Rare Disease Prediction by Generating Quality-Assured Electronic Health Records*

Fenglong Ma*† Yaqing Wang*‡ Jing Gao§ Houping Xiao¶ Jing Zhou‖

Abstract

Predicting diseases for patients is an important and practical task in healthcare informatics. Existing disease prediction models focus on common diseases, i.e., there are enough available EHR data and prior medical knowledge for analyzing them. However, those models may not work for rare disease prediction as it is extremely hard to collect enough EHR data with such diseases. To tackle these issues, in this paper, we design a novel rare disease prediction system, which not only generates EHR data but also automatically selects high-quality generated data to further improve the predictive performance. Three components are designed in the system: data generation, data selection, and prediction. In particular, we propose MaskEHR to generate diverse EHR data based on the data from patients suffering from the given diseases. To remove noise information in the generated EHR data, we further design a reinforcement learning-based data selector, called RL-Selector, which can automatically choose the high-quality generated EHR data. Finally, the prediction component is used to identify patients who will potentially suffer the given diseases. These three components work together and enhance each other. Experiments on three real healthcare datasets show that the proposed system outperforms existing approaches on rare disease prediction task.

1 Introduction

The advent of massive *Electronic Health Records* (EHR) makes it possible to predict patients' health status, and thus motivates studies on the predictions of diagnosis [7, 8, 16, 19], risk and disease [3, 5, 17]. Especially, the *disease prediction task* aims to predict whether a patient will suffer a certain disease based on the historical EHR data. Recently, deep learning based models have shown improved performance on disease and risk prediction tasks. In [5], the authors propose a simple convolutional neural network (CNN) based prediction model. To incorporate medical knowledge into the prediction models, the authors in [17] design a framework PRIME to improve the prediction performance. It is well-known that training these models needs abundant disease-specific EHR data. However, for those **rare diseases**[1] that affect only a small percentage of the population, it is extremely hard to collect enough patients' EHR data. Moreover, the prior medical knowledge on rare diseases is usually too little to make accurate diagnosis for the patients. Therefore, directly applying existing approaches on such data is impractical.

The key problem of rare disease prediction is *how to obtain enough training data from patients with the target disease (i.e., case patients' data).* An intuitive idea is to directly generate EHR data only based on case patients. Since the number of control patients, i.e., the patients who do not suffer the given rare disease, is far greater than that of case ones in the database, the generative models should focus on generating more case patients rather than control patient data. Though traditional data generation models [3, 9] can be used to generate EHR data, the generated data may have very low quality because EHR data generation has its unique challenges discussed as follows.

When generating fake EHR data, existing models [3,9] only use the current visit information but ignore all the previous visits. Though they still can generate EHR data, those generated data all lose the *temporal characteristics among visits.* Besides, the generated data should be *task-specific.* The goal of data generation is not only to increase the number of case patients, but also to improve the final prediction, i.e., guaranteeing the quality of the generated data. However, current EHR data generation approaches [3, 9] cannot assure the quality of the generations. The low-quality data may further hurt the performance of the target task. Therefore, *how to generate high-quality EHR data by considering the temporal characteristics among visits is the challenge.*

To tackle the aforementioned challenge, in this paper, we introduce a novel rare disease prediction system, which consists of three components: data generation, data selection and prediction. In the data generation

*The first and second authors contributed equally.
†Pennsylvania State University, fenglong@psu.edu.
‡University at Buffalo, yaqingwa@buffalo.edu.
§University at Buffalo, jing@buffalo.edu
¶Georgia State University, hxiao@gsu.edu
‖eHealth Inc., jing.zhou@ehealth.com

[1] https://en.wikipedia.org/wiki/Rare_disease

component, we propose a new case patient generation approach in Section 4, called MaskEHR, which employs a recurrent neural network (RNN) as the generator to model the *temporal characteristics* among visits. However, there still exist many low-quality patient visits among the generated samples with MaskEHR. To further *automatically choose high-quality samples*, we design a reinforcement learning-based data selector (named RL-Selector) in the data selection component (Section 5). The reward calculated by RL-Selector guides MaskEHR to generate diverse and high-quality case patient data. It is clear that the designed three components mutually enhance each other. Experimental results on three rare disease datasets show that the proposed system can significantly improve the prediction performance and generate high-quality data. It is worthwhile to highlight the contributions of our work:

- To the best of our knowledge, this is the first end-to-end deep learning-based system for rare disease prediction task, which utilizes both deep generative model and reinforcement learning techniques together to improve the performance.
- We recognize the uniqueness of generating sequential EHR data, transform the difficult generation problem into the problem of filling in masked visits, and propose a novel model MaskEHR to generate the visits of patients with rare diseases.
- We introduce reinforcement learning technique to the disease prediction task and design a reinforcement learning-based data selector (i.e., RL-Selector) to assure the quality of the generated "fake" data.
- We conduct extensive experiments on three datasets to demonstrate the effectiveness of the designed system for rare disease prediction task. In addition, analyses are conducted to illustrate the importance of each component respectively.

2 Related Work

2.1 Disease Prediction Disease prediction can be regarded as a classification task, and many traditional classification approaches have been applied to solve this task. Recently, deep learning-based approaches are proposed to mine knowledge from EHR data [10, 21]. Among these approaches, RNNs are used to classify diagnoses [1, 7, 8, 15, 16, 18, 23, 27], identify patient subtyping [2], and model disease progression [22]. There are a few deep learning-based approaches proposed for disease prediction task [5, 17, 30]. Though these models are effective for disease prediction task, training both models requires a large amount of EHR data. However, for rare diseases, the number of patients is extremely small, so directly applying existing approaches may result in unsatisfactory performance.

Some generative models, such as [3] and [9], are proposed to generate EHR data, in order to enhance the training set and improve prediction performance. However, these generative approaches are not designed for rare disease prediction and may not perform well when there are insufficient cases to start from. Additionally, the generated EHR data are not diverse as the data generation is only based on the current visit without considering the temporal characteristics of EHR data. Moreover, it is inevitable that some of the generated data are of low quality, which may cause the degrading of the prediction performance.

