Adversarial Multi-Residual-Header LGGNet: Domain Adaptative Reinforcement Causal Learning for Label Graph Generation

Anonymous submission

Abstract

The International Classification of Diseases (ICD) automatic coding is frequently regarded as a multi-label prediction problem requiring methods for appropriately assigning one or more codes for a given discharge summary. Furthermore, existing automatic ICD coding methods struggle to accurately classify medical diagnosis texts that encode deep sparse categories when their parameters are updated using simple back-propagation methods. To address the abovementioned issues, we propose LGGNet, a new adversarial network that converts automatic ICD coding into a labeled graph generation problem. This strategy faces three significant challenges: (1) How can the relationship between clinical text and ICD encoding and the multi-integration relationship between ICD encoding be extracted? (2) What training methods should be used to improve the label graph generation network's generalization, robustness, and effectiveness? (3) How can the quality of labeled graphs generated by LGGNet be assessed? For (1), we develop an information integration module (MIM) to encode the relationship between clinical text and ICD codes and capture the hierarchical and co-occurring semantics in ICD codes. For (2), we present adversarial domain adaptation training algorithms, such as reinforcement causal learning and adversarial perturbation regularization fine-tuning. For (3), We propose a label graph discriminator (LG) that estimates the reward of each ICD code in the generator label graph. In summary, we use an adversarial reinforcement architecture to train LGGNet in an adversarial domain adaptation training approach. The experiments were carried out on the well-known open dataset MIMIC-III, and the results show that LGGNet outperforms current state-of-the-art methods on core evaluation metrics such as micro-F1, micro-AUC, and P@K, resulting in the creation of a new SOTA.The code is available at anonymous github link: https://anonymous.4open.science/r/LGGNet-BB17/.

Introduction

A clinical text contains various patient information, including admissions, clinical notes, medical history, and lab test results (Nadathur 2010; Miotto et al. 2018). ICD coding is usually considered a multi-label prediction task, requiring multiple proper ICD codes to a given clinical text. Figure 1 shows the hierarchical structure of the ICD codes, in which siblings are unlikely to occur in one clinical text simultaneously. In addition, as the patients' visits increase, the ICD codes increase, which attracts researchers' attention

ICD codes	Code Description
783	Symptoms concerning nutrition, metabolism, and development
783.0	Anorexia
783.1	Abnormal weight gain
783.2	Abnormal loss of weight and underweight
.....	.....

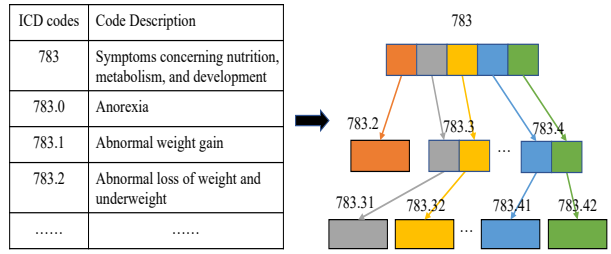

Figure 1: Hierarchical diagram of the ICD-9 codes.

to automatic ICD coding because manual coding is time-consuming and error-prone (Huang, Osorio, and Sy 2019).

Most neural network methods handle automated coding as a multi-label prediction task (Mullenbach et al. 2018a; Cao et al. 2020; Xie et al. 2019). In terms of ICD codes, there are some special characteristics. First, of all the 9219 codes, the number of the corresponding codes related to the most frequent top50 is 122, which means the distribution of codes is extremely unbalanced, and most codes are inactive in clinical texts. Second, most previous methods neglect or underestimate the relations between ICD codes, such as parent-child relations, sibling relations, and mutually exclusive relations (Mullenbach et al. 2018b; Xie and Xing 2018). For example, in Figure 1, "783.1" (ICD code of Abnormal weight gain) is mutually exclusive with "783.2" (ICD code of Abnormal loss of weight and underweight). Third, current methods only focus on one training way to update their parameters (Wang et al. 2020b; Vu, Nguyen, and Nguyen 2020; Ji, Cambria, and Marttinen 2020), which may cause failure in some clinical texts containing rare diseases.

To address the above issues, we propose reformulating automatic ICD coding as a labeled graph generation task along the ICD code graph. This new formulation generates graph labels (nodes or codes) one by one. Intuitively, firstly, we need to build the global code graph with ground labels of training clinical texts. Every two labels that occur in one clinical text have an edge in the global code graph. Next, we generate the first graph label with root and embed the input clinical text. Then, we utilize the neighbors of the first generated label to predict the second one, which reduces the candidate space of ICD codes. Then use the second and the input embedding to predict the third one.

However, there are three main challenges: (1) How to extract the relationships between clinical texts and ICD codes, and multi-comprehensive relationships between ICD codes? For example, sibling relations, parent-child relations, and mutually exclusive relations. (2) How to train the label graph generation network? (3) How to evaluate the quality of generated label graphs of LGGNet? For example, how to make a judgment system to measure the label graph generation networks automatically?

In this paper, we address the issues and challenges above by proposing a multi-algorithm label graph generation framework named Label Graph Generation Network (LGGNet). LGGNet employs four training ways to generate code graph labels, and it ends the generation process while the generated labels could form a circle. As a result, LGGNet assigns the labels in the final circle as the codes to the given clinical text. LGGNet contains two main components: a Label Graph Generator (LG) and a Label Discriminator (LD). The LG consists of a text encoder (MHR-CNN) and a Message Integration Module (MIM). The text encoder can obtain an input representation when given a clinical text. Then, the MIM is introduced to model the relations between clinical text and ICD codes, as well as between ICD codes. Specifically, the LG is meant to generate graph labels indistinguishable from positive ones, while the LD is meant to distinguish positive and generated labels.

We conduct extensive experiments on the MIMIC-III benchmark dataset (Johnson et al. 2016) to obtain empirical evidence for the effectiveness of LGGNet. Our experimental results demonstrate that LGGNet significantly outperforms state-of-the-art methods by a large margin.

To sum up, the contributions of this work are as follows:

- We first formulate automatic EHRs coding as a labeled graph generation task and propose a multi-algorithm label graph generation framework LGGNet for automatic ICD coding.
- We propose a Message Integration Module (MIM) that models the relations between EHRs and ICD codes, including parent-child, sibling, and mutually exclusive relations.
- We first employ four training ways with reinforcement learning in ICD coding.
- We devise a Label Discriminator with an adversarial reward learning mechanism to evaluate the intermediate rewards as supervision signals for learning LGGNet.
- We conduct extensive experiments on a widely used dataset to verify and analyze the effectiveness of LGGNet.

Related Work

We first describe the prior works in Automated ICD coding models, then we discuss the adversarial and reinforcement learning methods for healthcare.

Automatic ICD Coding Automated ICD coding is actually a multi-label prediction task. Some CNN and attention methods are proposed to capture the key information in clinical texts, including (Mullenbach et al. 2018a; Allamanis, Peng, and Sutton 2016; Yin and Schütze 2017; dos Santos et al. 2016). Tree-of-sequences LSTM (Xie and Xing 2018) is proposed to encode the code tree hierarchy information.

Adversarial Networks for Healthcare Generative Adversarial Networks (GANs) and their successors have achieved great success in many domains, such as computer vision and dialogue generation. The training process is formalized as a game in which the generative module is trained to generate outputs to fool the discriminator. The goal of the discriminator is to distinguish the output of the generator and the ground truth. In healthcare, several GAN-based models have been proposed to handle tabular data in EHRs. RGAN and RCGAN (Esteban, Hyland, and Rätsch 2017) can generate real-valued time-series data. (Choi et al. 2017; Guan et al. 2018) have proposed medGAN and mtGAN, respectively, to generate realistic synthetic patient records. (Yu et al. 2019) uses GANs in a semi-supervised learning setting to improve the detection performance of rare diseases.

