PromptEHR: Conditional Electronic Healthcare Records Generation with Prompt Learning

Zifeng Wang\textsuperscript{1} and Jimeng Sun\textsuperscript{1,2}
\textsuperscript{1}Department of Computer Science, University of Illinois Urbana-Champaign
\textsuperscript{2}Carle Illinois College of Medicine, University of Illinois Urbana-Champaign
\{zifengw2,jimeng\}@illinois.edu

Abstract

Accessing longitudinal multimodal Electronic Healthcare Records (EHRs) is challenging due to privacy concerns, which hinders the use of ML for healthcare applications. Synthetic EHRs generation bypasses the need to share sensitive real patient records. However, existing methods generate single-modal EHRs by unconditional generation or by longitudinal inference, which falls short of low flexibility and makes unrealistic EHRs. In this work, we propose to formulate EHRs generation as a text-to-text translation task by language models (LMs), which suffices to highly flexible event imputation during generation. We also design prompt learning to control the generation conditioned by numerical and categorical demographic features. We evaluate synthetic EHRs quality by two perplexity measures accounting for their longitudinal pattern (longitudinal imputation perplexity, $1p_l$) and the connections cross modalities (cross-modality imputation perplexity, $mp_l$). 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 $1p_l$ and 45.3\% decrease of $mp_l$ on average compared to the best baselines) with low privacy risks.\textsuperscript{1}

1 Introduction

The prevalence of electronic patient healthcare records fuel the development of machine learning models for many healthcare applications (Choi et al., 2016a,b; Wang et al., 2021a,b; Wang and Sun, 2022a). However, sharing EHR data usually undergoes strict and expensive de-identification and administration processes thus being difficult. Although there have been attempts on perturbing potentially identifiable attributes as the de-identification step (Emam et al., 2015), they were argued not immune to the hack for re-identification (El Emam et al., 2011; Choi et al., 2017). Alternatively, generating synthetic but realistic EHRs can circumvent data leakage while preserving the patterns of real EHRs for further research and development (Biswal et al., 2020).

Deep generative models like GANs (Goodfellow et al., 2014) and VAEs (Kingma and Welling, 2013) have become popular for unconditional EHRs generation (Choi et al., 2017) and longitudinal EHRs generation (Biswal et al., 2020; Zhang et al., 2020) for diagnosis codes. However, EHRs are often multimodal with different types of events, including diagnoses, procedures, medications, and also patient baseline demographic features like age and gender (Johnson et al., 2016). GANs & VAEs usually struggle to model complex multimodal and non-Gaussian distributions as well as sparse one-hot-encoded vectors (Xu et al., 2019). By contrast, generative language models (LMs) are proved highly powerful to represent large and complex distributions on discrete data (e.g., texts) (Liu et al.,...
in this work, we propose to leverage generative language models (LMs) for EHRs generation. We try to generate a sequence of visits with mixed types of events, e.g., diagnosis and medications. As Fig. 1 shows, previous works make unconditional generation for single-modal static EHRs (Choi et al., 2017) or for single-modal longitudinal EHRs (Zhang et al., 2021). However, real EHRs are heterogeneous with multiple types of temporal events and have baseline patient features, e.g., demographic information. We seek to (1) generate realistic mixed-type longitudinal EHRs with scale and (2) support flexible conditional generation to fit the need for personalized EHRs. Specifically, our contributions are

- We propose a new EHRs generation method making the best of LMs, which enables generating multimodal EHRs.
- We design prompt learning for controllable and flexible EHRs generation with LMs.
- We design comprehensive evaluation for both quality and privacy of the generated EHRs.

2 Related Works

2.1 EHRs Generation

Early works on generating EHRs (Lombardo and Moniz, 2008; Buczak et al., 2010; McLachlan et al., 2016) are rule-based methods. However, they were argued not capable of providing realistic data for machine learning tasks and were still vulnerable to re-identification (Choi et al., 2017). Deep generative models advanced by the power of deep learning, e.g., variational auto-encoders (VAE) (Kingma and Welling, 2013) and generative adversarial network (GAN) (Goodfellow et al., 2014), gained most attention recently. Choi et al. (2017) pioneered in adapting GAN for discrete patient records generation, namely MedGAN, which was followed by improving GANs for EHRs generation (Guan et al., 2018; Baowaly et al., 2019; Zhang et al., 2020); using VAE (Biswal et al., 2020), hybrid GANs (Lee et al., 2020; Cui et al., 2020), or conditional GANs (Xu et al., 2019). However, most methods only generate static tabular EHRs or longitudinal single-modal EHRs. GANs are often riddled with mode collapse, non-convergence, and instability, which cause their training tricky in practice (Saxena and Cao, 2021). Moreover, due to the representation limit, GANs struggle in modeling multimodal distributions and sparse one-hot-encoded vectors (Xu et al., 2019) while EHRs are with these properties. By contrast, we bypass these challenges by LMs. A comprehensive review of EHR synthesis is provided by Wang et al. (2022).