2.2 Deep Generative Networks with Reinforcement Learning The adversarial learning framework, especially generative adversarial networks (GAN) [13], has been successfully used in several tasks, such as image generation [4, 31] and domain adaption [12, 14, 25]. The core idea of adversarial learning framework is to design a set of competing components which learn together. Existing GAN models mostly focus on continuous data, but recently the extension of GAN to discrete space attracts considerable attention. Since discrete elements break the differentiability, reinforcement learning has usually been incorporated to tackle this problem. SeqGAN [28] trains a language model to fool the discriminator by policy gradients and uses Monte Carlo rollouts to get a loss signal on each word. In order to precisely evaluate the reward of every token in a sequence, MaskGAN [11] trains the generator to fill in missing text conditioned on the surrounding context and uses an actor-critic method to obtain reward signals for the generated sequence.

3 Terminologies & Overview

DEFINITION 1. (DIAGNOSIS CODE SET) *The diagnosis code set is* $\mathcal{C} = \{c_1, c_2, \cdots, c_{|\mathcal{C}|}, c_{mask}\}$*, where* $|\mathcal{C}|$ *is the number of unique diagnosis codes from the EHR data, and* c_{mask} *is named mask code to indicate whether a visit is masked or not.*

DEFINITION 2. (VISIT) *The visit of the p-th patient at time t is denoted as a binary vector* $\mathbf{x}_t^{(p)} \in \{0,1\}^{|\mathcal{C}|+1}$*. If the visit contains the diagnosis code* $c_i \in \mathcal{C}$*, then the i-th element of* $\mathbf{x}_t^{(p)}$ *is 1.* $n_t^{(p)}$ *denotes the number of diagnosis codes in the t-th visit of patient p.*

DEFINITION 3. (VISIT RECORDS) *The visit records of the p-the patient are represented as a matrix* $\mathbf{X}^{(p)} = [\mathbf{x}_1^{(p)}, \mathbf{x}_2^{(p)}, \cdots, \mathbf{x}_T^{(p)}] \in \mathbb{R}^{T \times (|\mathcal{C}|+1)}$*, where T is the number of visit records for the p-th patient. Note that for different patients, T may be different.*

DEFINITION 4. (DISEASE PREDICTION) *Given the p-th*

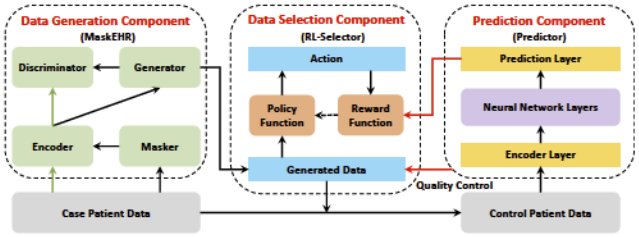


Figure 1: Overview of the Proposed System for Rare Disease Prediction Task.

patient's visit records $\mathbf{X}^{(p)} = [\mathbf{x}_1^{(p)}, \mathbf{x}_2^{(p)}, \cdots, \mathbf{x}_T^{(p)}]$, *disease prediction task aims to identify whether the input patient suffers the given disease, i.e.,* $y^{(p)} \in \{0, 1\}$.

To avoid cluttered notation, we describe the algorithm for a single patient and drop the superscript (p) where it is unambiguous.

Figure 1 shows an overview of the proposed end-to-end system for rare disease prediction, which consists of three components: data generation, data selection and prediction. The data generation component aims to generate "fake" patients' visits based on the input case patient data. Toward this end, we proposed a new EHR data generation approach, called MaskEHR. The data produced by the data generation component may contain noisy records. To select the high-quality generated samples from such noisy data, we design a reinforcement learning-based data selector, called RL-Selector, which can be used to train a good predictor. In the prediction component, the inputs are the case and control patient data and the selected data. We develop a neural network based approach to make predictions. Next, we will provide the details of each component in the following sections.

4 EHR Data Generation

Different from text generation with MaskGAN [11], generating time-ordered EHR data is more challenging due to their own characteristics. On one hand, *the diagnosis codes within each visit are without any order*, which leads to the failure of traditional approaches for text generation. On the other hand, even if each visit can be seen as a word and the whole visits can be considered as a sentence, text generation models still cannot work. The reason is that when generating a missing visit, we have to *estimate the number of diagnosis codes in this visit*. However, in text generation, they simply select the term with the highest probability as the filled-in missing word. To address these unique challenges of EHR data, MaskEHR is proposed, which consists of the following four parts: masker, encoder, generator and discriminator, as shown in Figure 2. MaskEHR tasks a patient's visit records $\mathbf{X} \in \mathcal{D}^+$ as input and generates "fake" patient visits $\hat{\mathbf{X}}$ based on the input $\mathbf{X}$, where $\mathcal{D}^+$ denotes the set of case patients in the training dataset.

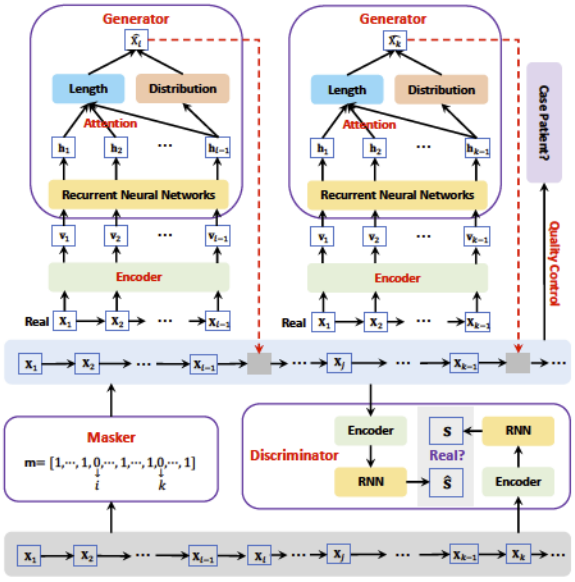


Figure 2: Overview of the Proposed MaskEHR Model.