Method and Theoretical Analysis

Given an clinical text $X = \{x_1, x_2, ..., x_N\}$, where x_i represents the *i-th* token. The task of automatic ICD coding with label graph generation is to generate labels $Y = \{y^1, y^2, ..., y^m\}$, where y^i indicates the *i-th* label (code) in the generated graph. Note that all the graphs start with the root node and end while a label circle is formed. Next, we describe LGGNet in detail.

Overview of LGGNet

LGGNet contains two major components: a label graph generator G_θ and a label graph discriminator D_ζ, which is shown in Figure 2.

Given a clinical text, the label graph generator G_θ generates ICD graph labels, parameterized by θ, where θ represents parameters of the network. G_θ consists of two modules: a text encoder and a Message Integration Module (MIM). The text encoder is used to initial the input text and obtain its representations by CNN; MIM aims to obtain information between clinical texts and ICD codes, as well as relations between ICD codes. The information obtained from the MIM determines the initial state s_i of the *i-th* label to predict. A hybrid policy network (π_θ) is designed to generate graph label y_i until there exists a label circle. The generated label y_i is then integrated with the random ground truth and fed into the label discriminator D_ζ until the D_ζ cannot distinguish the generated label.

The path discriminator D_ζ, parameterized by ζ, is a binary classifier that takes the random groud truth *t* and generated y_i as inputs and outputs a probability r_i, indicating whether the current label is generated or ground truth. The initial representations of ground truth and generated labels are obtained by averaging the semantic of their descriptions, which is shown in Fig. 1. r_i is considered as a reward to guide the learning of label graph generator G_θ.

Label Graph Generator G_θ

We model the above overview's process as a Markov Decision Process (MDP) (Bennett and Hauser 2013) $<$

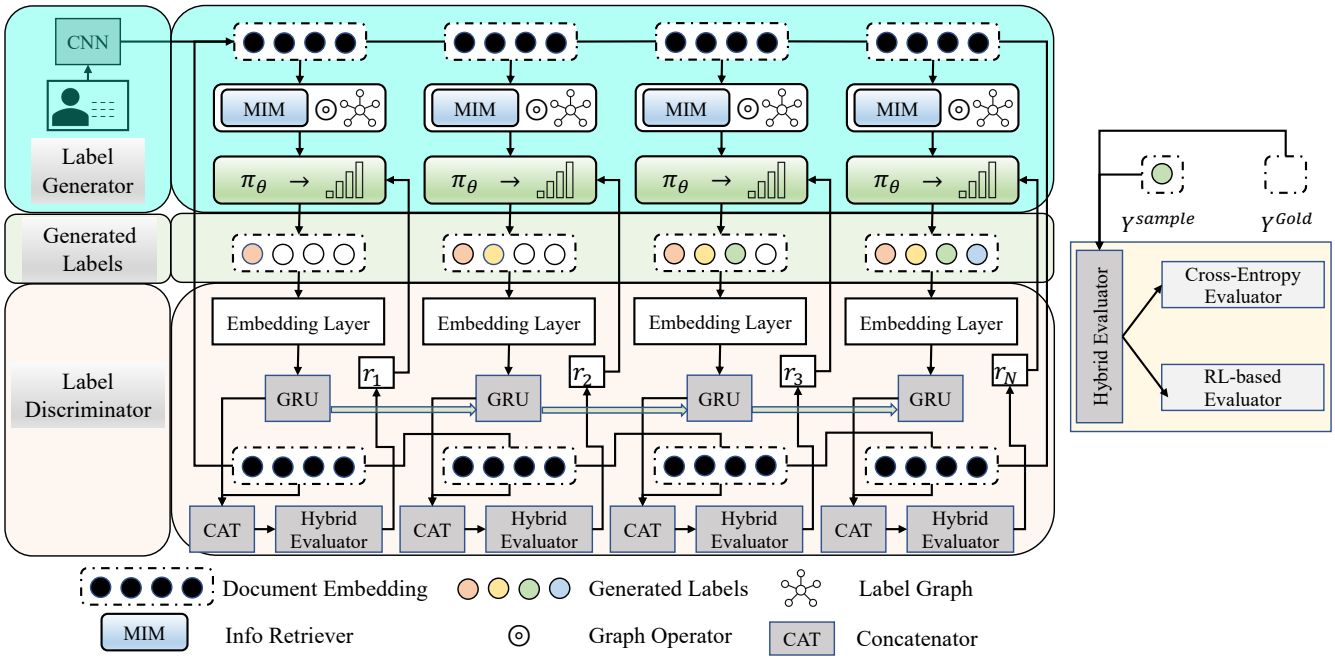

Figure 2: The overall framework of LGGNet. LGGNet contains two main parts: Label Generator G_θ and Label Discriminator D_ζ. MIM is an info retriever and EEM is a Multi-Residual-Header CNN Attentional Layer that will be described in the following Sections in detail.

$\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} >$, where $\mathcal{S}$ represents state space, $\mathcal{A}$ is the set of all available actions. For example, the subset of $\mathcal{A}$ corresponding to a label is its neighbours in the global graph. $\mathcal{T}$ is the state transition function (from current state to the next state), $\mathcal{R}$ is the reward function of each pair (state, action). In order to inspire G_θ to generate ground truth-like labels, we propose to maximize the expected rewards of G_θ using the reinforce algorithm (Williams 1992). For a trajectory τ=$\{s_1, a_1, s_2, a_2, ..., s_T, a_T\}$ where a means an action, we can get its expected reward as Equation 1. The average expected reward of trajectories can be obtained by $\bar{R}(\theta)$

$$\begin{aligned} R(\theta) &= E_{\tau \sim P_\theta(\tau)}[R(\tau)] \\ &= E_{a \sim \pi(a)} \\ \bar{R}(\theta) &= E_{a \sim \pi(a|S=s, X=x;\theta)}[\Sigma_i R(s=s_i, X=x, a_i)] \\ &= \Sigma_t \Sigma_{a_i \in \mathcal{A}} \pi(a_i|s=s_i, X=x;\theta) R(s=s_i, X=x, a_i) \end{aligned} \tag{1}$$

where $R(\theta)$ is the expected reward of one trajectory; $\bar{R}(\theta)$ is the expected total rewards for one episode; τ means trajectory; $\pi(a_i|s=s_i, X=x;\theta)$ is the hybrid policy network of the path generator G_θ; a_i is the generated label based on current state s_i and x; $R(s=s_i, X=x, a_i)]$ is the reward of generated a_i based on s_i and x. We can implement a_i in the module D_ζ. θ can be updated by the policy gradient as follows ($R(s=s_i, X=x, a_i)$ is irrelevant with θ):

$$\begin{aligned} \bigtriangledown \bar{R}(\theta) = \sum_t \sum_{a_i \in \mathcal{A}} & \pi(a_i|s=s_i, X=x;\theta) \\ & \bigtriangledown \log \pi(a_i|s=s_i, X=x;\theta) \end{aligned} \tag{2}$$

We model $\pi(a_i|s=s_i, X=x;\theta)$ as Equation 3:

$$\pi(a_i|s=s_i, X=x;\theta) = \sigma(W(s_i) + b_i) \tag{3}$$

where W is a matrix and b is a bias; σ is the sigmoid activation function.

Label Graph Discriminator D_ζ

We design the path discriminator module D_ζ to get the reward m_i for each code in the generated path (c_1, c_2,...,c_i) until timestamp i. Specifically, we model h_i as the discrimination probability as follows:

$$\begin{aligned} h_i &= R_{(s=s_i, X=x, a=a_i)} \\ &= p_s((c_1, c_2, c_3, ..., c_i), x) \\ &= \sigma(M_h(LSTM(h_{k-1}, c_k) \oplus x)) \end{aligned} \tag{4}$$

where $\oplus$ denotes the concatenation operation, and M_h is the weight matrix; c_i is the representation of the current generated path, which is obtained by recurrently applying an LSTM to the ICD code path.

