the performance of the baseline LSTM+MLP method trained on all real training records

Table 4: A synthetic patient generated by PromptEHR based on a real patient record. The imputed events are marked red. For demonstration, we cut the events after the fifth for each visit due to the space limit.

visit-1	diagnosis: Pneumonia, Hematemesis, Heart failure, Emphysema labtest: Leukocytes, Urea Nitrogen, Calcium, Ketone procedure: Enteral infusion of nutrition, Insertion of airway, Replace tracheostomy tube, Temporary tracheostomy
visit-2	diagnosis: Heart failure, Respiratory conditions, Tracheostomy status, Stomach disorder labtest: Urine Appearance, Yeast, Platelet Count, Calculated Total CO2 procedure: Biopsy of bronchus, Replace gastrostomy tube, Invasive mechanical ventilation, Infusion of nesiritide
visit-3	diagnosis: Pneumonia, Mechanical complication, Pulmonary manifestations , Disorders of urinary tract labtest: INR(PT), Epithelial Cells, RBC, Urine Appearance procedure: Insertion of airway, Enterostomy, Lysis of peritoneal adhesions, Lung biopsy
visit-4	diagnosis: Mechanical complication, Hodgkin's paragranuloma, Pressure ulcer, Heart failure labtest: Urine Color, Urobilinogen, Bands, Urea Nitrogen procedure: Infusion of nesiritide, Endoscopy of small intestine, Gastrostomy, Replace tracheostomy tube
visit-5	diagnosis: Urethra disorder, Attention to tracheostomy/gastrostomy, Pneumonia, Heart failure labtest: MCH, Bacteria, Lymphocytes, Calculated Total CO2 drug: Fluticasone Propionate 110mcg, SW, Bisacodyl, Iso-Osmotic Dextrose procedure: Replace tracheostomy tube, Heart cardiac catheterization, Enteral infusion of nutrition
visit-6	diagnosis: Pneumonia, Heart failure, Endomyocardial fibrosis, Mechanical complication labtest: pH, Epithelial Cells, WBC, Protein drug: Neutra-Phos, Mirtazapine, Fluconazole, SW procedure: Invasive mechanical ventilation, Airway infusion, Monitoring of cardiac output, Lung biospy

(~40k) in red dotted lines for comparison. We can see that with 5k training records, PromptEHR has worse generation quality than the baseline. When additional 5k records are involved, PromptEHR not only outperforms the LSTM baseline but also all other baselines reported in Table 2, which demonstrates that PromptEHR is amenable to low resources and superior than the baselines.

5 CASE STUDY

We present two randomly picked synthetic patients generated by PromptEHR in Tables 3 and 4. Due to the space limit, we cut the events after the fifth in the second case (Table 4). Four types of events are included in the synthetic examples during generation: diagnosis, lab test, procedure, and drug. In general, we observe that PromptEHR is capable of generating diverse events where events within / across visits evolve in a logical manner.

The first case was generated from scratch (Table 3), it describes a patient who goes into ICU because of a cesarean. During the operation, a test of Hematocrit should be conducted to ensure blood loss of the patient within the safe range. In the second visit, the patient suffers from a bacteria infection. The patient then receives a series of lab tests regarding the inflammation. And spinal tap is performed to help cure serious infections. Antibiotic drugs, e.g., Ampicillin Sodium and Gentamicin, are used to cure the patient. It can be seen that the generated events all center around the same topic (liveborn) and the longitudinal and cross-modal connections are coherent.

The second case was generated based on a real patient EHR by leveraging flexible imputation functions of PromptEHR (Table 4). The model scans through the record in time order. For each modality in a visit, we randomly choose to keep all events, remove all events, or remove a part at random. The imputed events are marked red. For example, in visit-1, the model takes the diagnosis codes with prompts as inputs and generates the lab tests. Then, the generated lab tests are involved in the input with prompts. In addition, the procedure 'Enteral infusion of nutrition' is also kept in the inputs. The model then generates the remaining procedures in this visit. This process repeats until reaches visit-6 where the real EHR ends.

In general, the events in the second case are coherent under the topic of pneumonia and heart failure. The patient is diagnosed as suffering from pneumonia due to bacteria with many complications like a hemorrhage of gastrointestinal tract, heart failure, and pulmonary collapse. At the same time, procedures like the enteral infusion of nutrition, insertion/replacement of endotracheal tube, and temporary tracheostomy are all included to maintain the patient's life regarding his/her nutrition and breath. Besides this visit, the remaining synthetic visits are also reasonable: he/she gets diagnoses regarding heart failure, respiratory diseases, stomach disorders, etc., which all correspond to relevant issues appearing in the first visit. These two cases offer an intuitive demonstration of the effectiveness of PromptEHR in generating realistic EHRs, especially when we take the advantage of multiple imputation functions to generate rather realistic EHRs based on real EHRs, which was hardly mentioned in previous works.

6 CONCLUSION

In this paper, we study how to leverage real EHRs to train a prompt learning based generative language model for synthetic EHRs generation, namely PromptEHR. Unlike previous EHRs generation methods, PromptEHR is able to learn from and generate heterogeneous EHRs by both longitudinal and latitudinal inference. To evaluate its performance, we draw the idea of perplexity from the text generation literature and propose two perplexity measures: spatial and

temporal perplexity. Experiments on MIMIC-III data demonstrates the quality of generated EHRs are better than the baselines. And the synthetic data provides both utility and privacy for downstream healthcare applications.

REFERENCES

[1] Ali Amin-Nejad, Julia Ive, and Sumithra Velupillai. 2020. Exploring transformer text generation for medical dataset augmentation. In *Language Resources and Evaluation Conference*. 4699–4708.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*. PMLR, 214–223.