4.1 Masker and Encoder The *masker* aims to randomly delete some visits from $\mathbf{X}$ according to a random vector $\mathbf{m} \in \{0, 1\}^T$. $\mathbf{m}_t = 0$ means that the t-th visit has been removed from $\mathbf{X}$. We assume that the first visit cannot be masked, i.e., $\mathbf{m}_1$ is always equal to 1. Note that we introduce a parameter $\eta \in [0, 1]$ to control the masked percentage of the input visits. This parameter is very important since it directly affects the final prediction performance. If η is small, then the proposed model may repeat the input data. If η is large, then the quality of the generated data may be low. Even with the same masked percentage, the mask vector may have multiple values. It enables us to generate different EHR data, which meets the requirement of diversity.

Given the input case patient $\mathbf{X} \in \mathcal{D}^+$ and the mask vector $\mathbf{m}$, we can obtain the masked visit matrix $\mathbf{m}(\mathbf{X}) \in \mathbb{R}^{T \times (|\mathcal{C}|+1)}$. Each visit $\mathbf{x}_t \in \mathbf{X}$ is then mapped to a v-dimensional vector via the encoder as follows:

$$\mathbf{v}_t = f_{\mathsf{ENC}}(\mathbf{x}_t; \mathbf{\Phi}_v) = \mathbf{W}_v \mathbf{x}_t + \mathbf{b}_v, \tag{4.1}$$

where $\mathbf{\Phi}_v = \{\mathbf{W}_v \in \mathbb{R}^{v \times (|\mathcal{C}|+1)}, \mathbf{b}_v \in \mathbb{R}^v\}$ is the parameter set. The embeddings from the encoder are taken as the input data of the generator.

4.2 Generator The goal of the proposed generator is to fill in the masked visits in $\mathbf{m}(\mathbf{X})$. In order to generate useful EHR data, we fill in one missing visit per time given all the previous real visit records. Assume that the k-th visit is masked, then we will use the real visits from $\mathbf{x}_1$ and $\mathbf{x}_{k-1}$ to generate the *discrete codes* in $\mathbf{x}_k$

(i.e., $\hat{\mathbf{x}}_k$).

• **Recurrent Neural Networks**. The generator employs a recurrent neural network (RNN) which consists of Gated Recurrent Units (GRU) [6] to adaptively capture dependencies among patient visits. Let $\mathbf{\Phi}_g$ denote the set of all the parameters in RNN, and the GRU can be simplified as follows:

$$\mathbf{h}_t = f_{\mathsf{GRU}}(\mathbf{v}_t; \mathbf{\Phi}_g). \tag{4.2}$$

Based on the hidden states $\{\mathbf{h}_1, \cdots, \mathbf{h}_i, \cdots, \mathbf{h}_t\}$ ($\mathbf{h}_i \in \mathbb{R}^g$) of GRU, we can not only learn the distribution of $\hat{\mathbf{x}}_k$, but also the number of diagnosis codes in $\hat{\mathbf{x}}_k$.

• **Distribution Calculation**. Like language modeling [20], to predict the k-th visit's diagnosis codes, we need to obtain the distribution of $\hat{\mathbf{x}}_k$. Towards this goal, a softmax layer with a fully connected layer is utilized, i.e., $\mathbf{d}_t = \text{softmax}(f_{\mathsf{DIS}}(\mathbf{h}_t; \mathbf{\Phi}_d))$, where $f_{\mathsf{DIS}}(\mathbf{h}_t; \mathbf{\Phi}_d) = \mathbf{W}_d\mathbf{h}_t + \mathbf{b}_d$, $\mathbf{W}_d \in \mathbb{R}^{(|\mathcal{C}|+1)\times g}$ and $\mathbf{b}_d \in \mathbb{R}^{|\mathcal{C}|+1}$ are parameters. Since the learned distribution $\mathbf{d}_t$ is a *continuous numerical* vector, it is impossible and impractical to use all the codes with non-zero probabilities as the final generation. Thus, to obtain a reasonable number for the *discrete* diagnosis codes, we design an attention-based approach to estimate the length of the t-th visit based on all the previous visits.

• **Attention-based Length Estimation**. Actually, the number of diagnosis codes in different patient visits is very random, and thus it is hard to estimate the length. A naive way is to use the average number of all the previous visits' lengths. However, we aim to generate high-quality EHR data, and this simple approach may introduce a lot of noise. To address this problem, we propose an attention-based length estimation method. The intuition behind this approach is that if the vector representations of two visits are similar to each other, then they may have the same length with a high probability.

Since the hidden state $\mathbf{h}_{k-1}$ can be used to predict the latent representation of the visit $\mathbf{x}_k$, we first calculate the attention score or similarity between each previous hidden state $\mathbf{h}_i$ ($1 \le i \le k-1$) and the current $\mathbf{h}_{k-1}$, which is $\alpha_i = f_{\mathsf{ATT}}(\mathbf{h}_{k-1}, \mathbf{h}_i; \mathbf{\Phi}_\alpha) = \mathbf{v}_\alpha^\top \tanh(\mathbf{W}_\alpha[\mathbf{h}_{k-1}; \mathbf{h}_i])$, where $\mathbf{\Phi}_\alpha$ represents the parameter set, including $\mathbf{W}_\alpha \in \mathbb{R}^{q\times 2g}$ and $\mathbf{v}_\alpha \in \mathbb{R}^q$. We can obtain an attention score vector $[\alpha_1, \cdots, \alpha_{k-1}]$, then a softmax layer is used to normalize this vector: $\alpha = \text{softmax}([\alpha_1, \cdots, \alpha_{k-1}])$. Since we assume that the first visit cannot be masked, the length of the k-th visit can be estimated according to $\hat{n}_k = \sum_{i=1}^{k-1} \alpha_i n_i (k \ge 2)$, where n_i denotes the number of diagnosis codes in the i-the visit. Based on the estimated length $\hat{n}_k$ and the distribution $\mathbf{d}_t$, the diagnosis codes with the highest top-$\lceil \hat{n}_k \rceil$ probabilities can be selected to represent the generated visit $\hat{\mathbf{x}}_k$. After filling in all the missing visits, we can obtain the generated data $\hat{\mathbf{X}}$.