2.2 Language Models & Prompt Learning

LMs are often used for text generation tasks attributed to their auto-regressive nature, e.g., T5 (Raffel et al., 2020) and BART (Lewis et al., 2020). Nonetheless, they cannot be directly applied to EHRs generation since EHRs consist of not only plain clinical notes but also longitudinal sequences of events. Although there were works on encoding and generating medical texts by LMs (Amin-Nejad et al., 2020; Libbi et al., 2021; Kagawa et al., 2021; Wang and Sun, 2022b), none has been done for synthetic EHRs generation. Prompt learning was used to control the topic of text generation (Li and Liang, 2021; Yu et al., 2021; Qian et al., 2022). However, they only consider one-hot encoded topics as prefix. In this work, we leverage prompt learning for EHRs generation conditioned on patient baseline features, which include both categorical and numerical values.

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 EHRs generation models.

3.1 Problem Formulation

Consider there are $N$ patients where the $n$-th patient is represented by $X_{n,1:T_n} = \{x_{n,1}, x_{n,2}, \ldots, x_{n,T_n}\}$ where $x_{n,t}$ are the baseline features, e.g., age and gender; $x_{n,t}$ signifies events happened at the $t$-th visit; $T_n$ is the total number of visits. For each visit $x_{n,t}$, we have $K$ types of events as $x_{n,t} = \{x_{n,t}^1, x_{n,t}^2, \ldots, x_{n,t}^K\}$. $x_{n,t}^k = \{c_1, c_2, \ldots, c_l\}$ are all events of type $k$, $l$ is the number of events.

We formulate three basic functions to support EHRs generation:

\[
\begin{align*}
\text{We propose to leverage generative language models (LMs) for EHRs generation.} \quad \text{We try to generate a sequence of visits with mixed types of events, e.g., diagnosis and medications.} \\
\text{As Fig. 1 shows, previous works make unconditional generation for single-modal static EHRs (Choi et al., 2017) or for single-modal longitudinal EHRs (Zhang et al., 2021).} \quad \text{However, real EHRs are heterogeneous with multiple types of temporal events and have baseline patient features, e.g., demographic information.} \\
\text{We seek to (1) generate realistic mixed-type longitudinal EHRs with scale} \quad \text{and (2) support flexible conditional generation to fit the need for personalized EHRs.} \\
\text{Specifically, our contributions are} \\
\text{- We propose a new EHRs generation method making the best of LMs, which enables generating multimodal EHRs.} \\
\text{- We design prompt learning for controllable and flexible EHRs generation with LMs.} \\
\text{- We design comprehensive evaluation for both} \quad \text{quality and privacy of the generated EHRs.} \\
\end{align*}
\]
3.2 Encoding

The overview is shown by Fig. 2. The first step is to transform the raw inputs $X_{n,1:T_n}$ to token sequences hence acceptable to the encoder.

**Inputs tokenization.** PromptEHR is compatible with all sequence-to-sequence models (Cho et al., 2014). We choose to utilize BART (Lewis et al., 2020) as the base model. BART uses a bidirectional encoder thus allowing arbitrary corruption for the input sequences and a left-to-right decoder to reconstruct the inputs. Motivated by the application of prompts in language (Liu et al., 2021a), we leverage prompts to specify the inputs. Without loss of generality, we assume two modalities: diagnosis (DX) and medication (Med). Denote $[X]$ and $[Z]$ as the input and answer slots, we can formulate the longitudinal imputation task by a prefix prompt problem: $\langle v \rangle [X] \langle v \rangle [Z]$. The model tries to fill the answer slot $[Z]$ which are the events in the next visit; the cross-modal imputation task is built by a cloze prompt problem: $[X] \langle dx \rangle [Z]$ where $\langle dx \rangle$ signifies the start of diagnosis events and $[X]$ represents the multimodal context events.