To learn the path discriminator D_ζ, we adapt an adversarial-based domain adaptative training method, where we consider the paths generated from the path generator G_θ as negative samples and the ground truth paths as positive samples. The objective of path discriminator D_ζ can be formulated as minimizing the following cross-entropy function:

$$Loss_s = -\sum_{(y_i,x)\in S^+} \log p_s(y_i, x) - \sum_{(y_i,x)\in S^-} \log(1 - p_s(y_i, x)) \quad (5)$$

where S^+ and S^- denote positive and negative samples respectively; $p_s(y_t, x)$ represents the probability of sample (y_i, x) belonging to a positive sample. The update of path discriminator D_ζ is identical to the common binary classification problem, which can be optimized by any gradient-based algorithm.

Message Integration Module

Inspired by (Wang et al. 2020a; Lanchantin, Sekhon, and Qi 2019; Srivastava, Greff, and Schmidhuber 2015), we propose a MIM to encode the state e_i by taking into account the relation between a clinical text and hierarchical ICD structure, sibling relations, and parent-child relations of ICD codes, as shown in Figure 2. Formally, s_i is defined as:

$$e_i = m_i \odot T(x, W_T) + x \odot (1 - T(x, W_T)) \quad (6)$$

where W_T is the weight matrix; T is a gate that is used to control the information transforming from the EHRs representation x and m_i, respectively; m_i represents the relation representation obtained by three steps of message passing: EHRs-to-EHRs, parent-to-child message passing, and sibling-to-sibling message passing, as shown in Figure 2. Next, we introduce the modeling of these three steps one by one.

EHRs-to-Path Message Release This step is used to integrate the EHRs representation and ICD path representation. The relation representation between an EHRs and an ICD path is denoted as g_i, which can be obtained as follows:

$$g_t = \tanh(W_g(x \oplus c_i \oplus (x \odot c_i) \oplus (x + c_i) \oplus (x - c_i) \oplus (c_i - x))) \quad (7)$$

where W_g is the weight matrix and the parameter in W_g is obtained through various transformations of EHRs representation x and path representation c_i; $\oplus$ indicates vector concatenation, $\odot$ indicates the elementwise product, and c_i is the representation of selected ICD code at timestamp i.

Parent-to-Child Message Release This step is used to capture the relation between parent and child ICD codes of ICD code a_i. After obtaining relation representation o_i between an EHRs and an ICD path, we propagate this relation representation from parent code to all its child codes to generate relation representation v_i:

$$u_i = e_i \odot c_i^c \quad (8)$$

where $\odot$ indicates the element-wise product operation, and c_i^c is the vector representation of each child ICD.

Sibling-to-Sibling Message Release This step is used to capture the relation between sibling ICD codes by spreading information between sibling ICD codes. Specifically, we achieve this by passing information between sibling ICD codes using intraattention (also called self-attention [34]) neural message passing, which enables ICD codes to attend over their sibling ICD codes differently. This allows for the network to learn different degrees of importance for different sibling ICD codes. Sibling-to-sibling message passing is formulated as follows:

$$m_i = u_i + \sum_{n\in A_{ui}} F_{atn}(u_i, u_i^n) \quad (9)$$

where F_{atn} is the attention function; A_{ui} indicates all the sibling ICD codes of u_i, and u_i^n represents the n-th sibling ICD of code u_i; The attention weights β_i^n for a pair of sibling codes(u_i, u_i^n) can be calculated using the attention function $f(\cdot)$. F_{atn} is used to pass the message from the n-th sibling ICD code to the current ICD code u_i using the learned attention weights α_i^n:

$$\begin{aligned} F_{atn}(u_i, u_i^n) &= \beta_i^n u_i^n \\ &= softmax_n(e_i^n) \\ &= \frac{e_i^n}{\Sigma_{k\in A_{u_i}} \exp(e_i^k)} \end{aligned} \quad (10)$$

where e_i^n represents the importance of the n-th sibling code for the current code before normalization; the e_i^n are normalized across all sibling nodes of the current ICD code using a softmax function (Eq. 11) to get β_i^n. For the attention function $f(\cdot)$, we use a dot product with linear transformations W_q on code u_i and W_u on u_i^n:

$$\begin{aligned} e_i^n &= f(u_i, u_i^n) \\ &= (W_q u_i)^T (W_u u_i^n) \end{aligned} \quad (11)$$

Multi-Header Residual CNN Embedding

In this part, we will introduce our Multi-Header Residual Embedding Layer to encode EHRs, whose architecture is shown in Figure 3.Throughout this paper, we employed the following notation rules: matrices are written as italic uppercase letters (e.g., X); vectors and scalars are written as italic lowercase letters (e.g., x).

Our model leverages a word sequence $w = \{w_1; w_2; ...; w_n\}$ as input, where n denotes the sequence length. Assuming that $\tilde{E}$ denotes the word embedding matrix, which is pretrained via word2vec (Mikolov et al. 2013) from the raw text of the dataset. A word w_n will correspond to a vector e_n by looking up $\tilde{E}$. Therefore, the input will be a matrix $E = \{e_1, e_2, \ldots, e_n\} \in \mathbb{R}^{n\times d^e}$.

Multi-Header Convolutional Filter To capture the patterns with different lengths, we leveraged the multi-filter convolutional neural network (Kim 2014), where each filter has a different kernel size (i.e., word window size). Assuming we have m filters $f_1, f_2, ..., f_m$ and their kernel sizes denote as $k_1, k_2, ..., k_m$. Therefore, m 1-dimensional convolutions can be applied to the input matrix E. The convolutional procedure can be formalized as:

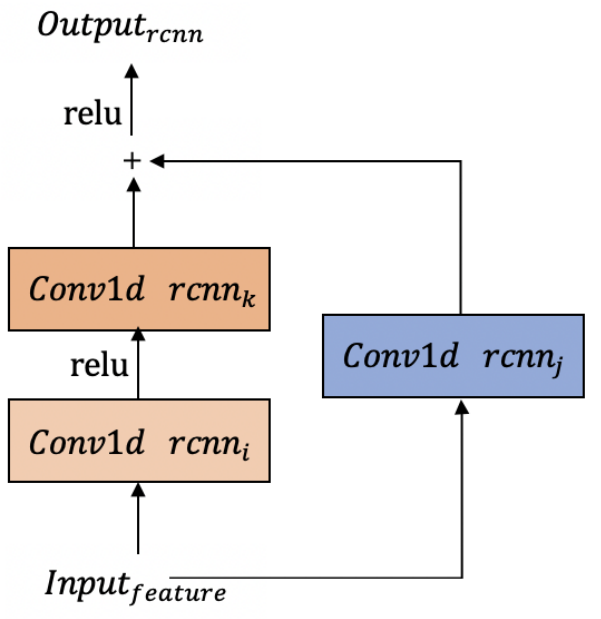

Figure 3: Resudial-CNN Block RCNN. "+" represents the element-wise addition.

$$H_1 = f_1(E) = \bigwedge_{j=1}^{n} \tanh\left(W_1^T E^{j:j+k_1-1}\right) \tag{12}$$

$$H_m = f_m(E) = \bigwedge_{j=1}^{n} \tanh\left(W_m^T E^{j:j+k_m-1}\right) \tag{13}$$

where $\bigwedge_{j=1}^{n}$ indicates the convolutional operations from left to right. Here we forced the row number n of the output H_1 or $H_m \in \mathbb{R}^{n\times d^f}$ to be the same as that of the input E, because we aimed to keep the sequence length unchanged after convolution. It is simple to implement such goal, e.g., setting the kernel size, padding and stride as k, $floor(k = 2)$ and 1. d^f indicates the out-channel size of a filter and every filter has the same output size.

