Self-Supervised Graph Learning With Hyperbolic Embedding for Temporal Health Event Prediction

Chang Lu, Chandan K. Reddy, *Senior Member, IEEE*, and Yue Ning, *Member, IEEE*

Abstract—Electronic health records (EHRs) have been heavily used in modern healthcare systems for recording patients' admission information to health facilities. Many data-driven approaches employ temporal features in EHR for predicting specific diseases, readmission times, and diagnoses of patients. However, most existing predictive models cannot fully utilize EHR data, due to an inherent lack of labels in supervised training for some temporal events. Moreover, it is hard for the existing methods to simultaneously provide generic and personalized interpretability. To address these challenges, we propose Sherbet, a self-supervised graph learning framework with hyperbolic embeddings for temporal health event prediction. We first propose a hyperbolic embedding method with information flow to pretrain medical code representations in a hierarchical structure. We incorporate these pretrained representations into a graph neural network (GNN) to detect disease complications and design a multilevel attention method to compute the contributions of particular diseases and admissions, thus enhancing personalized interpretability. We present a new hierarchy-enhanced historical prediction proxy task in our self-supervised learning framework to fully utilize EHR data and exploit medical domain knowledge. We conduct a comprehensive set of experiments on widely used publicly available EHR datasets to verify the effectiveness of our model. Our results demonstrate the proposed model's strengths in both predictive tasks and interpretable abilities.

Index Terms—Electronic health records (EHRs), event prediction, graph learning, hyperbolic embeddings, model interpretability.

I. INTRODUCTION

GIVEN the promising potential of electronic health records (EHRs), mining interpretable predictive patterns from EHR data has a significant value in healthcare and has drawn a lot of attention in recent years. EHR data are complex in nature and typically contain sequences of patients' admission records, such as diagnoses, clinical notes, and medications. Effective analysis of EHR data is important for
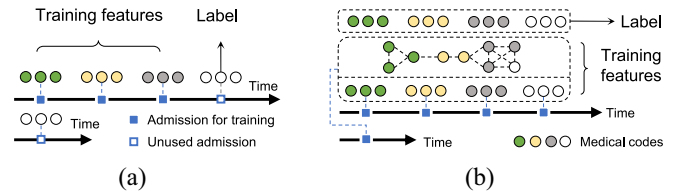
Fig. 1. Example of predicting temporal events with supervised training on original EHR data in a temporal setting and with self-supervised training on extracted EHR data in a graph structure. (a) Supervised training. (b) Self-supervised training.

both medical professionals and patients as it can provide preventative health alerts and personalized care plans.

A variety of predictive models using deep learning technology has been proposed for predicting temporal events, such as diagnosis prediction [1]–[5]; mortality prediction [6]–[9]; risk prediction [10]–[13]; and medication recommendation [14], [15]. A common supervised training approach to utilize EHR data for temporal event prediction is to use previous records as features and the records of next admissions as labels. However, this approach will inherently ignore patients' final admissions due to the lack of labels. Moreover, learning effective representations for medical concepts by effectively leveraging the domain knowledge is still an open problem in healthcare applications. In summary, there are still some challenges for predictive models using temporal information of EHR data.

A. Fully Utilizing EHR Data

A large number of EHR records are unused in tradiitional supervised training. Final admission records of patients, including single admissions, are discarded for training because the labels are missing for the next potential admissions. Fig. 1(a) shows the reason for the lack of labels while predicting temporal events. For multiple-admission patients (top arrow), the final admission record is used as the label for a prediction and cannot be used as training features. For single-admission patients (bottom arrow), there are no labels because there are no next admissions. However, according to the statistics of a widely used EHR dataset [16], the number of multiple-admission patients is only 20% of single-admission patients. Hence, a majority of valuable information in EHR data is discarded.

B. Exploiting Disease Hierarchies During Prediction

In modern disease classification systems like ICD-9-CM [17], diseases are classified into various categories as medical codes in multiple levels and form a hierarchical structure. Existing methods, such as GRAM [3] and G-BERT [15], mainly use this structure to extract disease features using attention methods. However, this type of domain knowledge is helpful in guiding model predictions. By predicting disease hierarchies, representations of diseases can be further refined.

C. Learning Hidden Representations for Related Diseases

Most existing works treat diseases independently while neglecting disease interactions (i.e., complications). However, such complications are generally crucial in medical practice. For example, longstanding hypertension will eventually lead to heart failure (HF). Consequently, it is very common for patients with HF to have been suffering from hypertension prior to being admitted for an HF condition [18].

D. Simultaneously Providing Generic and Personalized Interpretability

Generic interpretability provides discovered common knowledge such as disease complications from the entire set of patient records. Personalized interpretability refers to explanations for individual patients based on their personal admission records. Both these two kinds of interpretabilities should be considered in healthcare models to make prediction results reliable to doctors and patients. However, current approaches [3], [4], [19], [20] mainly focus on one type of interpretability, while it is critical to simultaneously provide both generic and personalized interpretability.

To address these challenges, we propose **Sherbet**, a self-supervised graph learning framework with hyperbolic embeddings for temporal health event prediction. As a subset of unsupervised learning methods, the self-supervised learning method in this work is different from other pretraining methods, such as G-BERT [15]. We design a special proxy task for self-supervised learning to hierarchically predict historical diagnoses of patients. Fig. 1(b) shows the manner in which self-supervised learning can fully utilize EHR data. The proxy task constructs an interaction graph for medical codes in all admissions rather than treating each admission independently. It enables us to incorporate single-admission patients and the final admissions of multiple-admission patients by generating new labels for all admissions. When implementing this task, we first pretrain disease representations using a new hyperbolic embedding method with information flow to reconstruct the disease hierarchical structure. In order to model disease interactions, we next construct a weighted and directed graph for diseases based on their occurrences in patients' admission records. Then, we design a graph encoder architecture for self-supervised learning. The first part of the graph encoder is a graph neural network (GNN) on the constructed graph to extract hidden disease representations and further learn the disease complications. Then, we develop a multilevel attention mechanism as the encoder to learn the representation of admissions and patients from the admission records. The

contribution of specific diseases and admissions to a given prediction task can thus be quantified. The self-supervised learning component, which is also the decoder, is designed with the proxy task of hierarchically predicting historical diseases. Finally, we build a fine-tuning module for specific tasks. The main contributions of this work are summarized as follows.

1) We propose a novel self-supervised graph learning framework and a hierarchy-enhanced historical prediction task to fully exploit the admission records in EHR data and hierarchical structures of medical codes.
2) We propose a new hyperbolic embedding method with an information flow strategy to pretrain medical code representations using the disease hierarchical structure. It can simultaneously consider hierarchical domain knowledge and similarities among medical codes.
3) We design a weighted and directed disease interaction graph to learn the disease complications as generic interpretability. Together with multilevel attention, the proposed model is able to provide generic and personalized interpretability.

The remainder of this article is organized as follows. Section II summarizes the related work. Section III formally defines the prediction problem and self-supervised learning task. Then, we demonstrate the experimental settings and results in Sections IV and V. Finally, we summarize our work and discuss potential future research in Section VI.

II. RELATED WORK

Predictive Models in Healthcare: Deep learning methods have been widely adopted for learning effective representations of complex and dynamic data in a wide range of applications including temporal event modeling in healthcare. Choi *et al.* [1] proposed DoctorAI to predict diagnoses in following admissions and the time interval of hospital readmissions using recurrent neural networks (RNNs) with gated recurrent unit (GRU [21]) cells. A reverse-time RNN model with attention, RETAIN, was proposed by Choi *et al.* [19] to predict HF and provide some interpretability of predictive models. Nguyen *et al.* [10] proposed Deepr, which regards diseases as words and admission records as sentences and uses convolutional neural networks (CNNs) as a language model to predict readmission possibilities of patients in the next three months. Ma *et al.* [20] proposed Dipole using a bidirectional RNN with various attention methods to predict future diagnoses. Bai *et al.* [4] considered the time duration between two admissions and proposed the Timeline model. Luo *et al.* [22] used a self-attention-based method to detect key time steps in patients' historical admissions. However, as discussed above, these predictive models usually do not consider data that lack labels (such as single and last admissions) and, thus, cannot fully utilize the potential of EHR data.