[3] Mrinal Kanti Baowaly, Chia-Ching Lin, Chao-Lin Liu, and Kuan-Ta Chen. 2019. Synthesizing electronic health records using improved generative adversarial networks. *Journal of the American Medical Informatics Association* 26, 3 (2019), 228–241.

[4] Siddharth Biswal, Soumya Ghosh, Jon Duke, Bradley Malin, Walter Stewart, and Jimeng Sun. 2020. EVA: Generating Longitudinal Electronic Health Records Using Conditional Variational Autoencoders. *arXiv preprint arXiv:2012.10020* (2020).

[5] Anna L Buczak, Steven Babin, and Linda Moniz. 2010. Data-driven approach for creating synthetic electronic medical records. *BMC Medical Informatics and Decision Making* 10, 1 (2010), 1–28.

[6] Edward Choi, Mohammad Taha Bahadori, Joshua A Kulas, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. RETAIN: An interpretable predictive model for healthcare using reverse time attention mechanism. In *International Conference on Neural Information Processing Systems*. 3512–3520.

[7] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Doctor AI: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*. PMLR, 301–318.

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

[9] Limeng Cui, Siddharth Biswal, Lucas M Glass, Greg Lever, Jimeng Sun, and Cao Xiao. 2020. CONAN: Complementary pattern augmentation for rare disease detection. In *AAAI Conference on Artificial Intelligence*, Vol. 34. 614–621.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics*.

[11] Khaled El Emam, Elizabeth Jonker, Luk Arbuckle, and Bradley Malin. 2011. A systematic review of re-identification attacks on health data. *PloS one* 6, 12 (2011), e28071.

[12] Khaled El Emam, Sam Rodgers, and Bradley Malin. 2015. Anonymising and sharing individual patient data. *BMJ: British Medical Journal* 350 (2015), h1139.

[13] Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical Neural Story Generation. In *Annual Meeting of the Association for Computational Linguistics*. 889–898.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in Neural Information Processing Systems* 27 (2014).

[15] Jiaqi Guan, Runzhe Li, Sheng Yu, and Xuegong Zhang. 2018. Generation of Synthetic Electronic Medical Record Text. In *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE Computer Society, 374–380.

[16] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. 2017. Improved training of Wasserstein GANs. In *International Conference on Neural Information Processing Systems*. 5769–5779.

[17] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[18] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The Curious Case of Neural Text Degeneration. In *International Conference on Learning Representations*.

[19] Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. 2020. ClinicalBERT: Modeling clinical notes and predicting hospital readmission. In *Conference on Health, Inference, and Learning*.

[20] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific Data* 3, 1 (2016), 1–9.

[21] Rina Kagawa, Yukino Baba, and Hideo Tsurushima. 2021. A practical and universal framework for generating publicly available medical notes of authentic quality via the power of crowds. In *IEEE International Conference on Big Data*. IEEE, 3534–3543.

[22] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[23] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[24] Dongha Lee, Hwanjo Yu, Xiaoqian Jiang, Deevakar Rogith, Meghana Gudala, Mubeen Tejani, Qiuchen Zhang, and Li Xiong. 2020. Generating sequential electronic health records using dual adversarial autoencoder. *Journal of the American Medical Informatics Association* 27, 9 (2020), 1411–1419.

[25] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Annual Meeting of the Association for Computational Linguistics*. 7871–7880.

[26] Claudia Alessandra Libbi, Jan Trienes, Dolf Trieschnigg, and Christin Seifert. 2021. Generating Synthetic Training Data for Supervised De-Identification of Electronic Health Records. *Future Internet* 13, 5 (2021), 136.

[27] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030* (2021).

[28] Joseph S Lombardo and Linda J Moniz. 2008. A Method for Generation and Distribution of Synthetic Medical Record Data for Evaluation of Disease-Monitoring Systems. *Johns Hopkins APL Technical Digest* 27, 4 (2008), 356.

[29] Scott McLachlan, Kudakwashe Dube, and Thomas Gallagher. 2016. Using the caremap with health incidents statistics for generating the realistic synthetic electronic healthcare record. In *IEEE International Conference on Healthcare Informatics*. IEEE, 439–448.

[30] Graham Neubig. 2017. Neural machine translation and sequence-to-sequence models: A tutorial. *arXiv preprint arXiv:1703.01619* (2017).

[31] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020* (2021).

[32] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. [n. d.]. Language models are unsupervised multitask learners. ([n. d.]).

[33] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21, 140 (2020), 1–67. http://jmlr.org/papers/v21/20-074.html

[34] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy*. IEEE, 3–18.

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[36] Zifeng Wang, Rui Wen, Xi Chen, Shilei Cao, Shao-Lun Huang, Buyue Qian, and Yefeng Zheng. 2021. Online Disease Diagnosis with Inductive Heterogeneous Graph Convolutional Networks. In *Proceedings of the Web Conference 2021*. 3349–3358.

[37] Zifeng Wang, Yifan Yang, Rui Wen, Xi Chen, Shao-Lun Huang, and Yefeng Zheng. 2021. Lifelong Learning Based Disease Diagnosis on Clinical Notes. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 213–224.

[38] Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural Text Generation With Unlikelihood Training. In *International Conference on Learning Representations*.

[39] Ziqi Zhang, Chao Yan, Thomas A Lasko, Jimeng Sun, and Bradley A Malin. 2021. SynTEG: A framework for temporal structured electronic health data simulation. *Journal of the American Medical Informatics Association* 28, 3 (2021), 596–604.

[40] Ziqi Zhang, Chao Yan, Diego A Mesa, Jimeng Sun, and Bradley A Malin. 2020. Ensuring electronic medical record simulation through better training, modeling, and evaluation. *Journal of the American Medical Informatics Association* 27, 1 (2020), 99–108.