Moreover, $E^{j:j+k_1-1} \in \mathbb{R}^{k_1\times d^e}$ and $E^{j:j+k_m-1} \in \mathbb{R}^{k_m\times d^e}$ indicate the sub-matrices of E, starting from the j-th row and ending at the $j+k_1-1$ or $j+k_m-1$ row. $W_1 \in \mathbb{R}^{(k_1\times d^e)\times d^f}$ and $W_m \in \mathbb{R}^{(k_m\times d^e)\times d^f}$ indicate the weight matrices of corresponding filters. Throughout this paper, the biases of all layers are ignored for conciseness. The overview of a 1-dimensional convolution filter f_m is shown in Figure todo.

Multi-Residual Convolutional Block On top of each filter in the multi-filter convolutional layer, there is a residual convolutional layer which consists of p residual blocks (He et al. 2016). Take the m-th filter as an example, the computational procedure of its corresponding residual blocks $r_{m1}, r_{m2}, ..., r_{mp}$ can be formalized as:

Algorithm 1: Computational procedure of residual blocks

Input: Input matrix H_m; Residual blocks $r_{m1}, r_{m2}, ..., r_{mp}$
Parameter: Initialized H with the number of filter m; Intermediate process result X; Number of iterations p
Output: Integration results after filtering H_{mp}

while epoch in range($EPOCHS$) **do**
  while each $iter$ = 1, 2, . . . ,p **do**
    $H_{mi} = r_{mi}(X)$
    $X = H_{mi}$
  end while
  return H_{mp}
end while

For the residual block r_{mi} (Figure todo), it consists of three convolutional filters, namely r_{m1}, r_{m2} and r_{m3}. The computational procedure can be denoted as:

$$\begin{aligned} X_1 = r_{mi_1}(X) &= \bigwedge_{j=1}^{n} \tanh\left(W_{mi_1}^T X^{j:j+k_m-1}\right), \\ X_2 = r_{mi_2}(X_1) &= \bigwedge_{j=1}^{n} W_{mi_2}^T X_1^{j:j+k_m-1}, \\ X_3 = r_{mi_3}(X) &= \bigwedge_{j=1}^{n} W_{mi_3}^T X^{j:j}, \\ H_{mi} &= \tanh\left(X_2 + X_3\right), \end{aligned} \tag{14}$$

where $\bigwedge_{j=1}^{n}$ indicates the convolutional operations. X denotes the input matrix of this residual block and $X^{j:j+k_m-1} \in \mathbb{R}^{k_m\times d^{i-1}}$ indicate the sub-matrices of X, starting from the j-th row and ending at the $j+k_m-1$ row. $H_{mi} \in \mathbb{R}^{n\times d^i}$ denotes the output matrix of the residual block. d^{i-1} and d^i denote the in-channel and out-channel sizes of this residual block. Therefore, the in-channel size of the first residual block r_{m1} should be d^f and the out-channel size of the last residual block r_{mp} is defined as d^p . Similar with the multifilter convolutional layer, we let the row numbers of H_{mi} as well as X_1, X_2 and $X_3 \in \mathbb{R}^{n\times d^i}$ be n, which is identical to that of the input X.

Moreover, $W_{mi_1} \in \mathbb{R}^{\left(k_m\times d^{i-1}\right)\times d^i}$ and $W_{mi_3} \in \mathbb{R}^{\left(1\times d^{i-1}\right)\times d^i}$ denote the weight matrices of the three convolutional filters, r_{mi_1}, r_{mi_2} and r_{mi_3}. Thereinto, r_{mi_1} and r_{mi_2} have the same kernel size k_m with the corresponding filter f_m in the multi-filter convolutional layer, but they have different in-channel sizes. r_{mi_3} is a special convolutional filter whose kernel size is 1.

Because the m-th filter f_m in the multi-filter convolutional layer corresponds to p residual blocks r_{m1}, r_{m2},...,r_{mp} in the residual convolutional layer, we employed the output $H_{mp} \in \mathbb{R}^{n\times d^p}$ of the p-th residual block r_{mp} as the output of these residual blocks. Since there are totally m filters in the multi-filter convolutional layer, the final output of the residual convolutional layer is a concatenation of the output of m residual blocks, namely $H =$

Algorithm 2: Adversarial reinforcement learning algorithm for training LGGNet

Input: dataset D; EHRs
Parameter: Initialized G_θ, D_ζ with random weights θ, ζ; Pre-trained G_θ on dataset D with teacher forcing using the ground truth paths
Output: Label Graph Generator G_θ; Label Graph Discriminator D_ζ

1: **while** epoch in range($EPOCHS$) **do**
2: **while** each j = 1, 2, . . . ,M **do**
3: **while** each EHRs x in batch EHRs **do**
4: Sample a path for EHRs x from G_θ, y from G_θ;
5: Compute rewards for each code in y with Eq.19;
6: **end while**
7: Update generator G_θ via policy gradient with Eq.2;
8: Train G_θ on batch EHRs and ground truth paths with teacher forcing;
9: **end while**
10: **while** each j = 1, 2, . . . , N **do**
11: **while** each EHRs x in batch EHRs **do**
12: Sample a path for EHRs x from G_θ;
13: Sample a ground truth path for EHRs x from D;
14: **end while**
15: Train path discriminator D_ζ with Eq.21;
16: **end while**
17: **end while**
18: **return** G_θ; D_ζ

$H_{1p} \oplus H_{2p} \dots H_{mp} \in \mathbb{R}^{n \times (m \times d^p)}$.

Adversarial Domain Adaptative Algorithm

Adversarial Reinforcement Causal Learning The alternating training process of the path generator G_θ and the path discriminator D_ζ is shown in Algorithm 2. After randomly initializing G_θ and D_ζ, we use ground truth paths to pretrain G_θ(line 1–2). The training times for G_θ and D_ζ in each epoch are M and N respectively (line 4 and line 12). For training G_θ, we generate a single path y for each EHRs x in the batch EHRs (line 6). Then we calculate the reward for each ICD code in the generated path y(line 7).We use batch EHRs and generated paths to update the generator G_θ with policy gradient (line 9). To make the exploration of G_θ more effective, we train G_θ on batch EHRs and ground truth paths(line 10). For training D_ζ, we generate a single path y for each EHRs x in the batch EHRs (line 14) and also sample a ground truth path for this EHRs (line 15). Then, we train the path discriminator D_ζ with generated paths and ground truth paths (line 17).

Adversarial Perturbations Regularization Fine-tuning Improving the generalization performance of the model is one of the goals that machine learning strives to pursue. Among them, the generalization performance of the model can be improved through data enhancement, but one of its obvious shortcomings is "specificity". Random noise may not cause significant interference to the model, so it is of limited help to improve the generalization performance.

The process of countermeasure training is to add a small disturbance to the original input to obtain adversarial samples, which can be used for training. It can be abstracted as the following model:

$$\begin{aligned} loss &= -\log p(y|x + r_a dv; \theta) \\ wherer_a dv &= -\underset{r,\|r\|<\epsilon}{argmax} \log p(y|x + r; \bar{\theta}) \\ &= \underset{r,\|r\|<\epsilon}{argmin} \log p(y|x + r; \bar{\theta}) \end{aligned} \tag{15}$$

In addition, We propose to impose an explicit regularization to effectively control the model complexity at the fine-tuning stage. Specifically, given the model $f(;\theta)$ and n data points of the target task denoted by $(x_i, y_i)_{i=1}^n$, where x_i's denote the embedding of the input sentences obtained from the first embedding layer of the language model and y_i's are the associated labels, our method essentially solves the following optimization for fine-tuning:

$$min_\theta = Loss(\theta) + \lambda_s R_s(\theta) \tag{16}$$

where $Loss(\theta)$ is the loss function defined as:

$$Loss(\theta) = \frac{1}{n} \sum_{i=1}^{n} l(f(x_i; \theta), y_i) \tag{17}$$

and l(·,·) is the loss function depending on the target task, λ_s is a tuning parameter, and $R_s(\theta)$ is the smoothness-inducing adversarial regularizer. Here we define $R_s(\theta)$ as:

$$R_s(\theta) = \frac{1}{n} \sum_{i=1}^{n} \underset{\|x_i - x_i\|_p \leq \epsilon}{max} l_s(f(\widetilde{x_i}; \theta), f(x_i; \theta)) \tag{18}$$

where ϵ is a tuning parameter. Note that for classification tasks, f (·;θ) outputs a probability simplex and l_s is chosen as the symmetrized KL-divergence.

$$ls(P, Q) = D_{KL}(P||Q) + D_{KL}(Q||P) \tag{19}$$

For regression tasks, f(·;·) outputs a scalar and l_s is chosen as the squared loss, i.e., l_s(p,q) = $(p - q)^2$.Note that the computation of $R_s(\theta)$ involves a maximization problem and can be solved efficiently by projected gradient ascent.