**Conditional prompt featurizer.** We introduce conditional prompt embeddings to enable conditional generation based on patient features. We consider both categorical $\mathbf{x}_{\text{cat}}$ and numerical features $\mathbf{x}_{\text{num}}$. The categorical prompt embeddings $\mathbf{E}_{\text{cat}}$ is obtained by

$$\mathbf{E}_{\text{cat}} = (\mathbf{x}_{\text{cat}} \mathbf{W}_0 + \mathbf{b}) \mathbf{W}_1. \quad (1)$$

$\mathbf{x}_{\text{cat}}$ has $m_c$ multi-hot encoded indices indicating the classes of each feature; $\mathbf{W}_0 \in \mathbb{R}^{m_c \times d_0}$, $\mathbf{W}_1 \in \mathbb{R}^{d_0 \times d_1}$. Therefore, $\mathbf{e}_{\text{cat}}$ encodes the instruction of $\mathbf{x}_{\text{cat}}$ and steers the LM to generate specific populations. We transform $\mathbf{x}_{\text{num}} \in \mathbb{R}^{m_u}$ to $\mathbf{e}_{\text{num}}$ with another set of $\mathbf{W}_0$, $\mathbf{W}_1$, and $\mathbf{b}$. $\mathbf{E}_{\text{cat}}$ and $\mathbf{E}_{\text{num}}$ then prepend to token embeddings by

$$\mathbf{E} = [\mathbf{E}_{\text{cat}}; \mathbf{E}_{\text{num}}; \mathbf{E}_{\text{tok}}] \quad (2)$$

to serve as the inputs to the encoder. We build the inputs for the decoder with the other featurizer to get $\mathbf{E}'_{\text{cat}}$ and $\mathbf{E}'_{\text{num}}$ and the shared token embeddings $\mathbf{E}_{\text{tok}}$.

3.3 Decoding & Training

The inputs tokens for the decoder are shifted encoder inputs such that the decoder predicts the next
token based on the prior tokens. Denote the context by $X$ and the target event by $x$, the true conditional distribution is $p(x|X)$. For instance, in the longitudinal imputation task, the context is the historical record of the patient $X_{1:t}$ and the target is the events in the next visit $x_{t+1}$. Correspondingly, $p(x|X; \theta)$ is the prediction made by the model. We use $X \sim q(X)$ to represent the perturbations added to the context inputs. The training objective is to minimize the negative log-likelihood as

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

The model is hence pushed to maximize the predicted probability to the true next tokens $x$ conditioned by the corrupted inputs $X$.

We apply the following corruptions during training: (1) Token mask, infill, and deletion; (2) Span shuffle and permutation. For (1), we randomly replace multiple tokens with <mask> or delete as length $\sim$ Poisson(3). For (2), we randomly shuffle the tokens within the same visits and shuffle the modality orders in the same visits.

### 3.4 Harmless Randomness in Generation

Apart from preciseness, the diversity of the generated data is also of great importance. PromptEHR samples from the conditional distribution by

$$x \sim p(x_t|X_{1:t-1}; \theta),$$

which allows to adjust diversity by many techniques existing in natural language generation literature. For instance, to prevent low probability events, we can apply top-$k$ sampling (Fan et al., 2018). Temperature is also useful to flatten or sharpen the conditional distribution. More advanced methods, e.g., beam search (Welleck et al., 2019) and nucleus sampling (Holtzman et al., 2019) are all available for exploitation by PromptEHR, which brings a great potential to achieve higher quality EHRs with diversity. By contrast, GANs & VAEs depend on sampling random noise vectors to introduce diversity, which is not controllable and usually undermines generation quality.

### 3.5 Quality Evaluation

We provide a recipe to evaluate EHRs generation 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 (Neubig, 2017):

$$\text{ppl} = e^{-(\log \prod_{t=1}^{T} p(v_t|x_{1:t-1}; \theta))/L},$$

where $p(v_t|x_{1:t-1})$ indicates how the model predicts the next word using all previous words as the context; $L$ is the length of the document; $\theta$ is the model parameter. Intuitively, a random predictor will produce ppl that is equal to the cardinality of vocabulary $|\mathcal{C}|$. We hereby adapt it to the longitudinal imputation perplexity (1p1) and cross-modality imputation perplexity (mpl) taking the structure of EHR into account.