Unsupervised and Self-Supervised Learning in Healthcare: Self-supervised learning refers to training models with automatically generated labels [23]. It is used for obtaining distinguishable features of samples by pretraining the model on proxy tasks. Gidaris *et al.* [24] created a pretext task to predict

image rotation using ConvNet. Xu *et al.* [25] proposed a self-supervised framework LabNet by leveraging word vectors of both seen and unseen labels for cross-modal retrieval. A common approach when using EHR data is to treat the diseases and the admission records as words and sentences, respectively. Then, popular language models, such as Transformer [26] and BERT [27], can be applied to learn the representation of diseases. Choi *et al.* [3] applied GloVe [28] to initialize disease embeddings in a medical ontology tree using labeled data. Shang *et al.* [15] proposed the G-BERT model to recommend medicines for patients considering an admission as a sentence and using BERT to pretrain disease embeddings. However, one problem of applying language models on EHR data is that diagnoses in an admission record typically do not have an ordering (like words in a sentence). Therefore, language models may not fit EHR data well because a different order of diagnoses may lead to significantly different prediction results.

Graph Neural Networks and Hyperbolic Representations: GNNs are developed for data with graph structures. Yu *et al.* [29] proposed an adaptive hypergraph learning approach by varying the neighborhood size for transductive image classification. Kipf and Welling [30] proposed graph convolutional networks (GCNs) to generalize CNN for node classifications. Nickel and Kiela [31] designed a Poincaré ball model and an optimization method to learn representations for hierarchical data. Chami *et al.* [32] leveraged hyperbolic embedding to improve knowledge graph representations in low dimensions. Furthermore, a self-supervised hyperboloid embedding learning method [33] was proposed to capture hierarchical semantic information in knowledge graphs. Recently, GNNs have also become popular and effective for modeling the EHR data. Choi *et al.* [3] used a medical ontology graph based on hierarchical domain knowledge and applied an attention mechanism to aggregate disease embeddings in different hierarchies. Shang *et al.* [15] also utilized this knowledge but designed a two-stage attention method for diseases. Choi *et al.* [34] proposed GCT, a graph convolutional transformer by constructing a graph of diagnoses, treatments, and lab results. Lu *et al.* [35] considered horizontal links in the medical ontology graph and constructed a patient-disease graph to learn hidden disease relations. Most models treat diseases independently or only apply graph-domain knowledge (e.g., hierarchical medical code classification) in feature extractions, but do not take into account a common fact that diseases in different classes can also have strong interactions, that is, disease complications.

Given these problems in the existing work, we develop a method that can utilize more information in EHR data. The proposed method takes disease hierarchical structures and hidden disease relations into account in temporal predictions. In addition, we also focus on providing interpretability from both disease and patient aspects.

III. PROPOSED METHODOLOGY

We first describe the basic notations and formulate the problem of predicting temporal events (Section III-A). Next, we demonstrate an unsupervised learning method to initialize
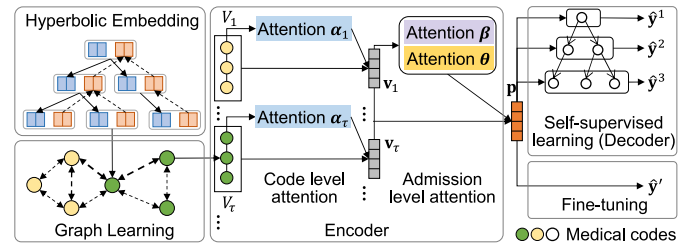

Fig. 2. Overview of the proposed Sherbet model. Hyperbolic representations of medical codes are first pretrained with a hierarchical structure. Then, the Sherbet graph is constructed based on the occurrence of medical codes in admission records. A GNN is adopted to learn the hidden embedding of medical codes. Next, an encoder including the code level attention and admission level attention is designed to encode the admission records of a patient into a patient vector. For self-supervised learning, a historical hierarchy prediction task is designed to utilize the hierarchical structure of medical codes in prediction. Finally, after self-supervised learning, we can apply fine-tuning for specific prediction tasks.

TABLE I
NOTATIONS USED IN THIS ARTICLE

Notation	Description
$\mathcal{D}$	EHR dataset
$\mathcal{H}, H$	Disease hierarchical structure and level number
$\mathcal{U}, \mathcal{C}$	Sets of patients and medical codes
V_τ	τ-th admission record for a patient
$\mathbf{E}$	Embeddings of medical codes
$\mathbf{X}$	Learned hidden embeddings of medical codes
$\mathbf{A}$	Adjacency matrix of medical codes
$\mathbf{v}_\tau$	Embedding vector of the τ-th admission
$\mathbf{p}$	Embedding vector of a patient

disease representations using hyperbolic embeddings for the disease hierarchical structure (Section III-B). Then, we introduce the strategy to construct the disease graph (Section III-C), followed by the proposed self-supervised learning method using a graph encoder–decoder (Section III-D). The fine-tuning of the model is performed after self-supervised learning (Section III-E). Finally, we discuss the interpretability of Sherbet from two perspectives including: 1) disease complications as well as specific contributions of historical diagnoses and 2) admissions to the predictions (Section III-F). An overview of the system framework of Sherbet is shown in Fig. 2.

A. Problem Formulation

An EHR dataset contains temporal admission records of patients. In each admission, a patient is diagnosed with one or more diseases represented by medical codes, in the format of ICD-9-CM [17] or ICD-10 [36]. We denote the entire set of medical codes by $\mathcal{C} = \{c_1, c_2, \ldots, c_{|\mathcal{C}|}\}$ in the EHR dataset. For a patient u, one clinical record $V_\tau^u \subset \mathcal{C}$ is a subset of $\mathcal{C}$, where $\tau = 1, 2, \ldots, T^u$ denotes the τth admission record of patient u who has a total of T^u admissions. In the rest of this article, we drop the superscript u in V^u and T^u for better readability unless otherwise stated. The important symbols used in this article are listed in Table I.

Definition 1 (EHR Dataset): An EHR dataset is given by $\mathcal{D} = \{r_u | u \in \mathcal{U}\}$, where $\mathcal{U}$ is the entire set of patients in $\mathcal{D}$,

and $r_u = (V_1, V_2, \ldots, V_T)$ is the admission records of patient u. Each admission $V_\tau \subset \mathcal{C}$ contains a subset of $\mathcal{C}$.

Definition 2 (Health Event Prediction): Given an EHR dataset $\mathcal{D}$ and a patient u who has T historical admissions, the goal of the prediction task in this article is to predict the future health event $\mathbf{y}_{T+1}$ for patient u, such as diagnosis or HF prediction. For instance, if the task is to predict the diagnoses for a patient's $(T+1)$th admission, the goal will be estimating the probabilities of all medical codes, that is, $\mathbf{y}_{T+1} \in \{0, 1\}^{|\mathcal{C}|}$ in the $(T+1)$th admission of this patient.