We remark that the proposed smoothness-inducing adversarial regularizer was first used in Miyato et al. (2018) for semi-supervised learning with p = 2, and then in Shu et al. (2018) for unsupervised domain adaptation with p = 2, and more recently in Zhang et al. (2019) for harnessing the adversarial examples in image classification with p =1. To the best of our knowledge, we are the first applying such a regularizer to fine-tuning of pre-trained language models.

The smoothness-inducing adversarial regularizer is essentially measuring the local Lipschitz continuity of f under the metric l_s. More precisely speaking, the output of f does not change much if we inject a small perturbation (l_p norm bounded by ϵ) to x_i. Therefore, by minimizing the objective

in 16, we can encourage f to be smooth within the neighborhoods of all x_i's. Such a smoothness-inducing property is particularly helpful to prevent overfitting and improve generalization on a low resource target domain for a certain task.

Note that the idea of measuring the local Lipschitz continuity is similar to the local shift sensitivity criterion in existing literature on robust statistics, which dates back to 1960's (Hampel, 1974; Huber, 2011). This criterion has been used to characterize the dependence of an estimator on the value of one of the sample points.

Experimental Setup

In this section, we conduct extensive experiments to answer the following research questions:

RQ1 How does LGGNet perform in ICD code prediction compared with the existing automated ICD coding methods?

RQ2 How can the label graph generation network be trained so that it has better generalization, robustness, and effectiveness?

RQ3 What are the influences of different model configurations?

Dataset

The validation experiments in this paper are based on the well-known and freely available MIMIC-III dataset. This dataset is a one-of-a-kind, publicly accessible electronic large database of medical diagnostic records. As the final benchmark dataset in medical text research, it contains anonymized medical diagnostic data for less than 50,000 patients collected over twelve years beginning in 2000. [17, 39].

Preprocessing This paper focuses on the mainstream EHR discharge summary text object in the all-inclusive MIMIC-III dataset, which summarizes medical diagnosis text information of inpatients. Of course, the discharge summary text data object still contains noisy data irrelevant to this paper's research task, such as physician and hospital information, so we remove this invalid information in the data preprocessing section by introducing prior expert knowledge. The following experimental data processing was performed in this paper based on the frequency of distribution of text encoding occurrences to more effectively examine the model's generalization performance and ability to classify long-tailed medical text data for the MIMIC-III dataset. We randomly divided the below two datasets into the training set, validation set, and test set in a 4:1:1 ratio after processing the original MIMIC-III data into two types of datasets.

MIMIC-III Top50 We keep only the most common 50 ICD codes and use them to filter the text data in the MIMIC-III dataset. If medical text data does not contain any of the top 50 ICD codes in the frequency of occurrence, it is excluded until all of the discharge medical abstract data has been processed.

MIMIC-III Full We kept all diagnostic ICD codes in the discharge diagnosis summary text, so we kept everything that corresponded to any one ICD code.

Metrics

For a fair and objective comparison with previous work, we used macro-averaged and micro-averaged AUC (physical area under the ROC curve) and macro-averaged and micro-averaged F1 for the underlying core metrics, where we first calculated the performance score for each label and then averaged the scores when calculating macro-averaged AUC or F1. We count the scores of each pair of clinical records and codes for independent prediction when calculating the micro-averaged AUC or F1.

Furthermore, we personalize various MIMIC datasets and use precision at K (P@K) as an evaluation metric, where precision at K (P@K) represents the proportion of correctly predicted labels among the top K predicted labels. We use precision at level 5 (P@5) for the relevant experiments built on the MIMIC-III Top50 dataset and precision at level 8 (P@8) for the correlation experiments built on the MIMIC-III Full dataset.

Implementation Details

Because the LGGNet model proposed in this paper has a large number of hyperparameters, the time and computational cost to find the optimal values for all hyperparameters for all models are prohibitively expensive. As a result, expert bias is introduced into some fundamental hyperparameter values in the LGGNet model, which is a reasonable choice based on the authors' prior knowledge of tuning parameters.

This paper uses a fully connected layer containing 300 hidden neural units in the local and global strategies for adversarial reinforcement causal learning (Equation 3). In the EHRs encoder module, wi (Equation 14) is a 100-dimensional vector, and the parameters in this 100-dimensional vector are randomly initialized.We set the hidden side of the transformation gate T to 300 in this paper's message integration module (Equation 5) and the hidden sizes in the transformation gates W_q and W_u to 300 and 500, respectively (Equation 13).

We use the mainstream Adam optimizer [38] to optimize all parameter iterations and set the momentum parameters to the standard $beta1$ = 0.9 and $beta2$ = 0.999. Furthermore, in this paper, we set the learning rate $alpha$ to a lower learning rate of 0.00001, in conjunction with a larger number of iterations $epoch$ = 3000, to satisfy the need for full model learning. Given the GPU memory limitations, the number of training iterations M and N for both the path generator $Gtheta$ and the path discriminator $Dzeta$ is limited to 1 during model learning for each epoch (Algorithm 1), and the batch size is limited to 64 in this paper.

Finally, we used PyTorch to implement LGGNet and ran joint training on eight 32GB NVIDIA-V100 GPUs.

Baselines

To demonstrate LGGNet's efficacy, we compare it to a variety of mainstream modeling schemes, including the current best EHRs coding model and its corresponding hierarchical text classification algorithm, on a variety of core metrics, and our baseline model and its related brief are as follows.

ISD The proposed scheme uses an interactive shared representation network to address the long-tail issue and proposes a symbiotic model to establish connections between codes while building. The authors also create a self-distillation learning mechanism to help the model extract more helpful information from clinical notes while minimizing the interference from noisy data.

MSMN A neural network based on various synonym matching methods is proposed to aid in ICD code classification by applying synonym matching algorithms and data augmentation to increase the model's learning ability to perform code embedding characterization.

FUSION The redundant and sparse disease diagnosis vocabulary is compressed into information-dense local representations. The relationship modeling between local features is then realized using an attention mechanism based on an essential query, and the generation of global features is completed. Furthermore, the ICD code prediction task is completed with the previously generated global features.

CAML & DR-CAML CAML employs convolutional attention networks to learn the embedding representations of each ICD code, followed by Text-CNN [20] to obtain the embedding representations of each EHR before performing binary classification on top of both, yielding state-of-the-art results on the MIMIC dataset on time. Normalization of descriptions CAML (DR-CAML) improves CAML by normalizing the model with the help of EHRs for each ICD code by using regularization terms in the classification weights of the loss function.

BI-GRU & HA-GRU BI-GRU employs a two-way gate-based recursive unit to generate a highly informative embedded representation of EHRs and then binary classify ICD codes based on the embedded representation of EHRs. HA-GRU extends BI-GRU by applying BI-hierarchical GRU's attention to the two-way gate-based recursive unit and identifying the most relevant text in EHRs for each ICD code in the EHRs to achieve the encoding output of the EHRs and, ultimately, improve disease classification.