1p1 takes the temporal coherence of the patient visits into account. For instance, chronic diseases like diabetes can cause complications (e.g., heart disease and kidney failure) in the future. Following Eq. (5), we can write the 1p1 of a patient’s records $X = \{x_1, \ldots, x_T\}$ as

$$1p1 = e^{-\sum_{t=1}^{T} \log p(x_t|x_{1:t-1}; \theta)/(t*st)}$$

$$= e^{-\sum_{t=1}^{T} \sum_{1:l}^{T} \log p(v_t|x_{1:t-1}; \theta)/(l*st)}.$$ 

Here, $x_t = \{c_1, \ldots, c_t\}$ are all events during the $t$-th admission. Inside this admission, concurrent events are independently generated conditioned on previous visits, therefore we can decompose $p(x_t|x_{1:t-1}; \theta) = \prod_{l=1}^{K} p(c_l|x_{1:t-1}; \theta)$ then come to the results.

mpl accounts for the correlations between modalities. For example, high body temperature in lab test may correspond to fever in diagnosis. We focus on the $t$-th admission where the joint distribution of all $K$ modalities $p(x_1^t, \ldots, x_K^t|x_{1:t-1}; \theta)$. We can write the NLL here by

$$\text{NLL}_d = -\frac{1}{K} \sum_{k=1}^{K} \log p(x_k^t|x_{1:t-1}; \theta)$$

$$=-\frac{1}{K} \sum_{k=1}^{K} \sum_{l=1}^{I_k^t} \log p(v_l^t|x_{1:t-1}; \theta),$$

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

$$\text{mpl} = e^{\sum_{t=1}^{T} \text{NLL}_d/T}.$$ 

### 3.6 Privacy Evaluation

It is crucial to measure the privacy preserving when sharing the synthetic data. We try to evaluate two privacy risks: **membership inference** and
attribute inference. We split the data into the training data $D_1 = \{X_{n,1:T_n}\}_{n=1}^N$ and testing data $D_2$, and generate synthetic data $D_S$ with the same length as $D_1$.

Membership Inference. Attackers would try to infer the membership of the patient records based on the real records they own. We design this adversary based on shadow training (Shokri et al., 2017). In the first stage, a shadow model $M_{sd}$ is trained on $D_S$. 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 $M_{sd}(X)$ where $X \in D_S \cup D_2$. $D_S$ is a subset of $D_S$ with the same number as $D_2$. A model $M_{mi} : Y_{\text{ptt}} \rightarrow \{0, 1\}$ is trained to differentiate if $X$ comes from $D_S$ or $D_2$. We will then evaluate the success rate of $M_{mi}$ on identifying $X \in D_1 \cup D_2$. The better the adversary $M_{sd}(X)$ 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 (Zhang et al., 2021). 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 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 $P_0 = p(v_l | \{v_{l,1}, \ldots, v_{l,T_l}\})$. With the prior, the attacker will train an attribute imputation model on the synthetic data $D_S$, i.e., $\hat{P} = p(v_l | \{v_{l,1}, \ldots, v_{l,T_l}\})$. The attacker then believe the code $v_l$ exists when $\log \hat{P} - \log P_0 \geq \delta$. $\delta$ is a pre-defined threshold. In experiments, we train another attribute imputation model on $D_1$ to approximate the prior knowledge. We evaluate the success rate of this attack. Besides, we create a control arm where another imputation model is trained on the test set. Comparison between the control and the treatment (imputation model trained on $D_S$) suffices for an immediate evaluation of the synthetic data’s risk level.

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 the secondary use by predictive modeling in practice?

• Q4. How is the generation quality of PromptEHR influenced by the size of training records?

4.1 Experimental Setup

Dataset. (Johnson et al., 2016) We use MIMIC-III data 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.

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Number</th>
<th>Event Type</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patients</td>
<td>46,520</td>
<td>Diagnosis</td>
<td>1,071</td>
</tr>
<tr>
<td>Total Visits</td>
<td>58,976</td>
<td>Drug</td>
<td>500</td>
</tr>
<tr>
<td>Total Events</td>
<td>5,401,961</td>
<td>Procedure</td>
<td>668</td>
</tr>
<tr>
<td>Events per Patient</td>
<td>116</td>
<td>Lab Test</td>
<td>185</td>
</tr>
</tbody>
</table>

Baselines. We compare the following baselines:

• LSTM+MLP. This is the baseline that leverages LSTM (Hochreiter and Schmidhuber, 1997) 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 (Choi et al., 2017). 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.