B. Hyperbolic Embedding With Information Flow

The ICD-9-CM system provides a domain knowledge base to classify diseases into various categories represented by medical codes on multiple levels. In this system, the medical codes form a hierarchical structure $\mathcal{H}$ with H levels, that is, a tree. To obtain effective representations of medical codes, we aim to learn the embeddings of medical codes by reconstructing the skeleton of $\mathcal{H}$. We take advantage of the Poincaré ball model [31], [37] to learn the representations of the hierarchical structure of diseases, which encodes nodes in $\mathcal{H}$ to a hyperbolic space. The distance in the hyperbolic space between embedding vectors $\mathbf{e}_i$ and $\mathbf{e}_j$ of two medical codes $c_i, c_j \in \mathcal{H}$ is defined as

$$d(\mathbf{e}_i, \mathbf{e}_j) = \cosh^{-1}\left(1 + 2\frac{\|\mathbf{e}_i - \mathbf{e}_j\|^2}{\left(1 - \|\mathbf{e}_i\|^2\right)\left(1 - \|\mathbf{e}_j\|^2\right)}\right). \quad (1)$$

In $\mathcal{H}$, higher level diseases can be regarded as a summary of their children, while lower level diseases provide more precise descriptions for their parents. Following this intuition, we also propose an information flow strategy to model the similarity and distinction among ancestor nodes and children nodes in $\mathcal{H}$. Together with the hyperbolic embedding, we are able to simultaneously consider the hierarchy and similarity of medical codes.

In practice, medical codes recorded in EHR datasets are usually leaf nodes. In some cases when a patient is diagnosed with higher level diseases, that is, nonleaf nodes, we recursively create virtual child nodes for each nonleaf node to pad them into virtual leaf nodes in the same level. Therefore, in this article, set $\mathcal{C}$ contains only leaf nodes and virtual leaf nodes in the last level. We use $|\mathcal{H}|$ to denote the total number of medical codes in $\mathcal{H}$. To represent the information flow in $\mathcal{H}$, we first assign each medical code c_i with two randomly initialized trainable embedding vectors: 1) a shared vector $\mathbf{s}_i \in \mathbb{R}^d$ and 2) a local vector $\mathbf{t}_i \in \mathbb{R}^d$. The shared vector $\mathbf{s}_i$ is designed to represent the information inherited from its parent. The local vector $\mathbf{t}_i$ contains private and more precise information of c_i that makes c_i different from its parent. Then, the public embedding vector $\mathbf{e}'_i$ of c_i is calculated as

$$\mathbf{e}'_i = \lambda_i \times \mathbf{s}_i + (1 - \lambda_i) \times \mathbf{t}_i \in \mathbb{R}^d \quad (2)$$

where λ_i is a trainable coefficient to integrate the shared and local vectors. To capture the information flow in the disease hierarchical structure, we propose a hierarchical embedding method using the shared and local embedding vectors of each
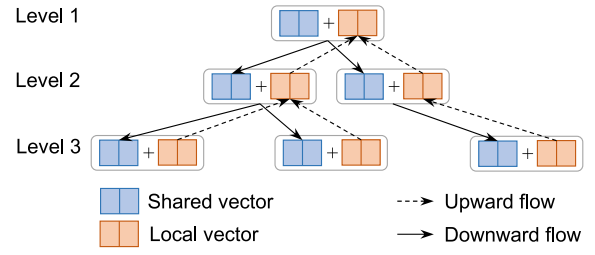

Fig. 3. Information flow in disease hierarchical structure. The downward flow starts from a parent node to the shared part of a child node. The upward flow starts from the local part of a child node to the local part of a parent node.

medical code. Fig. 3 shows the information flow in two directions, that is: 1) downward and 2) upward flows. A downward flow indicates that the shared vector of a lower level medical code inherits the public vector of its parent, and an upward flow simulates the summary of this medical code's children by aggregating the children's local vectors. Such flows can be summarized as follows:

$$2\mathbf{s}'_i = \mathbf{e}'_j \quad \text{(Downward flow)} \quad (3)$$

$$\mathbf{t}'_i = \frac{1}{n_i}\sum_{k=1}^{n_i} \mathbf{t}_k \quad \text{(Upward flow)} \quad (4)$$

where $c_j \in \mathcal{H}$ is the parent of c_i, $c_k \in \mathcal{H}$ is a child of c_i, and n_i is the number of c_i's children. After this flow process, we use (2) on $\mathbf{s}'_i$ and $\mathbf{t}'_i$ to calculate the new public embedding vector $\mathbf{e}_i$.

To reconstruct the hierarchical structure $\mathcal{H}$, we aim to minimize an overall distance among all nodes in $\mathcal{H}$. We calculate the distance between the public embeddings of two medical codes because $\mathbf{e}_i$ contains the information passed from c_i's parent and children. We assume the distance should be small between two connected nodes, while large for unconnected nodes. Let $\mathcal{A} = \{(i,j)|c_i, c_j \in \mathcal{H}\}$ be the set of connected node pairs in $\mathcal{H}$, the loss function $\mathcal{L}_{\text{rec}}$ of reconstructing the hierarchical structure using the hyperbolic distance between two node embedding vectors is defined as

$$\mathcal{L}_{\text{rec}} = -\sum_{(i,j)\in\mathcal{A}} \log \frac{e^{-d(\mathbf{e}_i, \mathbf{e}_j)}}{\sum_{j' \in \mathcal{N}(i)\cup\{v\}} e^{-d(\mathbf{e}_i, \mathbf{e}_{j'})}}. \quad (5)$$

Here, $\mathcal{N}(i) = \{j'|(c_i, c_{j'}) \notin \mathcal{A}\}$ denotes the set of non-adjacent nodes for c_i. Finally, we use $\mathcal{L}_{\text{rec}}$ to pretrain the representation of medical codes using backpropagation and obtain the final embedding $\mathbf{E} \in \mathbb{R}^{|\mathcal{C}|\times d}$, which is the collection of $\mathbf{e}_i$ of medical codes in the lowest level, that is, leaf nodes. The pretrained medical code embeddings $\mathbf{E}$ will be further used in the self-supervised graph learning stage.

The pseudocode of hyperbolic embedding is shown in Procedure 1. Lines 2–6 summarize the information flow and lines 7–9 correspond to the optimization process to pretrain the embeddings using $\mathcal{L}_{\text{rec}}$.

C. Sherbet Graph Construction

Disease complications are closely related to their co-occurrence frequencies. For example, hypertension and HF are

---

Procedure 1: HyperbolicEmbedding($\mathcal{H}$)

Input: The hierarchical structure $\mathcal{H}$ of medical codes
Output: Embeddings $\mathbf{E}$ for leaf nodes of $\mathcal{H}$
1 Randomly initialize shared and local vectors
2 **for** $c_i \in \mathcal{H}$ **do**
3     $\mathbf{e}'_i \leftarrow$ combine shared and local vector $\mathbf{s}_i, \mathbf{t}_i$
4     $\mathbf{s}'_i, \mathbf{t}'_i \leftarrow$ Downward and upward flow
5     $\mathbf{e}_i \leftarrow$ combine $\mathbf{s}'_i, \mathbf{t}'_i$
6 **end**
7 **repeat**
8   Calculate distance by Equation (1)
9   Optimizing embedding vectors using $\mathcal{L}_{rec}$
10 **until** *convergence*
11 **return** $\{\mathbf{e}_i | c_i \text{ is a leaf node in } \mathcal{H}\}$