LAAT & JointLAAT The Label Attention Model (LAAT) proposes learning the attention distribution of each ICD code encoding hidden states on the LSTM, which is used to classify ICD encodings. On top of LAAT, JointLAAT proposes a hierarchical joint learning algorithm, which improves the effectiveness and efficiency of ICD encoding classification.

Result and Analysis

Dataset Analysis

Figure 4 and 5 depict the data statistics and visualization of the spatial data distribution of two datasets, MIMIC-III Top50 and MIMIC-III Full, using different representations. The statistical results presented in the two figures allow us to draw the following conclusions:

Sparse EHR's ICD Code The average number of ICD codes for each EHR is statistically analyzed in paper. Using the fourth layer as an example, in the MIMIC-III Top50 dataset, the average number of ICD codes of each EHR is 1.69, accounting for only 3.4 percent of the total number of ICD codes and 1.4 percent of the total number of ICD codes in this layer. In contrast, the MIMIC-III Full has even more sparse ICD code data, with an average of 5.36 ICD codes per EHR, accounting for only 0.8 of the total number of ICD codes in that layer and 0.6 of the total number of ICD codes. This demonstrates that the average number of ICD codes for a single EHR is relatively sparse compared to the total number of ICD codes in the hierarchy and the total number of ICD codes, and the data is highly sparse. We discovered that when predicting the generation path of multiple ICD codes for a single EHR, it is critical to design the MIM to integrate the interaction of ICD codes between different levels and the same level in terms of noise reduction.

Uneven Hierarchical Distribution The number of ICD codes in the fourth level alone accounts for 40.1 percent of the MIMIC-III Top50 dataset, while the number of all codes from the first to the third level accounts for only 24.9 percent of the total in the MIMIC-III Full dataset. This demonstrates that the number of ICD codes at different levels varies greatly, with shallow levels having far fewer than deep levels. As a result, searching for ICD code generation paths from low to high levels using the ICD tree hierarchy can effectively reduce the search space of ICD codes, accelerate inference and prediction speed, and improve the effectiveness of LGGNet model learning.

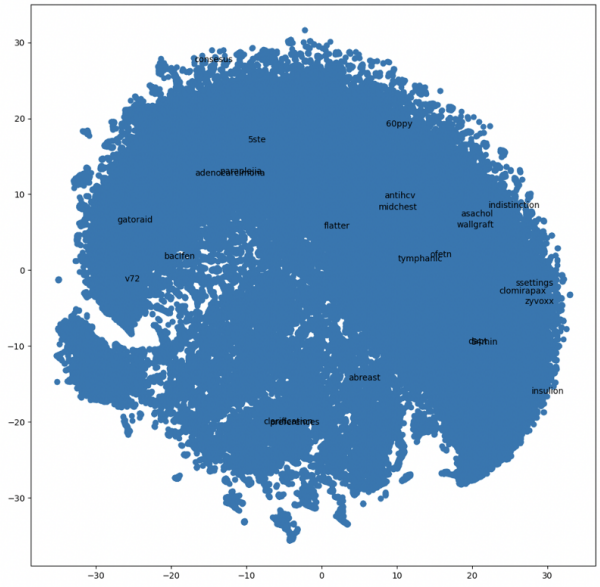

Figure 4: MIMIC-III Full Description in Scatter Graph.

Compared with Baselines (RQ1)

To answer the questions posed by RQ1, we detail the experimental results data from the MIMIC-III Full dataset and the

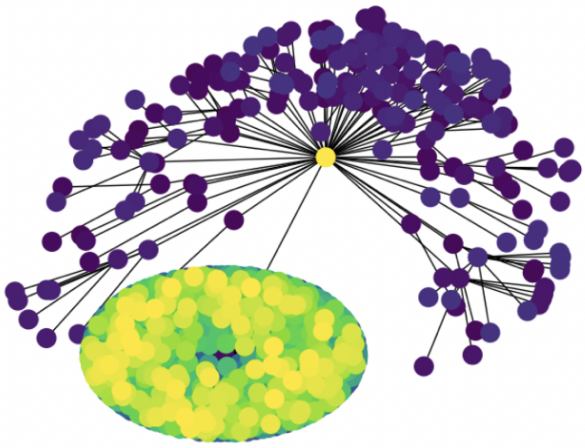

Figure 5: MIMIC-III Top50 Description in Tree Graph.

MIMIC-III Top50 dataset on the basic core assessment metrics and the individualized metrics in Table 1. Meanwhile, we will conduct a rigorous theoretical analysis of the data in Table 1, with the following conclusions.

First, LGGNet achieved the highest scores on the fundamental core assessment and personalized indexes, demonstrating the validity and advancedness of the LGGNet model framework. The standard deviation of the LGGNet model's experimental results on each assessment index is minimal and volatile, confirming the model framework's stability. Furthermore, the same model's performance on different datasets demonstrated the expected variability. The model's overall performance on the MIMIC-III Full dataset is better than on the MIMIC-III Top50 dataset, indicating that the LGGNet model can learn the classification pattern of medical diagnosis text at a deeper level with the expansion of the data level. Meanwhile, the MIMIC-III Full dataset will have more sparse data on rare diseases than the Top50 dataset, demonstrating that the model framework proposed in this paper can better mitigate the long-tail effect of disease diagnosis texts. We use adversarial reinforcement causal learning to explore ICD code generation paths, which aids in the discovery of ICDs of rare disease paths, thereby improving overall ICD code coverage. Finally, the overall data analysis results show that the model framework can generate coding paths for EHRs consistently and efficiently using the hierarchical structure of ICDs, and has achieved excellent overall ICD code classification results thus far.

Second, when compared to LGGNet, the relatively low AUC and F1 values of the two types of models, CAML, DR-CAML, LAAT, and JointLAAT, indicate that these two types of models have low coverage of rare codes and are likely to forget a lot of sparse ICD code data, posing a significant challenge to the models' generalization performance. Because it is critical in the medical diagnosis practice of diseases to diagnose sparse diseases in a timely and correct manner, the space of ICD codes will only grow larger over time, increasing the difficulty in predicting the correct ICD codes and high coverage of rare codes is essential. LGGNet outperforms other baseline models in terms of experimental results because it can reason about the generation path of ICD codes from the root node to the leaf node. Identifying along the path of ICD code nodes, one level at a time, from top to bottom, reduces the search space while improving inference calculation efficiency, effectively alleviating the abovementioned issues.

Third, analyzing and comparing the performance of recursive models based on GRU class (BI-GRU, HA-GRU) in Table 1 reveals that GRU class models perform relatively poorly compared to other models on the difficult task of medical diagnosis disease classification. This is since EHRs are a long sequence of text data that can easily cause the gradient disappearance problem in recursive models based on the GRU class [22], resulting in the model forgetting some critical information and the association between the preceding and the following information. We discovered that keywords or phrases of diseases could provide more critical information due to the specificity of the encoding task of EHRs, and CNN-like models can better capture such important information. As a result, we created the MHR-CNN embedding representation module specifically for LGGNet, which can obtain a more comprehensive view of EHRs using multi-headed CNNs. The gradient disappearance problem can be alleviated by employing a cleverly designed CNN residual connection structure. It is confirmed that LGGNet can solve the above problems by comparing the experimental results of BI-GRU, HA-GRU, and LGGNet.

Generalization in Sparse MIMIC-III (RQ2)

Because ICD codes are increasingly sparse category codes as the hierarchy's depth increases, the performance of our model for sparse data can be intuitively observed by evaluating the LGGNet model framework at different levels. LGGNet can generate ICD code paths by reasoning about the EHR, and we can consider the cumulative category code paths in the codes as a result of EHR inference at specific layers, making it easy to record and compare the model's performance at different ICD layers.