• SynTEG (Zhang et al., 2021). 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 (Vaswani et al., 2017) for temporal dependency learning and conditional Wasserstein
Figure 3: Perplexity compared between generation w/ (cond.) and w/o conditional prompts (w/o cond.) for four types of events. Note that both lpl and mpl are the less the better.

(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 $\delta$.

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 $D_S$ by PromptEHR; the Control curves indicate the one trained on test set $D_2$.

GAN with gradient penalty (WGAN-GP) (Arjovsky et al., 2017; Gulrajani et al., 2017) for event inference.

• GPT-2 (Radford et al., 2019). We pick GPT-2 as the LM baseline that only does causal language modeling on EHRs. Then, it is able to do event generation like texts generation.

4.1.1 Evaluation metrics

We use the proposed lpl and mpl to evaluate generation quality. 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.

We use two adversaries: membership inference (MI) and attribute inference (AI), to test the privacy risk. In MI, we use LSTM+MLP as the shadow model to mimic the outputs of PromptEHR. A three-layer MLP predicts the membership. ROC curve is plotted to evaluate the attack success rate; In AI, we train an LSTM+MLP on $D_1$ to approximate the prior and another LSTM+MLP on $D_S$ as the attribute imputation model.

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 which also appear in the following figures.

To test the utility of the synthetic data for downstream predictive tasks, we train LSTM+MLP on $D_S$ or $D_2$ and test it on $D_2$ to compute the recall@20/30.

4.2 Implementation Details

All the used LSTM+MLP model consists of 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 (Kingma and Ba, 2014). The 12-layer transformer based pre-trained 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.

For PromptEHR, we use BART model as the backbone (Lewis et al., 2020). 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 stage, the perplexity computed on the validation set is used to pick the best checkpoint. All experiments are conducted with an RTX-3090 GPU, 251 GB RAM, and AMD Ryzen Threadripper 3970X 32-core CPU.

4.3 Q1. Generation Quality

The calculated mpl and lpl of all show in Table 2. It is 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. GPT-2 works better than LSTM+MLP on temporal perplexity crediting to its power in capturing series pattern through transformers.

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

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

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

Figure 6: Recall 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.

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 on mpl than on lpl. GPT-2 is trained by causal language modeling task where it models the sequence autoregressively. Without the prompt design, it is confused by the order of events within the same visit, which induces deteriorating performance.

Fig. 3 demonstrates the comparison made between generation w/ and w/o conditional prompts for PromptEHR. We identify that conditional prompts significantly improve the generation quality as they provide important characteristics of the patients. We are hence able to generate for specific populations with input prompts.

4.4 Q2. Privacy Evaluation

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 \( D_1 \cup D_2 \). It clearly shows the MI model has bad performance that is near random guess (AUC ≃ 0.5), which means the MI attack gains no sensitive membership information when trained on the synthetic data \( D_S \).

Fig. 4b shows the TPR/FPR of attribute inference attack based on shadow training with the varying threshold \( \delta \). Here, we cut the curve where \( \delta = 4 \) because all the remaining curves are approaching zero on its right. The threshold \( \delta \) adjusts to the confidence level of the attacker, i.e., the smaller \( \delta \) 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 of the control model. The TPR then keeps a much lower level when \( \delta \) increases, which implies the low attribute leakage risk of the synthetic data generated by PromptEHR. Although the FPR 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.

4.5 Q3. Synthetic EHRs Utility

We aim to measure the utility of synthetic data when we develop predictive models on top of them. We compare LSTM models on \( D_S \) and \( D_1 \) with multilabel prediction for diagnosis events similar to the setting in (Choi et al., 2016b). In particular, 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 real data where the synthetic data is regarded as data augmentation.

**Fully synthetic data.** 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 surprisingly find 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. It is expected to achieve comparable performance by synthetic data as the real data.

**Hybrid synthetc-real data.** In Fig. 6, we randomly sample 10k real data from $D_1$ and combine them with different sizes of synthetic data from $D_5$. 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 50k synthetic records plus 30k real records train better models than on all the real data.