---

often diagnosed on the same patient. Therefore, we design a directed and weighted graph $\mathcal{G}$ to represent the subgraph patterns of disease complications. In this graph, each vertex c_i of $\mathcal{G}$ is a medical code in $\mathcal{C}$. To describe the complication of two diseases, that is, medical codes c_i and c_j, we create directed edges between vertices c_i and c_j using the following rule: for each medical code pair (c_i, c_j), if c_i and c_j occur once in a record V_τ of a patient u, we add two edges (i, j) and (j, i) to $\mathcal{G}$. Here, we create a directed graph because we think such disease relationships are asymmetric in nature, since two diseases may not have the same influence on each other. One disease c_i could be a major complication of another disease c_j, however, the opposite might not be true. Therefore, we use directed and weighted edges to describe such dual relationships between two diseases by quantifying such influence in an adjacency matrix $\mathbf{A} \in \mathbb{R}^{|\mathcal{C}| \times |\mathcal{C}|}$, where each element $\mathbf{A}_{ij}$ is the weight of edge (i, j). To model the dual influence of diseases, we first define a co-occurrence matrix $\mathbf{B} \in \mathbb{N}^{|\mathcal{C}| \times |\mathcal{C}|}$ initialized with all zeros. When calculating the values of $\mathbf{B}$, we increase the elements $\mathbf{B}_{ij}$ and $\mathbf{B}_{ji}$ by 1 for each co-occurrence pair of c_i and c_j in each admission record of all patients. Letting $q_i = \sum_{j=1}^{|\mathcal{C}|} \mathbf{B}_{ij}$ be the sum of the ith row of $\mathbf{B}$, then the weighted adjacency matrix $\mathbf{A}$ is calculated as follows:

$$\mathbf{A}_{ij} = \begin{cases} 0, & \text{if } i = j \text{ and } q_i \neq 0 \\ 1, & \text{if } i = j \text{ and } q_i = 0 \\ \frac{\mathbf{B}_{ij}}{q_i}, & \text{otherwise.} \end{cases} \quad (6)$$

Note that $\mathbf{A}$ is typically not symmetric, which makes $\mathcal{G}$ a weighted and directed graph. The element $\mathbf{A}_{ij}$ measures the frequency of the disease pair (c_i, c_j) in all co-occurrence pairs of c_i. A higher frequency implies that c_j appears more times along with c_i than other medical codes. Therefore, we can infer c_j has more influence on c_i, and c_j is a more common complication of c_i than other diseases.

D. Self-Supervised Graph Learning

1) Graph Learning: Based on the key idea of constructing the disease graph $\mathcal{G}$, the influence of adjacent nodes on each other corresponds to the weights on their edges. Therefore, we adopt a multilayer GNN to further learn the hidden representation of medical codes, given GNN's scalability

and effective representation power. By incorporating weighted edges, GNN can scale features and pay more attention to the neighbors with higher weights. Given the initial embedding $\mathbf{E}$ of medical codes learned from hyperbolic embedding and the weighted adjacency matrix $\mathbf{A}$, the hidden representation $\mathbf{X}$ of medical codes can be calculated by a multilayer GNN: $\mathbf{X} = \text{GNN}(\mathbf{A}, \mathbf{E})$, based on GCN.

More specifically, we first use the embeddings of all medical codes as inputs: $\mathbf{H}^{(0)} = \mathbf{E}$. Then, the lth GNN layer to aggregate the features of medical codes is described as follows:

$$\mathbf{H}^{(l+1)} = \text{ReLU}\left(\hat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}_g^{(l)}\right). \quad (7)$$

Here, $\mathbf{H}^{(l)} \in \mathbb{R}^{|\mathcal{C}| \times d^{(l)}}$ and $\mathbf{H}^{(l+1)} \in \mathbb{R}^{|\mathcal{C}| \times d^{(l+1)}}$ are the input and output of the lth layer, respectively. $\mathbf{W}_g^{(l)} \in \mathbb{R}^{d^{(l)} \times d^{(l+1)}}$ is the weight of the lth layer. In addition, $\hat{\mathbf{A}}$ is a normalized adjacency matrix of $\mathbf{A}$, which is calculated as follows:

$$\hat{\mathbf{A}}_{ij} = \frac{\widetilde{\mathbf{A}}_{ij}}{\sum_{j=0}^{|\mathcal{C}|} \widetilde{\mathbf{A}}_{ij}}$$

$$\widetilde{\mathbf{A}} = (1 - \varphi)\mathbf{A} + \varphi\mathbf{I}$$

where $\mathbf{I}$ is the identity matrix, and $0 < \varphi < 1$ is a hyperparameter to adjust the weight of self-loops when adding $\mathbf{I}$ to $\mathbf{A}$. Note that we modify the calculation of $\hat{\mathbf{A}}_{ij}$ and $\widetilde{\mathbf{A}}$ in the original GCN to adjust the importance of center medical codes. A larger φ denotes higher importance of a center medical code when aggregating neighbor codes into this medical code. In the last GNN layer, that is, the Lth layer, we obtain the output of the GNN. We let $\mathbf{X} = \mathbf{H}^{(L)} \in \mathbb{R}^{|\mathcal{C}| \times m}$ be the hidden representation learned by the GNN, where $m = d^{(L)}$ is the dimension of the hidden embedding in the last layer.

2) Multilevel Attention: After obtaining the hidden representation $\mathbf{X}$ of all medical codes in $\mathcal{C}$, we want to use all records in the EHR data to further train the model parameters as well as the embedding matrix $\mathbf{E}$ of medical codes. The data include both single and multiple admission records. For each patient u, we use an encoder to encode all admission records of u into a patient vector $\mathbf{p}$ to represent u using $\mathbf{X}$

$$\mathbf{p} = \text{Encoder}(V_1, V_2, \ldots, V_T | \mathbf{X}). \quad (8)$$

Specifically, we apply a two-level attention mechanism as follows.

a) Code-level attention: Without loss of generality, we assume an admission record V_τ contains n medical codes. Then, the embedding $\mathbf{x}_i$ of each medical code $c_i \in V_\tau$ can be looked up with $\mathbf{X}$. We adopt a global attention mechanism [38] on the medical codes in V_τ to aggregate their embeddings to an admission embedding $\mathbf{v}_\tau$ for the τth admission

$$\mathbf{z}_i = \tanh(\mathbf{W}_c \mathbf{x}_i) \in \mathbb{R}^a$$

$$\alpha_\tau^i = \frac{\exp\left(\mathbf{z}_i^\top \mathbf{w}_\alpha\right)}{\sum_{j=1}^n \exp\left(\mathbf{z}_j^\top \mathbf{w}_\alpha\right)} \in \mathbb{R}$$

$$\mathbf{v}_\tau = \sum_{i=1}^n \alpha_\tau^i \mathbf{x}_i \in \mathbb{R}^m.$$

Here, $\mathbf{W}_c \in \mathbb{R}^{a \times m}$ is a weight matrix, where a is the attention dimension. $\mathbf{w}_\alpha \in \mathbb{R}^a$ is the weight to calculate

the attention score α_τ^i for medical code c_i in an admission. The code attention score $\boldsymbol{\alpha}_\tau = [\alpha_\tau^1, \alpha_\tau^2, \ldots, \alpha_\tau^n]$ measures the distribution of medical codes in an admission. Finally, we multiply the score with $\mathbf{x}_i$ and calculate a weighted sum as the admission embedding $\mathbf{v}_\tau$.

b) Admission-level attention: After calculating the embedding $\mathbf{v}_\tau$ of τth admission of a patient, we need to learn the patient embedding using all admissions. First, we project the admission embedding $\mathbf{v}_\tau$ to the patient dimension

$$\tilde{\mathbf{v}}_\tau = \text{LeakyReLU}(\mathbf{W}_u \mathbf{v}_\tau) \in \mathbb{R}^p. \qquad (9)$$

Here, $\mathbf{W}_u \in \mathbb{R}^{p \times m}$ is a weight matrix. p is the patient dimension. We select LeakyReLU [39] as the activation function based on a hyperparameter tuning procedure in experiments.