As a result, we designed validation experiments on the MIMIC-III Full dataset, which had the most data and the most complete sparsely labeled categories. Figure 4 depicts the experiment's visualization results. Analyzing the results, we can see that LGGNet's observation F1 decreases slightly as the ICD code level increases. Such an experimental result is entirely consistent with our expectations, as the number of ICD categories increases significantly as the level deepens. The number of ICD codes for rare diseases is even. The number of rare disease-based ICD codes grows exponentially with the depth of the hierarchy, resulting in a significant increase in the model's learning difficulty. To our great surprise, LGGNet's performance degradation due to increased sparse data is significantly controlled and mitigated. For example, increasing the layer level from 1 to 2 reduces LGGNet's microscopic accuracy by nearly 9%, but increasing the layer level from 3 to 4 increases sparse ICD coding far more than increasing the ICD level from 1 to 2. If the model is not designed innovatively with an adversarial domain adaptation algorithm, the model's microscopic

Model	MIMIC-III Full					MIMIC-III Top50				
	AUC		F1		P@8	AUC		F1		P@5
	Macro	Micro	Macro	Micro		Macro	Micro	Macro	Micro	
BI-GRU	0.500	0.547	0.002	0.140	0.317	0.501	0.594	0.035	0.268	0.228
HA-GRU	0.500	0.509	0.017	0.004	0.296	0.500	0.436	0.072	0.124	0.205
CAML	0.895	0.986	0.088	0.539	0.709	0.875	0.909	0.532	0.614	0.609
DR-CAML	0.897	0.985	0.086	0.529	0.609	0.884	0.916	0.576	0.633	0.618
LAAT	0.919	0.988	0.099	0.575	0.738	0.925	0.946	0.666	0.715	0.675
JointLAAT	0.921	0.988	0.107	0.575	0.735	0.925	0.946	0.661	0.716	0.671
ISD	0.938	0.990	0.119	0.559	0.745	0.935	0.949	0.679	0.717	0.682
MSMN	0.950	0.992	0.103	0.584	0.752	0.928	0.947	0.683	0.725	0.680
FUSION	0.915	0.989	0.088	0.554	0.736	0.909	0.933	0.619	0.674	0.647
LGGNet	**0.968** ± 0.001	**0.998** ± 0.002	**0.121** ± 0.001	**0.605** ± 0.003	**0.764** ± 0.001	**0.941** ± 0.002	**0.965** ± 0.001	**0.716** ± 0.004	**0.741** ± 0.001	**0.698** ± 0.003

Table 1: Experiment results on MIMIC-III Top50 and MIMIC-III Full datasets. The results of LGGNet are shown in means ± standard deviations.

Model	MIMIC-III Full					MIMIC-III Top50				
	AUC		F1		P@8	AUC		F1		P@5
	Macro	Micro	Macro	Micro		Macro	Micro	Macro	Micro	
LGGNet	**0.968**	**0.998**	**0.121**	**0.605**	**0.764**	**0.941**	**0.965**	**0.716**	**0.741**	**0.698**
No ARCL	0.834	0.867	0.093	0.509	0.645	0.813	0.852	0.594	0.619	0.521
No MIM	0.901	0.923	0.095	0.547	0.732	0.892	0.930	0.674	0.718	0.626
No MHR-CNN	0.862	0.901	0.099	0.515	0.659	0.833	0.889	0.637	0.629	0.573
No APRF	0.947	0.971	0.109	0.586	0.752	0.928	0.938	0.699	0.725	0.674

Table 2: Ablation experiment results on MIMIC-III Top50 and MIMIC-III Full datasets. The standard deviation of LGGNet results is consistent with the previous table, so it is omitted in this table.

accuracy will decrease by more than 9%, but the proposed microscopic accuracy of LGGNet only decreases by about 3% at this time. There may be instances where LGGNet's microscopic recall and AUC at level 4 are greater than those at level 3.

These findings show that by taking into account the relationship between ICD encoding and using a hybrid policy network for decision making, LGGNet is effective in encoding sparse EHR, reducing the difficulty of the long-tail effect. The beauty of this is that LGGNet can take advantage of the hierarchical relationship between EHR and ICD codes, and accurate prediction of lower-level ICD codes aids in generating deeper ICD-level paths. Furthermore, the hybrid policy network's global policies can detect and correct model inference perceptions at deeper levels in a timely and effective manner.

Ablation (RQ3)

We designed multiple sets of ablation experiments to investigate the role of different modules in LGGNet, and the results of the ablation experiments are shown in Table 2. The variants of LGGNet we considered are as follows: (1) No ARCL denotes LGGNet without adversarial reinforcement casual learning, where we remove the adversarial reinforcement causal learning process from the model and train our model using only teacher-forced and ground truth paths. (2) No MIM denotes LGGNet without a MIM module, i.e., the model loses the parent-child and sibling information transfer in MIM. (3) No MHR-CNN indicates LGGNet that loses the more accurate embedding representation, and we only use the ordinary CNN to complete the extraction of semantic information of EHRs. (4) No APRF denotes LGGNet without adversarial perturbation enhancement of the embedding layer, i.e., the model completes the update iteration of the embedding layer's parameter gradient by only backpropagation. From the experimental results obtained in Table 2, we conducted the following argumentation analysis.

First, the performance of LGGNet decreases significantly after removing the adversarial reinforcement casual learning (i.e., no ARCL). For example, on the MIMIC-III Full dataset, the model's macro AUC and micro AUC metrics decreased by 13.8% and 13.1%, respectively, and the macro F1 and micro F1 metrics of the model decreased by 23.1% and 15.9%, respectively, P@8 decreased by 15.6%. Furthermore, a similar decreasing trend is shown on the MIMIC-III Top50 dataset. The above experimental comparison results show that the design of adversarial reinforcement casual learning has an essential role in the performance of LGGNet. It is verified that adversarial reinforcement casual learning can effectively allow LGGNet to be pointed out by path discriminators, which generates ICD code paths closer to the ground truth paths. At the same time, we performed data augmentation of the ICD path samples by a random path sampling algorithm, which can help the model learn

rare disease ICD paths [9] and enhance the model's robustness.

Second, after removing all phases of messaging (i.e., no MIM), the LGGNet model loses the parent-to-child, sibling-to-sibling messaging function. The results of the ablation study show that LGGNet performs equally unsatisfactorily and suffers from some degree of performance degradation. For example, on the MIMIC-III Top50 dataset, the model's macro AUC and micro AUC metrics drop by 5.2% and 5.9%, respectively, and the macro F1 and micro F1 metrics of the model drop by 5.9% and 3.1%, respectively, P@5 decreased by 10.3%. At the same time, it also shows a certain degree of decline in the MIMIC-III Full dataset. The experiments verify that MIM successfully exploits the interaction between EHR text, ICD code paths, and ICD codes through the message passing mechanism. The experimental ablation results show that MIM plays a significant role in category ICD code path generation.

Third, after removing the multi-headed residual CNN (i.e., no MHR-CNN), we only use the ordinary text-CNN as the embedding layer, thus completing the embedding characterization of EHRs. As shown in the ablation experiments results in Table 2, the performance of LGGNet in each core metric decreases after the MHR-CNN is removed. Taking the MIMIC-III Full dataset as an example, the macro AUC and micro AUC metrics decreased by 10.3% on average, and the macro F1 and micro F1 metrics decreased by 5.9% on average. P@8 decreased by 13.7%. Meanwhile, we also conducted the same ablative experiments on the MIMIC-III Top50 dataset, and the experimental indicators also showed a certain magnitude of the decline. Analyzing the experimental comparison results, we found that the MHR-CNN module in the LGGNet model framework can better characterize the information of the MHR-CNN text. With the help of the multi-head CNN architecture and the mechanical design of the residual CNN, the MHR-CNN as an embedding layer can combine the text information of different visual field domains while alleviating the information forgetting caused by the gradient disappearance.