### 4.6 Q4. Quality w.r.t. Training 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 (~40k) in red dotted lines for comparison. It shows that PromptEHR trained on 5k training records 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.

### 4.7 Case Study

We demonstrate two use cases of PromptEHR: generating from scratch (Table 3) and generating by completion (Table 4). While previous works handle the former, only PromptEHR handles the completion setting because it makes flexible conditional generation based on either patient features or previous events. In Table 4, our model begins from all diagnosis of one patient and then generates labtests via cross-modal imputation. Then, we randomly sample one procedure and let the model impute all the remaining procedures based on diagnosis and the labtests. Iteratively applying this strategy yields diverse and realistic EHRs via conditional generation. We provide explanations of the two synthetic records in Appendix §A.

### 5 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. To evaluate its performance, we draw the idea of perplexity from the text generation literature and propose two per-
plexity measures: spatial and temporal perplexity. Experiments on MIMIC-III data demonstrates the quality of generated EHRs are better than the baselines. The synthetic data provides both utility and privacy for downstream healthcare applications.

Acknowledgement

This work was supported by NSF award SCH-2205289, SCH-2014438, IIS-1838042, NIH award R01 1R01NS107291-01.

Limitations

This work seeks to generate synthetic records hence avoid sharing sensitive personal electronic healthcare records for the development of machine learning models. In our experiments, we find the generated synthetic records by PromptEHR are invulnerable to two adversaries: membership inference and attribute inference. However, there is still possibility that there exists some more advanced attacking methods which can take the advantage of synthetic records. Obviously we cannot exhaust all adversaries for empirical privacy evaluation. In this viewpoint, it is promising to investigate theoretic-guaranteed EHRs generation approach. For instance, we may draw the idea of differential privacy to enhance the current method to provide a complete privacy protection.

References


A Case Study

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.

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

<table>
<thead>
<tr>
<th>Visit-1</th>
<th>Diagnosis: Liveborn</th>
<th>Labtest: Hematocrit</th>
<th>Procedure: Prophylactic vaccination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Procedure: Biopsy of spinal cord</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4: A synthetic patient generated by PromptEHR based on a real patient record. The imputed events are marked yellow. For demonstration, we cut the events after the fifth for each visit due to the space limit.

<table>
<thead>
<tr>
<th>Visit</th>
<th>Diagnosis</th>
<th>Labtest</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pneumonia, Hematemesis, Heart failure, Emphysema</td>
<td>Leukocytes, Urea Nitrogen, Calcium, Ketone</td>
<td>Enteral infusion of nutrition, Insertion of airway, Replace tracheostomy tube, Temporary tracheostomy</td>
</tr>
<tr>
<td>2</td>
<td>Heart failure, Respiratory conditions, Tracheostomy status, Stomach disorder</td>
<td>Urine Appearance, Yeast, Platelet Count</td>
<td>Biopsy of bronchus, Replace gastrostomy tube, Invasive mechanical ventilation, Infusion of nesiritide</td>
</tr>
<tr>
<td>3</td>
<td>Pneumonia, Mechanical complication, Pulmonary manifestations, Disorders of urinary tract</td>
<td>INR(PT), Epithelial Cells, RBC, Urine Appearance</td>
<td>Insertion of airway, Enteroctomy, Lysis of peritoneal adhesions, Lung biopsy</td>
</tr>
<tr>
<td>4</td>
<td>Mechanical complication, Hodgkin’s paragranuloma, Pressure ulcer, Heart failure</td>
<td>Urine Color, Urobilinogen, Bands, Urea Nitrogen</td>
<td>Urine Color, Urobilinogen, Bands, Urea Nitrogen</td>
</tr>
<tr>
<td>5</td>
<td>Urethra disorder, Attention to tracheostomy/gastrostomy, Pneumonia, Heart failure</td>
<td>MCH, Bacteria, Lymphocytes, Calculated Total CO2</td>
<td>Fluticasone Propionate 110mcg, SW, Bisacodyl, Iso-Osmotic Dextrose</td>
</tr>
<tr>
<td>6</td>
<td>Pneumonia, Heart failure, Endomyocardial fibrosis, Mechanical complication</td>
<td>pH, Epithelial Cells, WBC, Protein</td>
<td>Neutra-Phos, Mirtazapine, Fluconazole, SW</td>
</tr>
</tbody>
</table>