• **Rare Disease Predictor**. To assure the quality of the generated data as much as possible, we design a quality control mechanism with the proposed predictor, which aims to identify the label of the generated data. Given the generated data $\hat{\mathbf{X}}$, each visit $\hat{\mathbf{x}}_t \in \hat{\mathbf{X}}$ is first embedded into a v-dimensional vector $\hat{\mathbf{v}}_t$ using Eq. (4.1). Then $\hat{\mathbf{v}}_t$ is fed into GRU to produce the hidden state $\hat{\mathbf{h}}_t$ with Eq. (4.2). We consider the final hidden state $\hat{\mathbf{h}}_T$ as the representation of $\hat{\mathbf{X}}$. Finally, we can obtain the probability of $\hat{\mathbf{X}}$ belonging to the case group as follows:

$$P(\hat{y} = 1) = \sigma(\hat{\mathbf{h}}_T; \mathbf{\Phi}_p) = \{\exp(-\mathbf{w}_p^\top \hat{\mathbf{h}}_T + b_p)\}^{-1}, \tag{4.3}$$

where $\sigma()$ is the sigmoid function, and $\mathbf{\Phi}_p = \{\mathbf{w}_p \in \mathbb{R}^g, b_p \in \mathbb{R}\}$ is the parameter set.

• **Loss of Generator**. The goal of the generator is to automatically produce high quality but "fake" EHR data. In particular, it can fill in missing visits, estimate the length of the generated visits, and assign labels for them. Thus, the loss function of the generator includes three parts:

$$\mathcal{L}_{\mathsf{G}} = \mathcal{L}_{\mathsf{VISIT}} + \lambda \mathcal{L}_{\mathsf{LENGTH}} + \mathcal{L}_{\mathsf{CONTROLLER}},$$

where

$$\mathcal{L}_{\mathsf{VISIT}} = \frac{1}{|\mathcal{D}'|} \sum_{p=1}^{|\mathcal{D}'|} \frac{1}{M_p} \sum_{i=1}^{M_p} \|\mathbf{x}_i^{(p)} - \mathbf{d}_i^{(p)}\|_2^2,$$

$$\mathcal{L}_{\mathsf{LENGTH}} = \frac{1}{|\mathcal{D}'|} \sum_{p=1}^{|\mathcal{D}'|} \frac{1}{M_p} \sum_{i=1}^{M_p} (n_i^{(p)} - \hat{n}_i^{(p)})^2,$$

$$\mathcal{L}_{\mathsf{CONTROLLER}} = -\frac{1}{|\mathcal{D}'|} \sum_{p=1}^{|\mathcal{D}'|} \log(P(\hat{y}^{(p)})),$$

$|\mathcal{D}'|$ denotes the number of generated data, M_p represents the number of masked visits in the p-th generated patient's data, and λ is a predefined parameter to maintain the three losses in the same value scale, which is set as 0.1 in the experiments.

4.3 Discriminator The proposed discriminator aims to correctly identify whether the input $\tilde{\mathbf{X}}$ is real or fake. If the input is the generated $\hat{\mathbf{X}}$ that contains M_p filled-in visits, instead of directly learning the probability on $\hat{\mathbf{X}}$, we calculate the average probability on a set of constructed data $\{\hat{\mathbf{X}}'_i\}_{i=1}^{M_p}$. Each constructed data consists of one filled-in visit and $T-1$ real visits. We believe that considering the quality on each filled-in

visit separately is better than calculating an overall probability on the original generated data $\hat{\mathbf{X}}$.

Next, we will introduce how to calculate the probability $p(\tilde{\mathbf{X}} = \mathbf{X})$. Similar to the rare disease predictor, we can obtain the vector representation of $\tilde{\mathbf{X}}$ (i.e., $\tilde{\mathbf{s}}$), and a sigmoid function is used to predict the realness of the input $\tilde{\mathbf{X}}$ as follows: $P(\tilde{\mathbf{X}} = \mathbf{X}) = \sigma(\tilde{\mathbf{s}}; \mathbf{\Phi}_s) = \{1+\exp(-\mathbf{w}_s^\top \tilde{\mathbf{s}}+b_s)\}^{-1}$, where $\mathbf{\Phi}_s = \{\mathbf{w}_s \in \mathbb{R}^g, b_s \in \mathbb{R}\}$ denotes the parameter set. For the generated data $\hat{\mathbf{X}}$, the probability can be obtained by: $P(\hat{\mathbf{X}} = \mathbf{X}) = \frac{1}{M_p}\sum_{i=1}^{M_p} P(\hat{\mathbf{X}}'_i = \mathbf{X})$.

Finally, the loss of the proposed discriminator is defined as follows:

$$\mathcal{L}_{\mathsf{D}} = -\frac{1}{|\mathcal{D}'| + |\mathcal{D}^+|}\left[\sum_{i=1}^{|\mathcal{D}'|}\log(1 - P(\hat{\mathbf{X}}^{(i)})) + \sum_{j=1}^{|\mathcal{D}^+|}\log P(\mathbf{X}^{(j)})\right]. \tag{4.4}$$

4.4 Loss of MaskEHR In the proposed MaskEHR model, the generator intends to produce "fake" EHR data to fool the discriminator, while the discriminator tries to identify which input data are fake. It means that the generator hopes to minimize $\mathcal{L}_{\mathsf{G}}$ and maximize the loss of the discriminator $\mathcal{L}_{\mathsf{D}}$ simultaneously, but the discriminator only tries to recognize the generated data by minimizing the loss $\mathcal{L}_{\mathsf{D}}$. Since the discriminator is only in charge of identifying the input data, it is not related to the "fake" data generation. Thus, when optimizing the parameters of the discriminator, we can only use Eq. (4.4). Let Δ_d denote the parameter set of the discriminator, and we aim to seek the optimal parameters $\hat{\Delta}_d$ to minimize the loss $\mathcal{L}_{\mathsf{D}}$, i.e.,

$$\hat{\Delta}_d = \underset{\Delta_d}{\operatorname{argmin}}\, \mathcal{L}_{\mathsf{D}}. \tag{4.5}$$