Fourth, after removing the adversarial perturbation regularization fine-tuning (i.e., no APRF), we update the LGGNet model parameters iteratively by a simple back-propagation algorithm only. After the ablative experiments, we analyze the experimental results in Table 2. Taking the MIMIC-III Top50 dataset as an example, we can find that the macro AUC and micro AUC metrics of the proposed model in this paper decrease by 1.4% and 2.8%. The model's macro F1 and micro F1 indicators decreased by 2.4% and 3.1%, respectively, and P@5 decreased by 2.2%. In addition, the comparative values of the indicators also decreased in the experiments of the MIMIC-III Full dataset. After the experimental comparison, the effectiveness and necessity of APRF in the LGGNet model framework are verified. APRF enhances the characterization ability of the model embedding layer by optimizing the back-propagation algorithm, which makes the model loss curve smoother at the same time.

Data and Code

To facilitate reproducibility of ourwork, we are sharing all resources used in this paper at anonymous github link: https://anonymous.4open.science/r/LGGNet-BB17/.

Acknowledgments

We are sincerely grateful to the reviewers for your precious time and effort in reviewing our article, and we will adopt your valuable review comments seriously.

This work was partially supported by...

Any original ideas, thoughtful deductions, or experimental findings mentioned in this paper's text are solely the author's opinions. They do not necessarily represent the sponsor or employer of the author's endorsement or approval.

Conclusion and Future Work

In this research paper, we rethink the encoding and categorization of electronic health records (EHRs) as an adversarial hierarchical path generation task. The article suggests an adversarial migration-based labeled graph generation network (LGGNet), which not only includes a multi-header residual CNN embedding module to capture different medical text patterns but also a message integration module (MIM) to encode the connection between electronic health records (EHRs) and International Classification of Diseases (ICD) code numbers. Parallel to this, we develop an adversarial domain adaptive algorithm that has two parts: an adversarial perturbation regularization fine-tuning strategy and an adversarial reinforcement causal learning technique for training and fine-tuning LGGNet models.

The LGGNet model proposed in this paper significantly outperforms several competitive baseline models and achieves the best performance to date, according to experimental results on the MIMIC-III benchmark dataset. Furthermore, deep sparse coding and ablation experiments thoroughly illustrate the efficacy and generalizability of the MIM, MHR-CNN, and adversarial domain adaptive algorithms proposed in this paper. The synergy and functions of each module in the LGGNet model have undergone thorough analysis and verification.

Although LGGNet achieves the best classification results on the MIMIC-III dataset, we find that it still suffers from the long-tail effect and occasionally makes classification errors when inferring rare disease categories that infrequently appear in the training set. Furthermore, this paper only employs a limited number of tuning means, and the hyperparameters used thus far may not provide the best performance on the MIMIC-III dataset. There is still room for further improvement. In the next step, we hope to improve the LGGNet model's performance by investigating how to use prior knowledge, automated hyperparameter tuning, an improved loss function, and optimized embedding layer representation.

References

Allamanis, M.; Peng, H.; and Sutton, C. 2016. A Convolutional Attention Network for Extreme Summarization of Source Code. In Balcan, M.; and Weinberger, K. Q., eds.,

Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016, volume 48 of *JMLR Workshop and Conference Proceedings*, 2091–2100. JMLR.org.

Bennett, C. C.; and Hauser, K. 2013. Artificial intelligence framework for simulating clinical decision-making: A Markov decision process approach. *Artificial intelligence in medicine*, 57(1): 9–19.

Cao, P.; Chen, Y.; Liu, K.; Zhao, J.; Liu, S.; and Chong, W. 2020. HyperCore: Hyperbolic and Co-graph Representation for Automatic ICD Coding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 3105–3114. Online: Association for Computational Linguistics.

Choi, E.; Biswal, S.; Malin, B.; Duke, J.; Stewart, W. F.; and Sun, J. 2017. Generating multi-label discrete patient records using generative adversarial networks. In *Machine learning for healthcare conference*, 286–305. PMLR.

dos Santos, C. N.; Tan, M.; Xiang, B.; and Zhou, B. 2016. Attentive Pooling Networks. *CoRR*, abs/1602.03609.

Esteban, C.; Hyland, S. L.; and Rätsch, G. 2017. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*.

Guan, J.; Li, R.; Yu, S.; and Zhang, X. 2018. Generation of synthetic electronic medical record text. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 374–380. IEEE.

Huang, J.; Osorio, C.; and Sy, L. W. 2019. An empirical evaluation of deep learning for ICD-9 code assignment using MIMIC-III clinical notes. *Computer methods and programs in biomedicine*, 177: 141–153.

Ji, S.; Cambria, E.; and Marttinen, P. 2020. Dilated Convolutional Attention Network for Medical Code Assignment from Clinical Text. *arXiv preprint arXiv:2009.14578*.

Johnson, A. E.; Pollard, T. J.; Shen, L.; Li-Wei, H. L.; Feng, M.; Ghassemi, M.; Moody, B.; Szolovits, P.; Celi, L. A.; and Mark, R. G. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3(1): 1–9.

Lanchantin, J.; Sekhon, A.; and Qi, Y. 2019. Neural message passing for multi-label classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 138–163. Springer.

Miotto, R.; Wang, F.; Wang, S.; Jiang, X.; and Dudley, J. T. 2018. Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics*, 19(6): 1236–1246.

Mullenbach, J.; Wiegreffe, S.; Duke, J.; Sun, J.; and Eisenstein, J. 2018a. Explainable Prediction of Medical Codes from Clinical Text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 1101–1111. New Orleans, Louisiana: Association for Computational Linguistics.

Mullenbach, J.; Wiegreffe, S.; Duke, J.; Sun, J.; and Eisenstein, J. 2018b. Explainable prediction of medical codes from clinical text. *arXiv preprint arXiv:1802.05695*.

Nadathur, S. G. 2010. Maximising the value of hospital administrative datasets. *Australian Health Review*, 34(2): 216–223.

Srivastava, R. K.; Greff, K.; and Schmidhuber, J. 2015. Highway networks. *arXiv preprint arXiv:1505.00387*.

Vu, T.; Nguyen, D. Q.; and Nguyen, A. 2020. A Label Attention Model for ICD Coding from Clinical Text. *arXiv preprint arXiv:2007.06351*.

Wang, S.; Ren, P.; Chen, Z.; Ren, Z.; Nie, J.; Ma, J.; and de Rijke, M. 2020a. Coding Electronic Health Records with Adversarial Reinforcement Path Generation. In Huang, J.; Chang, Y.; Cheng, X.; Kamps, J.; Murdock, V.; Wen, J.; and Liu, Y., eds., *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, 801–810. ACM.

Wang, S.; Ren, P.; Chen, Z.; Ren, Z.; Nie, J.-Y.; Ma, J.; and de Rijke, M. 2020b. Coding Electronic Health Records with Adversarial Reinforcement Path Generation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 801–810.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4): 229–256.

Xie, P.; and Xing, E. 2018. A Neural Architecture for Automated ICD Coding. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1066–1076. Melbourne, Australia: Association for Computational Linguistics.

Xie, X.; Xiong, Y.; Yu, P. S.; and Zhu, Y. 2019. EHR Coding with Multi-scale Feature Attention and Structured Knowledge Graph Propagation. In Zhu, W.; Tao, D.; Cheng, X.; Cui, P.; Rundensteiner, E. A.; Carmel, D.; He, Q.; and Yu, J. X., eds., *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, 649–658. ACM.

Yin, W.; and Schütze, H. 2017. Attentive Convolution. *CoRR*, abs/1710.00519.

Yu, K.; Wang, Y.; Cai, Y.; Xiao, C.; Zhao, E.; Glass, L.; and Sun, J. 2019. Rare disease detection by sequence modeling with generative adversarial networks. *arXiv preprint arXiv:1907.01022*.