Similar to code-level attention, we also use the global attention for admissions to aggregate the admission embeddings and calculate the patient embedding $\mathbf{p}$. However, the original global attention can only measure the significance of each admission, while it cannot distinguish the contribution of each admission to the specific dimension of the output. For example, if a model predicts a set of diagnoses in the next admission, given five previous admissions, we aim to quantify the contribution of each admission to every predicted code. Therefore, besides the attention score, we also design a coefficient $\boldsymbol{\theta}_\tau$ to quantify such contribution

$$\mathbf{r}_\tau = \tanh(\mathbf{W}_v \tilde{\mathbf{v}}_\tau) \in \mathbb{R}^b$$

$$\beta_\tau = \frac{\exp(\mathbf{r}_\tau^\top \mathbf{w}_\beta)}{\sum_{\tau=1}^T \exp(\mathbf{r}_\tau^\top \mathbf{w}_\beta)} \in \mathbb{R}$$

$$\boldsymbol{\theta}_\tau = \frac{\exp(\mathbf{r}_\tau^\top \mathbf{W}_\theta)}{\sum_{\tau=1}^T \exp(\mathbf{r}_\tau^\top \mathbf{W}_\theta)} \in \mathbb{R}^p$$

$$\mathbf{p} = \sum_{\tau=1}^T \beta_\tau \boldsymbol{\theta}_\tau \odot \tilde{\mathbf{v}}_\tau \in \mathbb{R}^p. \qquad (10)$$

Here, $\mathbf{W}_v \in \mathbb{R}^{b \times p}$ is a weight matrix and $\mathbf{w}_\beta \in \mathbb{R}^b$ is the weight to calculate the attention score $\beta_\tau \in \mathbb{R}$ for t-th admission. The attention score $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_T]$ measures the distribution within admissions. In addition, $\mathbf{W}_\theta \in \mathbb{R}^{b \times p}$ is the weight to calculate the attention score $\boldsymbol{\theta}_\tau$. The attention score $\boldsymbol{\theta} = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_T]$ is the distribution of $\tilde{\mathbf{v}}_\tau$ over each dimension of the patient embedding. Finally, we calculate a weighted sum of $\tilde{\mathbf{v}}_\tau$ as the patient embedding $\mathbf{p}$. Specifically, the weight is calculated by the multiplying β_τ and $\boldsymbol{\theta}_\tau$, and $\odot$ is the elementwise multiplication. We propose (10) because we want to simultaneously measure the importance (β_τ) of an admission compared to other admissions, and the contribution ($\boldsymbol{\theta}_\tau$) of this admission to the output. We will further elaborate this idea in Section III-F.

Procedure 2 summarizes the pseudocode of the multilevel attention encoder. Lines 2–4 calculate admission vectors using the code-level attention. Then, the patient vector $\mathbf{p}$ is computed using the admission level attention at line 5.

3) Self-Supervised Learning: To leverage all records in an EHR dataset, we need to utilize records of single-admission patients and final admission records of multiple-admission patients. Since these records lack labels regarding their next

---

Procedure 2: Encoder($u, \mathcal{D}, \mathbf{X}$)

Input: A patient u, an EHR dataset $\mathcal{D}$, the GNN output of medical code embeddings $\mathbf{X}$

Output: The patient embedding vector $\mathbf{p}$ of u

1 $V_1, V_2, \ldots, V_T \leftarrow$ Get admission records of u from $\mathcal{D}$
2 **for** $\tau \leftarrow 1$ *to* T **do**
3     $\mathbf{v}_\tau \leftarrow$ Calculate the admission embedding using the code-level attention with $\mathbf{X}$
4 **end**
5 $\mathbf{p} \leftarrow$ Calculate the patient embedding using the admission-level attention
6 **return** $\mathbf{p}$

---

potential admissions, we focus on reconstructing historical diagnoses using the patient embedding $\mathbf{p}$. We conjecture that the learned patient representation will be able to reflect the historical admission records of this patient. Therefore, recovering the historical diagnoses of this patient takes advantage of the complete dataset and further optimizes the hidden representation of medical codes.

Assuming the historical diagnoses set is denoted as $\mathcal{V}$, we aim to decode the medical codes from $\mathbf{p}$ into the probability distribution $\hat{\mathbf{y}}$: $\hat{\mathbf{y}}_i = P(c_i \in \mathcal{V}|\mathbf{p})$. A naïve method is using a multilayer perceptron (MLP) to simulate the function: $\hat{\mathbf{y}} = \sigma(\text{MLP}(\mathbf{p}))$. This method directly predicts the distribution from $\mathbf{p}$ but does not utilize the hierarchical structure of medical codes. We can make levelwise predictions by calculating the conditional probability of lower levels, once we obtain the probabilities of higher levels. Therefore, we design a *hierarchy-enhanced historical prediction* task to incorporate the hierarchical structure of medical codes.

Definition 3 (Historical Hierarchy of Diagnoses): Given a patient u, the historical hierarchy of diagnoses in the hth level of $\mathcal{H}$ is defined as $\mathcal{V}^h = \{\rho_{c_i}^h | c_i \in \bigcup_{\tau=1}^T V_\tau\}$, where $\rho_{c_i}^h$ denotes the ancestor of c_i in the hth level. At the leaf level, we have $\rho_{c_i}^H = c_i$.

Here, $\mathcal{V}^h$ is the set of medical codes in level h that a patient has ever been diagnosed during their previous T admissions.

Definition 4 (Hierarchy-Enhanced Historical Prediction): Given a patient u, this task is to predict the probability distribution $\hat{\mathbf{y}}^h \in \mathbb{R}^{n_h}$ of ground-truth labels in the hth level ($h = 1, \ldots, H$), where n_h is the number of medical codes in the hth level, and $n_H = |\mathcal{C}|$.

This is a multilabel classification task for each level. Given a code $c_i = \rho_{c_i}^h$ in the hth level, according to the directed graphical model for joint probability [40], the joint probability $\hat{\mathbf{y}}_j^h = P(\rho_{c_i}^h \in \mathcal{V}^h, \rho_{c_i}^{h-1} \in \mathcal{V}^{h-1}, \ldots, \rho_{c_i}^1 \in \mathcal{V}^1|\mathbf{p})$ can be calculated as follows:

$$\hat{\mathbf{y}}_j^h = P\left(\rho_{c_i}^h \in \mathcal{V}^h, \rho_{c_i}^{h-1} \in \mathcal{V}^{h-1}, \ldots, \rho_{c_i}^1 \in \mathcal{V}^1|\mathbf{p}\right)$$

$$= P\left(\rho_{c_i}^1 \in \mathcal{V}^1|\mathbf{p}\right) \prod_{k=2}^h P\left(\rho_{c_i}^k \in \mathcal{V}^k|\rho_{c_i}^{k-1} \in \mathcal{V}^{k-1}, \mathbf{p}\right)$$

$$= \prod_{k=2}^h P\left(\rho_{c_i}^k \in \mathcal{V}^k|\rho_{c_i}^{k-1} \in \mathcal{V}^{k-1}, \mathbf{p}\right).$$

Here, $P(\rho_{c_i}^1 \in \mathcal{V}^1 | \mathbf{p}) = 1$ because $\mathcal{V}^1$ only contains the root of $\mathcal{H}$. Then, we use a dense layer to calculate each conditional probability

$$P\left(\rho_{c_i}^k \in \mathcal{V}^k | \rho_{c_i}^{k-1} \in \mathcal{V}^{k-1}, \mathbf{p}\right) = \sigma(\mathbf{w}_k \mathbf{p})_{\rho_{c_i}^k}$$

where $\mathbf{w}_k \in \mathbb{R}^{n_k \times p}$ decodes the patient embedding $\mathbf{p}$ to the probability of the medical codes in the kth level, and n_k is the number of medical codes in the kth level.