In the data generation procedure, the generator not only generates "fake" EHR data, but also fools the discriminator. It is a minimax game between the generator and the discriminator. Therefore, we can define the final loss of MaskEHR as the minimax game, i.e., $\mathcal{L}_{\mathsf{MaskEHR}} = \mathcal{L}_{\mathsf{G}} - \mathcal{L}_{\mathsf{D}}$. For the minimax game, the parameter set we seek is the saddle point of the final loss function. Let Δ_g represent all the parameters of the generator. By fixing the parameter set $\hat{\Delta}_d$ of the discriminator, we can minimize the generator loss function $\mathcal{L}_{\mathsf{G}}$ by seeking the optimal parameters $\hat{\Delta}_g$, and this process can be represented as:

$$\hat{\Delta}_g = \underset{\Delta_g}{\operatorname{argmin}}\, \mathcal{L}_{\mathsf{MaskEHR}}(\Delta_g, \hat{\Delta}_d). \tag{4.6}$$

5 Quality-Assured Data Selection

Utilizing the proposed MaskEHR, we can successfully generate EHR data based on the visits from case patients, which is denoted as $\mathcal{D}'$. Though they are similar to the real data, we cannot totally make sure that such data indeed represent the case patients. In other words, the labels of such generated data are unsure. Therefore, directly assigning positive labels (i.e., case patients) to them when training the predictor as [3, 9] is not reasonable. To address this issue, we propose a reinforcement learning-based data selector as shown in Figure 3. RL-Selector aims to choose high-quality generated EHR data to improve the performance of the designed predictor. Next, we give the details of the designed RL-Selector.

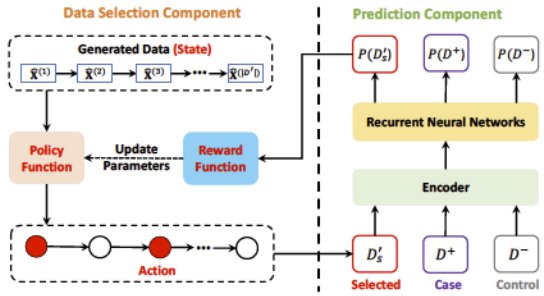


Figure 3: Overview of the Proposed RL-Selector Model.

In the designed RL-Selector, we define the action as choosing the current data or not and the state including the current data and all the chosen data. Moreover, it is obvious that the designed RL-Selector can obtain a *delayed reward* from the predictor when the data selector finishes all the selection. However, the proposed generator can continuously generate *unlimited* "fake" EHR data, which makes it impossible for RL-Selector to update the policy function. To tackle this problem, we force the data selector to terminate when collecting a certain number (referred to as $|\mathcal{D}'_s|$) of the generated data. The chosen dataset is then added into the training set. The predictor computes probabilities for the chosen data and sends feedback to the data selector for updating the policy strategy. In such a way, the RL-Selector can obtain more feedback and frequently update the policy function. Before introducing the detail implementation of each unit of reinforcement learning in our problem, we first describe the loss of the predictor.

• **Loss of Predictor.** For a given patient $\bar{\mathbf{X}}^{(p)} \in \{\mathcal{D}^+, \mathcal{D}^-, \mathcal{D}'_s\}$, where $\mathcal{D}^-$ represents all the data of the control patients, the encoder first embeds each visit from Eq. (4.1). The embedded vectors are the inputs of the RNN layer. Based on Eq. (4.2), we can obtain the vector representation of each patient as $\bar{\mathbf{s}}^{(p)} = \bar{\mathbf{h}}_T^{(p)}$, which is used to calculate the probability $P(\bar{y}^{(p)})$ with Eq. (4.3). Let $|\bar{\mathcal{D}}|$ denote the number of training data,

and the loss function is defined as follows:

$$\mathcal{L}_{\mathsf{P}} = -\frac{1}{|\bar{\mathcal{D}}|}\sum_{p=1}^{|\bar{\mathcal{D}}|}\log(P(\bar{y}^{(p)})). \tag{5.7}$$

• **State**. We define the state s_i as the current data $\hat{\mathbf{X}}^{(i)}$ and all the selected data. Assume that there are m selected data, where $m \geq 0$. According to Eq. (4.1) and Eq. (4.2), we obtain the vector representations of $\hat{\mathbf{X}}^{(i)}$ (i.e., $\hat{\mathbf{s}}^{(i)}$) and the selected data. We then average all the representations learned for the selected data, and the average representation is denoted as $\hat{\mathbf{s}}_m$. If $m = 0$, then $\hat{\mathbf{s}}_m = [0, \cdots, 0] \in \mathbb{R}^g$. Let $\mathbf{e}(s_i) \in \mathbb{R}^{2g}$ represent the vector representation of the state s_i, and then $\mathbf{e}(s_i) = [\hat{\mathbf{s}}_m; \hat{\mathbf{s}}^{(i)}]$.

• **Action**. An action $a_i \in \{0, 1\}$ is to indicate whether the i-th generated data $\hat{\mathbf{X}}^{(i)}$ will be added into the dataset $\mathcal{D}'_s$. Let $\pi(s_i, a_i)$ denote the policy function, and we make use of the logistic function as the policy strategy. To sample the value of a_i, we have $\pi(s_i, a_i) = P(a_i|s_i) = a_i\sigma(\mathbf{e}(s_i); \Phi_\pi) + (1 - a_i)(1 - \sigma(\mathbf{e}(s_i); \Phi_\pi))$, where Φ_π is the parameter.

• **Reward**. The reward function is used to measure the utility of the selected data. Let $s_{|\mathcal{D}'|}$ denote the terminal state, where $|\mathcal{D}'|$ is the number of all the input generated data. When s_i is at the terminal state, it means that RL-Selector finishes all the selection. RL-Selector only receives a delayed reward at the terminal state. For other intermediate states, the rewards are 0. Therefore, the reward function is formulated as follows:

$$r(s_i|\mathcal{D}') = \begin{cases} 0, & \text{if } i < |\mathcal{D}'| + 1; \\ \beta, & \text{if } i = |\mathcal{D}'| + 1 \text{ and } |\mathcal{D}'_s| > 0; \\ \gamma, & \text{if } i = |\mathcal{D}'| + 1 \text{ and } |\mathcal{D}'_s| = 0; \end{cases} \tag{5.8}$$

where

$$\beta = \frac{1}{|\mathcal{D}'_s|}\sum_{\hat{\mathbf{X}}_s^{(j)} \in \mathcal{D}'_s}\log(P(\hat{y}^{(j)})), \gamma = \frac{1}{|\bar{\mathcal{D}}|}\sum_{\bar{\mathbf{X}}^{(j)} \in \bar{\mathcal{D}}}\log(P(\bar{y}^{(j)})).$$

Note that when s_i is at the terminal state, but there is no selection in $\mathcal{D}'_s$ (i.e., $|\mathcal{D}'_s| = 0$), we use the average likelihood of all the training data in $\bar{\mathcal{D}}$ (i.e., γ) as the final reward.

In the designed RL-Selector, all the actions contribute to the final reward, and thus the reward is delayed, which can be handled by reinforcement learning approaches. In the following, we will introduce how to optimize the proposed RL-Selector.

• **Optimization**. The goal of the proposed RL-Selector is to maximize the expected total reward. Thus, we define the following loss function:

$$\mathcal{L}_{\mathsf{RL}} = V(s_1|\mathcal{D}') = \mathbb{E}_{s_1, a_1, s_2, \cdots, a_{|\mathcal{D}'|}, s_{|\mathcal{D}'|+1}}\sum_{i=0}^{|\mathcal{D}'|+1} r(s_i|\mathcal{D}'),$$

where $a_i \sim \pi(s_i, a_i)$ and $s_{i+1} \sim P(s_{i+1}|s_i, a_i)$. Since the state s_{i+1} is fully determined by the current state s_i and the action a_i, $P(s_{i+1}|s_i, a_i)$ is alway equal to 1. $V(\cdot)$ is the value function, and $V(s_1|\mathcal{D}')$ represents the expected total reward. In the proposed RL-Selector, there is only one non-zero terminal reward, and thus all the states have the same value function, i.e., $v_i = V(s_i|\mathcal{D}') = r(s_{|\mathcal{D}'|+1}|\mathcal{D}')$. According to the policy gradient theorem [24] and the REINFORCE algorithm [26], we update the current policy with the following gradient:

$$\Phi_\pi \leftarrow \Phi_\pi + \eta'\sum_{i=1}^{|\mathcal{D}'|} v_i \nabla_{\Phi_\pi}\log\pi(s_i, a_i). \tag{5.9}$$

6 Experiments

6.1 Experimental Setup The datasets used in our experiments are extracted from a real healthcare database, and we identify three rare diseases: Quadriplegia (QUAD), Spastic Quadriplegia Cerebral Palsy (SCP) and Diplegic Cerebral Palsy (DCP). For each rare disease, according to the medical diagnosis guidelines, we first identify a set of optional case patients, and then domain experts help us confirm whether the patients suffer these rare diseases. Finally, we select a set of matched control patients according to patient demographical information, such as age, gender, and location. For each patient in the case group, the diagnosed date is recorded, and then we use at most 150 visits before the recorded date as the input data. For each patient in the control group, we use at most the recent 150 visits as its input. The ICD-9 codes which appear less than 5 times are removed in the datasets, and we exclude patients who made less than 5 visits. The statistics of these three datasets are shown in Table 1.

Table 1: Statistics of Datasets.

Dataset	Quadriplegia	SCP	DCP
# of cases	514	424	273
# of controls	19,440	17,703	11,833
# of visits	534,663	375,927	243,066
Avg. # of visits per patient	26.85	20.74	20.08
# of unique ICD-9 codes	9,178	8,357	7,461
Avg. # of codes per visit	2.41	2.16	2.13

Baselines. For comparing with the proposed system, we select several baselines: basic, weighted sampling-based, weighted loss-based approaches, the state-of-the-art EHR data augmentation approaches, and the proposed MaskEHR.

• *Basic approaches.* We use Logistic Regression (LR), SVM, Random Forest (RF) and RNN (GRU) as baselines.

For LR, we apply l_2 regularization. In SVM, we use linear support vector machine with l_2 regularization. For RF, we use early stopping with at most 50 trees. The architecture of RNN is the same as that of the predictor in the proposed system.

• *Weighted sampling-based approaches* (WS). We use weighted sampling technique to force the datasets balanced, i.e., the number of case patients is equal to that of control patients. We test all the four basic approaches with the repeated datasets.

• *Weighted loss-based approaches* (WL). A coefficient is multiplied with the loss of case patients, and the coefficient is equal to the value of the number of control patients over the number of case patients. Also, we test all the four basic baselines with the weighted loss technique.

• *Data Augmentation approaches* (MedGAN [9]). We first run MedGAN[2] to generate fake data (whose labels are all 1) according to the real case patient data. We then combine the fake data and the real training data to train the four basic baselines, and finally validate them on the testing dataset.

• In addition, we use the proposed MaskEHR as a baseline. MaskEHR is first to generate EHR data, and the labels of all the generated data are 1, i.e., case patients. An RNN is used to train the model based on the generate EHR data $\mathcal{D}'$ and the input data ($\mathcal{D}^+$ and $\mathcal{D}^-$). In the testing procedure, only the RNN is used to make prediction on the testing dataset.

• MaskEHR+RL denotes the proposed system. In the testing procedure, only the rare disease predictor is used to predict the label of each input test data.

Implementation Details. We implement all the deep learning baselines and the proposed model with PyTorch 1.2.0. In the experiments, we set $v = g = q = 128$, and the dropout rate is 0.5. We also use l_2 norm regularization with the coefficient 0.001. For training models, we use Adadelta [29] with a mini-batch size of 50. We use 5-fold cross validation technique to randomly divide the datasets into the training and testing set in a 0.8:0.2 ratio. The training epochs is set as 20. We report the **average F_1 values** obtained with the trained parameters from the last epoch on the 5 testing sets.

6.2 Performance Table 2 shows the experimental results for different approaches on the three datasets. We can observe that the proposed MaskEHR+RL achieves the best performance. On the Quadriplegia dataset, the basic RNN obtains the lowest F_1 value because without sufficient case patient data, it is hard for RNN to learn optimal parameters, and thus it cannot make correct predictions. However, its performance dramatically increases when we use weighted sampling, weighted loss and data augmentation techniques. These approaches either increase the number of case patients or assign large weights to the case patient data, which makes RNN achieve comparable performance as other baselines.