Finally, the decoder with the hierarchical prediction is defined as follows:

$$\hat{\mathbf{y}}_j^h = \text{Decoder}(\mathcal{H}, \mathbf{p}) = \prod_{k=2}^h \sigma(\mathbf{w}_k \mathbf{p})_{\rho_{c_i}^k} \tag{11}$$

$$\mathcal{L} = \frac{1}{H-1} \sum_{h=2}^H \frac{1}{n_h} \text{CrossEntropy}\left(\hat{\mathbf{y}}^h, \mathbf{y}^h\right). \tag{12}$$

Here, $\mathbf{y}^h$ is the ground truth of ancestor medical codes in the hth level. $\mathbf{y}_j^h = 1$ means c_j is the ancestor $\rho_{c_i}^h$ of c_i in the hth level and $c_j \in \mathcal{V}^h$. Note that, as a self-supervised learning problem, this proxy task is not a simple input reconstruction. It predicts the hierarchy of a set of diagnoses given a sequence of admissions. Hence, it is different from G-BERT [15], which is a pretraining method and reconstructs the diagnoses and medicines given the same input.

E. Fine-Tuning and Inference

After self-supervised learning, we obtain the learned embedding vectors of medical codes and model parameters. For a specific task given the same format of input, we first calculate the patient embedding $\mathbf{p}$. Then, we use a fully connected layer for the real prediction task in Definition 2. Finally, we calculate the estimated output $\hat{\mathbf{y}}'$ and the fine-tuning loss $\mathcal{L}'$ to optimize the encoder model, including the medical code embeddings and model parameters

$$\hat{\mathbf{y}}' = g(\mathbf{W}\mathbf{p}) \in \mathbb{R}^o \tag{13}$$

$$\mathcal{L}' = \text{Loss}(\mathbf{y}', \hat{\mathbf{y}}', \Theta) \tag{14}$$

where $\mathbf{W} \in \mathbb{R}^{o \times p}$ is the weight matrix for the output and o represents the output size. g denotes the activation function. $\mathbf{y}'$ is the ground-truth label and Θ is the set of parameters of the model. Note that $\mathbf{y}'$, g, o, and the loss function depend on specific tasks. When optimizing the model with backpropagation, we still keep the embedding matrix and parameters learnable.

The pseudocode of Sherbet, including self-supervised learning and fine-tuning, is summarized in Algorithm 1. It first pretrains the hyperbolic embeddings of medical codes at line 1. Then, the Sherbet-Graph is constructed and a GNN is applied at lines 2–8. The self-supervised learning framework calculates the patient vector for each patient in $\mathcal{D}$, predicts the historical hierarchy of diagnoses, and optimizes the model at lines 9–17. Finally, fine-tuning is performed on specific tasks in lines 18–23.

F. Model Interpretability

Another advantage of Sherbet is its ability to provide interpretable prediction results and learned representations (which

---

Algorithm 1: Sherbet-Optimization($\mathcal{H}, \mathcal{D}$)

Input: The hierarchical structure $\mathcal{H}$ of medical codes, an EHR dataset $\mathcal{D}$

1   $\mathbf{E} \leftarrow$ HyperbolicEmbedding($\mathcal{H}$)
2   $\hat{\mathbf{A}} \leftarrow$ Construct graph $\mathcal{G}$ and calculate normalized adjacency matrix by Equation (8)
3   $L \leftarrow$ Graph layer number
4   $\mathbf{H}^{(0)} \leftarrow \mathbf{E}$
5   **for** $l \leftarrow 0$ *to* $L - 1$ **do**   // Graph learning
6     $\mathbf{H}^{(l+1)} \leftarrow$ Aggregate $\mathbf{H}^{(l)}$ by Equation (7)
7   **end**
8   $\mathbf{X} \leftarrow \mathbf{H}^{(L)}$
9   **repeat**   // Self-supervised learning
10     $u \leftarrow$ a patient in $\mathcal{D}$
11     $\mathbf{p} \leftarrow$ Encoder($u, \mathcal{D}, \mathbf{X}$)
12     $H \leftarrow$ level number of $\mathcal{H}$
13   **for** $h \leftarrow 2$ *to* H **do**
14       $\hat{\mathbf{y}}^h \leftarrow$ Predict level h's labels by Equation (11)
15   **end**
16   Optimizing the model using $\mathcal{L}$
17   **until** *convergence*
18   **repeat**   // Fine-tuning
19     $u \leftarrow$ a patient in $\mathcal{D}$ with multiple admissions
20     $\mathbf{p} \leftarrow$ Encoder($u, \mathcal{D}, \mathbf{X}$)
21     $\hat{\mathbf{y}}' \leftarrow$ Predict labels based on specific tasks
22   Optimizing the model using $\mathcal{L}'$
23   **until** *convergence*

---

is an important part of healthcare). Some existing models [3], [4] only focus on learning medical concept representations while neglecting personalized interpretability at the patient level. On the other hand, a few other models [20] are able to interpret at the patient level while lacking generic interpretability for common medical knowledge. In Sherbet, we provide generic interpretability by learning effective disease representations as well as personalized interpretability by quantifying contributions of each disease and admission for a patient.

Representation for Medical Codes: We use the output $\mathbf{X}$ of GNN as the hidden representation of medical codes since the proposed self-supervised graph learning framework helps to learn effective disease relations and captures reasonable disease complications as generic interpretability. We demonstrate the results of the learned representations in Section V-D.

Code-Level Contribution: We use the code-level attention scores $\boldsymbol{\alpha}_\tau = [\alpha_\tau^1, \alpha_\tau^2, \ldots, \alpha_\tau^n]$ to represent the contributions of medical codes in an admission since α is a probability distribution of medical codes.

Admission-Level Contribution: Similar to the code-level contribution, we use $\boldsymbol{\beta}$ as a part of the admission-level contribution. Furthermore, we want to quantify the contribution of a specific admission to the output. For example, if the task is to predict all diagnoses in the next admission, we want to comprehend the contribution of the t-th admission to each predicted medical code, that is, $\hat{\mathbf{y}}_i$. Therefore, we define a

coefficient $\boldsymbol{\delta} = [\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \ldots, \boldsymbol{\delta}_T]$ to measure this contribution

$$\boldsymbol{\delta}_\tau = \beta_\tau \frac{\exp(\mathbf{W}\boldsymbol{\theta}_\tau)}{\sum_{\tau=1}^{T} \exp(\mathbf{W}\boldsymbol{\theta}_\tau)} \in \mathbb{R}^o \qquad (15)$$

where $\mathbf{W}$ is the weight in (13). In this equation, first, $\mathbf{W}\boldsymbol{\theta}_\tau$ projects the τ-th admission weight into the output dimension. Then, it is multiplied by β_τ, so that we can measure the contribution from different admissions. Intuitively, $\boldsymbol{\delta}_\tau$ first calculates the contribution of each admission (β_τ), then it assigns this contribution to each dimension of the output ($\mathbf{W}\boldsymbol{\theta}_\tau$). Finally, the code-level and admission-level contributions can provide personalized interpretability for different patients.

IV. EXPERIMENTAL SETUP

A. Tasks and Evaluation Metrics

We conduct our experiments on two tasks following the settings of GRAM [3].
1) *Diagnosis Prediction:* This task predicts all medical codes in the next admission. It is a multilabel classification.
2) *Heart Failure Prediction:* This task predicts whether patients will be diagnosed with HF in the next admission. It is a binary classification.

The evaluation metrics for diagnosis prediction are weighted F_1 score (w-F_1) as in Timeline [4] and top-k recall (R@k) as in DoctorAI [1]. w-F_1 is a weighted sum of the F_1 score for each class, which measures an overall prediction performance on all classes. R@k is the ratio of true-positive numbers in top-k predictions to the total number of positive samples, which measures the prediction performance on a subset of classes.

The metrics to evaluate the HF prediction are the area under the ROC curve (AUC) and F_1 score since the HF prediction is a binary classification on imbalanced test data.

B. Datasets

We use the MIMIC-III dataset [16] in both tasks and the eICU dataset [41] in the first task to evaluate the performance of our model. MIMIC-III contains 58 976 de-identified admission records between 2001 and 2012 from 46 520 patients. In each record, the diseases are encoded by the ICD-9-CM system. There are 6981 medical codes in both single and multiple admissions. The eICU dataset records the patients' admissions to intensive care units (ICUs). For eICU, a patient may have multiple visits to hospitals, and in each hospital visit, there may be multiple admissions to ICU. However, there are no timestamps for hospital visits in the eICU dataset. Therefore, we regard one hospital visit as an independent patient, and each ICU admission as admission records V. In addition, we remove diseases in eICU that cannot be found in the ICD-9-CM or ICD-10 system. Table II shows the basic statistics of single and multiple admissions in the MIMIC-III and eICU datasets.

We randomly split the EHR data with multiple admissions into training/validation/test data, which contain 6000/493/1000 patients for MIMIC-III, and 8000/408/1000 patients for eICU. The Sherbet graph and self-supervised learning are constructed and trained with single admission data and training data

TABLE II
STATISTICS OF MIMIC-III AND eICU DATASETS FOR BOTH SINGLE (S) AND MULTIPLE (M) ADMISSIONS (ADM.)

Dataset	MIMIC-III		eICU	
	S	M	S	M
# patients	38,980	7,493	117,752	9,408
# adm.	38,980	19,894	117,752	20,209
Max. # adm.	1	42	1	7
Avg. # adm.	1	2.66	1	2.15
# codes	6,427	4,880	837	686
Max # codes per adm.	39	39	58	54
Avg # codes per adm.	10	13	3.47	4.40

with multiple admissions to guarantee there is no leakage of test data in fine-tuning. We use all medical codes for self-supervised learning on both tasks. For the diagnosis prediction task, we predict medical codes appearing in multiple admissions. For the HF prediction task, there are 38.5% positive samples and 61.5% negative samples in MIMIC-III.

C. Comparison Methods

We select the following state-of-the-art models to compare the performance with Sherbet.[1]
1) *MLP:* A deep neural network consists of three layers and uses multihot vectors for medical codes in an admission.
2) *DoctorAI [1]:* An RNN-GRU model, which also uses multihot vectors as inputs.
3) *RETAIN [19]:* A network of two RNNs with reverse time and attention methods. The inputs are multihot vectors for medical codes.
4) *Deepr [10]:* A CNN model that uses the embedding of medical codes as inputs.
5) *GRAM [3]:* An RNN model with a medical ontology graph. The inputs are medical code embeddings.
6) *Dipole [20]:* A bidirectional RNN model with attention. The inputs are multihot vectors.
7) *Timeline [4]:* An RNN model with attention, using the time duration information. The inputs are the embeddings of medical codes.
8) *G-BERT* [15]:* A BERT-based model with a medical ontology graph. It first pretrains the model on a self-prediction and dual-prediction task, then fine-tunes the model to predict the medicine. We modify G-BERT* by removing the medication module and changing the fine-tuning module to predict diagnoses and HF.
9) *HiTANet [22]:* A transformer-based model considering time intervals between admissions. The inputs are multihot vectors.

D. Parameter Settings for Sherbet

We randomly initialize all embeddings and model parameters. The embedding sizes d, m, and p for $\mathbf{E}, \mathbf{X}$, and $\mathbf{p}$ are 128, 64, and 64, respectively. We use one graph layer, where the hidden unit number is 64. The attention sizes a and b

---

[1]We do not compare with MiME [42], GCT [34], and MPVAA [43] since we do not use treatment, lab result, and demographic features in this work.

TABLE III
DIAGNOSIS PREDICTION RESULTS ON MIMIC-III AND eICU USING W-F_1 (%) AND R@k (%)

Models	MIMIC-III				eICU			
	w-F_1	R@10	R@20	# Params	w-F_1	R@10	R@20	# Params
MLP	11.68 (0.12)	26.01 (0.04)	26.99 (0.03)	1.38M	39.45 (0.08)	63.52 (0.12)	71.59 (0.09)	0.40M
DoctorAI	12.04 (0.04)	25.69 (0.11)	27.21 (0.08)	2.53M	64.13 (0.28)	77.08 (0.32)	81.79 (0.26)	0.50M
RETAIN	18.37 (0.79)	32.12 (0.82)	32.54 (0.62)	2.90M	**65.04 (1.03)**	78.68 (0.92)	83.48 (1.08)	0.85M
Deepr	11.68 (0.02)	26.47 (0.02)	27.53 (0.11)	1.16M	45.89 (0.21)	67.63 (0.15)	74.01 (0.18)	0.19M
GRAM	20.78 (0.14)	34.17 (0.19)	35.46 (0.26)	1.59M	57.95 (0.05)	75.67 (0.03)	81.52 (0.03)	0.41M
Dipole	14.66 (0.20)	28.73 (0.20)	29.44 (0.20)	2.18M	58.41 (0.18)	75.66 (0.24)	81.06 (0.19)	0.56M
Timeline	16.04 (0.72)	30.73 (0.79)	32.34 (0.90)	1.23M	57.04 (1.23)	74.40 (1.17)	79.86 (0.86)	0.30M
G-BERT*	22.28 (0.25)	35.62 (0.18)	36.46 (0.15)	5.63M	63.61 (0.33)	77.63 (0.44)	80.06 (0.39)	2.62M
HiTANet	21.15 (0.19)	34.68 (0.25)	35.97 (0.13)	3.33M	62.83 (0.38)	77.02 (0.41)	81.37 (0.46)	1.18M
Sherbet	**25.74 (0.04)**	**40.46 (0.08)**	**41.08 (0.08)**	1.23M	64.18 (0.08)	**80.05 (0.09)**	**84.43 (0.11)**	0.17M

for code-level and admission-level attention are 64 and 32, respectively. The weight φ on the adjacency matrix is 0.9. In addition, for fine-tuning, we add Dropout [44] on the graph layer and the input of the decoder. The graph layer's dropout rates of two tasks on MIMIC-III and diagnosis prediction on eICU are 0.2, 0.8, and 0.2, respectively. The decoder's dropout rates are 0.02, 0.15, and 0.15, respectively. For the fine-tuning of both tasks, the activation function is sigmoid; and the loss function is cross-entropy loss.

We use 500 epochs for pretraining hyperbolic embeddings. For self-supervised learning, we use 1000 and 300 epochs on the MIMIC-III dataset and the eICU dataset, respectively, and 200 epochs for fine-tuning with a learning rate decay strategy. The initial learning rate for hyperbolic embedding is 0.01 and decays by 0.01 every 100 epochs. The initial learning rate for self-supervised learning is 0.01 and decays at the (100, 500)th epoch by 0.1 on the MIMIC-III dataset, and decays at the (100, 250)th epoch by 0.1 on the eICU dataset. The initial learning rate for fine-tuning is 0.01 and decays at (20, 35, 100)th/(50, 60, 100)th epoch for diagnosis prediction on MIMIC-III/eICU, and decays at (25, 40, 45) epochs for HF prediction on the MIMIC-III dataset. We use the Adam optimizer [45] for hyperbolic embedding and RMSProp optimizer [46] for self-supervised learning and fine-tuning. The batch sizes for hyperbolic embedding, self-supervised learning, and fine-tuning are 256, 128, and 32, respectively.