Table 2: Performance on the Three Datasets.

Model		Dataset		
		QUAD	SCP	DCP
Basic Classifier	LR	0.4620	0.6230	0.4621
	SVM	0.4411	0.6379	0.4524
	RF	0.4032	0.6591	0.3433
	RNN	0.2815	0.5494	0.4197
Weighted Sampling	WS+LR	0.4870	0.6632	0.4623
	WS+SVM	0.4171	0.6263	0.4117
	WS+RF	0.4651	0.6844	0.4122
	WS+RNN	0.4800	0.6589	0.4597
Weighted Loss	WL+LR	0.4823	0.6734	0.4593
	WL+SVM	0.4487	0.6419	0.4551
	WL+RF	0.2961	0.5498	0.2967
	WL+RNN	0.4774	0.6738	0.4877
Data Augmentation	MedGAN+LR	0.4482	0.6095	0.4620
	MedGAN+SVM	0.4440	0.6354	0.4485
	MedGAN+RF	0.2113	0.5456	0.3541
	MedGAN+RNN	0.3983	0.6258	0.4671
The Proposed	MaskEHR	0.4383	0.6871	0.4654
	MaskEHR+RL	**0.5019**	**0.7054**	**0.5047**

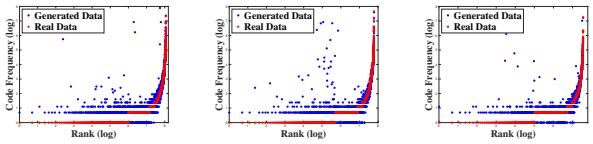
Compared with basic methods, the performance of weighted sampling-based approaches increases, except SVM. This is because repeating training samples of the case group may make SVM hard to identify the classification boundary. When weighted loss technique is used, only Random Forest drops its performance. The reason is that with such a technique, Random Forest will only focus on those patients with very clear characteristics. In this case, Random Forest can obtain a very high precision, but an extremely low recall. Thus, the overall performance, i.e., F_1 value, is not satisfactory. For data augmentation approaches, using MedGAN to generate fake data can only improve the performance of RNN. However, they are still much lower than the performance of the proposed MaskEHR+RL.

The performance of the proposed MaskEHR is not better than that of most baselines, which is reasonable because the generated EHR data contains a lot of noise. This is the common drawback of existing EHR data generation approaches [3, 9]. However, the performance of the proposed MaskEHR is higher or comparable compared with MedGAN+RNN. This observation confirms that the quality of generations produced by the proposed MaskEHR is greater than that of MedGAN, due to considering the temporal characteristic of EHR data. To remove the noisy data and further improve the perfor-

[2] https://github.com/mp2893/medgan

mance, we designed the RL-Selector. From the result of the proposed MaskEHR+RL, we can observe that the F_1 value of MaskEHR+RL is significantly greater than that of MaskEHR. Moreover, using reinforcement learning makes the proposed MaskEHR+RL achieves the greatest F_1 value compared with all the baselines. Similar results can be observed on both Spastic Cerebral Palsy (SCP) and Diplegic Cerebral Palsy (DCP) datasets.

6.3 MaskEHR Analysis The benefit of the proposed rare disease prediction system is that it can automatically generate EHR data based on the designed MaskEHR model. To validate the difference between the generated data and the real data, we conduct the following experiment. We first gather all the diagnosis codes in both the generated data and real data, then separately count the frequency for each diagnosis code in both data and rank the codes according to the frequency of codes in the real patient data, and finally plot Figure 4. X-axis represents the log frequency rank of each diagnosis code from the real patient data, and Y-axis is the corresponding log frequency. Each dot in Figure 4 denotes a diagnosis code.



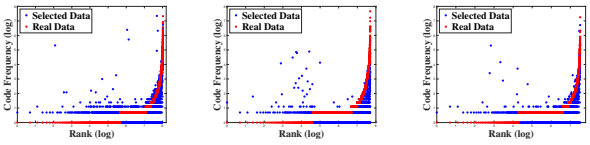
(a) QUAD (b) SCP (c) DCP

Figure 4: Rank v.s. Frequency on the Three Datasets for Analyzing MaskEHR.

From Figure 4, we can observe that for all the datasets, the proposed MaskEHR model can generate "new" diagnosis codes, i.e., the dots on the Y-axis when $Rank = 0$. At the same time, the proposed MaskEHR can discard a part of diagnosis codes when generating EHR data. Those codes are on the X-axis in blue color. Finally, MaskEHR can change the frequency or distribution of diagnosis codes. Some codes increase their frequency, but some reduce the number of occurrences.

Though these analysis cannot directly demonstrate that the generated data are meaningful in healthcare, we at least provide a feasible solution to produce more EHR data for those rare diseases. From Table 2, we also can observe that with the generated EHR data, the performance of MaskEHR is better than that of the basic RNN. This illustrates that the generated data are useful for the prediction. However, there exists noisy information among these data, and we need to remove the "bad" generated data to further improve the prediction performance. Next, we will show the importance of the proposed RL-Selector.

6.4 RL-Selector Analysis RL-Selector aims at removing the noisy generated EHR data and keeping the high-quality data to train a satisfactory prediction model. To analyze the selected EHR data, we conduct similar experiments as in MaskEHR analysis.



(a) QUAD (b) SCP (c) DCP

Figure 5: Rank v.s. Frequency on the Three Datasets for Analyzing RL-Selector.

We compare diagnosis code frequency from the selected data with the real data in Figure 5. We can observe that the frequency of diagnosis codes on these two kinds of data is different. Compared Figures 4 and 5, there is an interesting phenomenon, that is, the diagnosis codes with high frequency in the generated EHR data may have great probabilities to be selected by the proposed RL-Selector. This phenomenon shows that the generated data indeed contain a lot of useful information for the final prediction, but it needs a selector to pick important data out. Thus, the proposed RL-Selector is essential for rare disease prediction task.

7 Conclusions

In this paper, we design an effective, novel and end-to-end system that can assist doctors in the diagnosis of patients with rare diseases. The proposed system consists of three important components: data generation, data selection and prediction. These components are tightly coupled to achieve superior performance for rare disease prediction. Specifically, MaskEHR automatically generates "fake" EHR data based on the visits from case patients that simulate real data as much as possible. RL-Selector helps to guarantee the quality of data that enters the training set, which in turn leads to an accurate predictor. We conduct experiments on three real medical datasets to validate the effectiveness and reasonableness of the proposed system.

8 Acknowledgement

Research reported in this publication was supported in part by the US National Science Foundation under Grant Number IIS 1553411 and IIS 1747614. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] T. Bai, S. Zhang, B. L. Egleston, and S. Vucetic, *Interpretable representation learning for healthcare via capturing disease progression through time*, in KDD, 2018, pp. 43–51.

[2] I. M. Baytas, C. Xiao, X. Zhang, F. Wang, A. K. Jain, and J. Zhou, *Patient subtyping via time-aware lstm networks*, in KDD, 2017, pp. 65–74.

[3] Z. Che, Y. Cheng, S. Zhai, Z. Sun, and Y. Liu, *Boosting deep learning risk prediction with generative adversarial networks for electronic health records*, in ICDM, 2017, pp. 787–792.

[4] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, *Infogan: Interpretable representation learning by information maximizing generative adversarial nets*, in NIPS, 2016, pp. 2172–2180.

[5] Y. Cheng, F. Wang, P. Zhang, and J. Hu, *Risk prediction with electronic health records: A deep learning approach*, in SDM, 2016, pp. 432–440.

[6] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, *On the properties of neural machine translation: Encoder-decoder approaches*, arXiv, (2014).

[7] E. Choi, M. T. Bahadori, L. Song, W. F. Stewart, and J. Sun, *Gram: Graph-based attention model for healthcare representation learning*, in KDD, 2017, pp. 787–795.

[8] E. Choi, M. T. Bahadori, J. Sun, J. Kulas, A. Schuetz, and W. Stewart, *Retain: An interpretable predictive model for healthcare using reverse time attention mechanism*, in NIPS, 2016, pp. 3504–3512.

[9] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, *Generating multi-label discrete patient records using generative adversarial networks*, in MLHC, 2017, pp. 286–305.

[10] J. Dai, M. Zhang, G. Chen, J. Fan, K. Y. Ngiam, and B. C. Ooi, *Fine-grained concept linking using neural networks in healthcare*, in SIGMOD, 2018, pp. 51–66.

[11] W. Fedus, I. Goodfellow, and A. M. Dai, *Maskgan: Better text generation via filling in the _*, in ICLR, 2018.

[12] Y. Ganin and V. Lempitsky, *Unsupervised domain adaptation by backpropagation*, arXiv, (2014).

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial nets*, in NIPS, 2014, pp. 2672–2680.

[14] W. Jiang, C. Miao, F. Ma, S. Yao, Y. Wang, Y. Yuan, H. Xue, C. Song, X. Ma, D. Koutsonikolas, W. Xu, and L. Su, *Towards environment independent device free human activity recognition*, in MobiCom, 2018, pp. 289–304.

[15] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzell, *Learning to diagnose with lstm recurrent neural networks*, in Proceedings of ICLR, 2015.

[16] F. Ma, R. Chitta, J. Zhou, Q. You, T. Sun, and J. Gao, *Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks*, in KDD, 2017, pp. 1903–1911.

[17] F. Ma, J. Gao, Q. Suo, Q. You, J. Zhou, and A. Zhang, *Risk prediction on electronic health records with prior medical knowledge*, in KDD, 2018, pp. 1910–1919.

[18] F. Ma, Y. Wang, H. Xiao, Y. Yuan, R. Chitta, J. Zhou, and J. Gao, *A general framework for diagnosis prediction via incorporating medical code descriptions*, in BIBM, 2018, pp. 1070–1075.

[19] F. Ma, Q. You, H. Xiao, R. Chitta, J. Zhou, and J. Gao, *Kame: Knowledge-based attention model for diagnosis prediction in healthcare*, in CIKM, 2018, pp. 743–752.

[20] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, *Recurrent neural network based language model*, in INTERSPEECH, 2010, pp. 1045–1048.

[21] R. Miotto, F. Wang, S. Wang, X. Jiang, and J. T. Dudley, *Deep learning for healthcare: Review, opportunities and challenges*, Briefings in Bioinformatics, (2017), p. bbx044.

[22] T. Pham, T. Tran, D. Phung, and S. Venkatesh, *Deepcare: A deep dynamic memory model for predictive medicine*, in PAKDD, 2016, pp. 30–41.

[23] Q. Suo, F. Ma, G. Canino, J. Gao, A. Zhang, P. Veltri, and A. Gnasso, *A multi-task framework for monitoring health conditions via attention-based recurrent neural networks*, in AMIA, 2017.

[24] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, *Policy gradient methods for reinforcement learning with function approximation*, in NIPS, 2000, pp. 1057–1063.

[25] Y. Wang, F. Ma, Z. Jin, Y. Yuan, G. Xun, K. Jha, L. Su, and J. Gao, *Eann: Event adversarial neural networks for multi-modal fake news detection*, in KDD, 2018, pp. 849–857.

[26] R. J. Williams, *Simple statistical gradient-following algorithms for connectionist reinforcement learning*, Machine Learning, 8 (1992), pp. 229–256.

[27] Y. Xu, S. Biswal, S. R. Deshpande, K. O. Maher, and J. Sun, *Raim: Recurrent attentive and intensive model of multimodal patient monitoring data*, in KDD, ACM, 2018, pp. 2565–2573.

[28] L. Yu, W. Zhang, J. Wang, and Y. Yu, *Seqgan: Sequence generative adversarial nets with policy gradient*, in AAAI, 2017, pp. 2852–2858.

[29] M. D. Zeiler, *Adadelta: an adaptive learning rate method*, arXiv, (2012).

[30] X. Zhang, B. Qian, X. Li, J. Wei, Y. Zheng, L. Song, and Q. Zheng, *An interpretable fast model for predicting the risk of heart failure*, in SDM, SIAM, 2019, pp. 576–584.

[31] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, *Unpaired image-to-image translation using cycle-consistent adversarial networks*, arXiv, (2017).