All programs are implemented using Python 3.7.4 and Tensorflow 2.3.0 with CUDA 10.1 on a machine with Intel i9-9900K CPU, 64-GB memory, and Geforce RTX 2080 Ti GPU.[2]

E. Parameter Settings for Baselines

1) *MLP:* The hidden units for two hidden layers are 128 and 64, respectively.
2) *DoctorAI:* The hidden size for RNN is 128.
3) *RETAIN:* The embedding size for admissions is 256. The hidden layer size for two RNN layers is 128.
4) *Deepr:* The embedding size for medical codes is 100. The kernel size and filter number for a 1-D CNN layer are 3 and 128, respectively.

5) *GRAM:* The embedding size for medical codes and attention size are 100. The hidden size for RNN is 128.
6) *Dipole:* The embedding size for admissions is 256. The concatenation-based attention size is 128. The hidden size for RNN is 128.
7) *Timeline:* The embedding size for medical codes and attention size are 100. The hidden size for RNN is 128.
8) *G-BERT*:* The parameter settings are the same as given in [15]. a) *GAT part:* the input embedding size is 75, the number of attention heads is 4 and b) *BERT part:* the hidden size is 300. The positionwise feedforward networks include two hidden layers with four attention heads for each layer. The dimension of each hidden layer is 300.
9) *HiTANet:* The parameter settings are the same as [22]. a) HiTANet part: the dense space size for diseases is 256. The space size for a time interval, query vector size, and latent space size for time interval is 64 and b) Transformer part: the dimension of attention embedding is 64. The multihead number is 4. The size of the middle feedforward network is 1024.

V. EXPERIMENTAL RESULTS

A. Prediction Results

Table III shows the results of the weighted F_1 score and top k recall on the diagnosis prediction task using both datasets. We select $k = [10, 20]$ to calculate the R@k, because the average medical code numbers of an admission are 13 and 4.40 in MIMIC-III and eICU, respectively. In Table III, we can observe that Sherbet outperforms the baseline models in most cases. Note that the results of models on eICU are much better than MIMIC-III. This is primarily due to the fact that eICU has only 686 medical codes, while MIMIC-III has 4880. The comparison among Sherbet, G-BERT* and GRAM indicates that self-supervised learning significantly improves the prediction results on both datasets since they all use medical ontology information.

Table IV demonstrates the AUC and F_1 scores of the results for the HF prediction task. We use only MIMIC-III because some diseases are removed in the eICU dataset. Similar to the diagnosis prediction task, Sherbet obtains the superior performance in terms of AUC and F_1. In this task, compared

---

[2]The source code of the Sherbet model can be found at https://github.com/LuChang-CS/sherbet.

TABLE IV
HF PREDICTION RESULTS ON MIMIC-III USING AUC (%) AND F_1 (%)

Models	AUC	F_1	# Params
MLP	80.09 (0.06)	68.17 (0.05)	0.64M
DoctorAI	82.25 (0.02)	68.42 (0.02)	1.91M
RETAIN	82.73 (0.21)	71.12 (0.37)	1.67M
Deepr	81.29 (0.01)	68.42 (0.01)	0.53M
GRAM	82.82 (0.06)	71.43 (0.05)	0.96M
Dipole	81.66 (0.07)	70.01 (0.04)	0.92M
Timeline	80.75 (0.46)	69.81 (0.34)	0.95M
G-BERT*	83.61 (0.18)	72.37 (0.46)	2.69M
HiTANet	82.77 (0.35)	71.93 (0.29)	2.08M
Sherbet	**86.04 (0.16)**	**74.27 (0.07)**	0.91M

TABLE V
PREDICTION RESULTS OF THE VARIANTS OF SHERBET

Model	Diagnosis				HF	
	MIMIC-III		eICU			
	w-F_1	R@10	w-F_1	R@10	AUC	F_1
Sherbet	25.74	40.46	64.18	80.05	86.04	74.27
Sherbet$_b$	25.46	39.54	62.75	78.86	85.20	72.28
Sherbet$_c$	25.21	39.21	62.52	78.51	84.47	71.58
Sherbet$_d$	23.92	37.91	58.75	77.58	83.18	71.36
Sherbet$_e$	22.12	36.74	58.45	77.21	82.35	70.67
Sherbet$_f$	25.37	39.28	63.01	78.62	84.51	73.26
Sherbet$_g$	23.18	37.05	58.26	77.39	82.80	71.02

to the best baseline model, the improvement of Sherbet is not as significant as the diagnosis prediction. We infer that self-supervised learning is able to learn a general knowledge representation of diseases, while HF is only one of the many diseases. On the contrary, as a multilabel classification task, the diagnosis prediction task can fully utilize the representation learned by self-supervised learning and distinguish each type of disease in the output.

In summary, the prediction results of diagnoses and HF demonstrate the superior performance of the proposed framework over state-of-the-art models.

B. Ablation Study

To study the effectiveness of each module in Sherbet, we also compare five types of Sherbet with some modules removed or replaced.

1) *Sherbet$_b$:* Sherbet with self-supervised learning and hierarchical prediction, removing hyperbolic embedding.
2) *Sherbet$_c$:* Sherbet with self-supervised learning, removing hyperbolic embedding and hierarchical prediction.
3) *Sherbet$_d$:* Sherbet with hyperbolic embedding, removing self-supervised learning while only using fine-tuning with multiple admission data.
4) *Sherbet$_e$:* Sherbet removing hyperbolic embedding and self-supervised learning while only using fine-tuning with multiple admission data.
5) *Sherbet$_f$:* Sherbet with the multilevel attention encoder replaced by T-LSTM [47].
6) *Sherbet$_g$:* Sherbet removing the graph structure and GNN by directly using the learned hyperbolic embedding for multilevel attention.
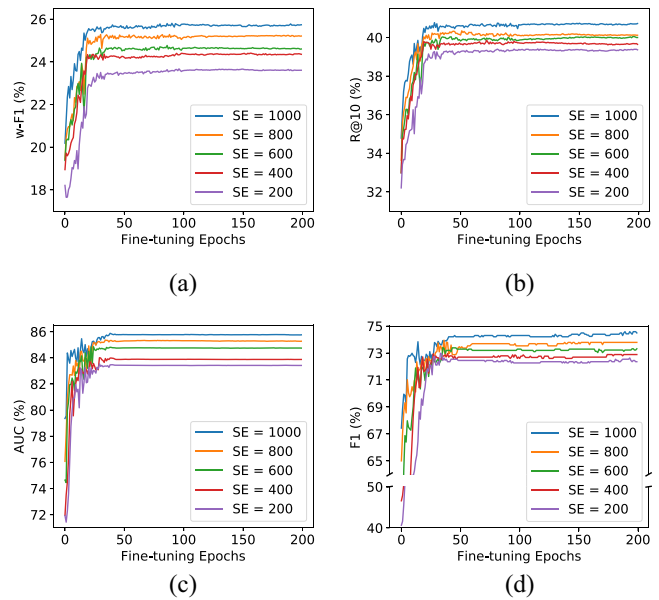

Fig. 4. Prediction results for fine-tuning using w-F_1, R@10 on the diagnosis prediction, and AUC, F_1 on the HF prediction with five different SE. (a) w-F_1. (b) R@10. (c) AUC. (d) F_1.

We report the results of Sherbet and Sherbet$_{b\sim g}$ on two tasks in Table V. We can first observe that the complete Sherbet achieves the best performance among Sherbet$_{b\sim g}$, which proves the effectiveness of our proposed model. The comparison between Sherbet and Sherbet$_b$, and Sherbet$_d$ and Sherbet$_e$ shows that the hyperbolic embedding can effectively initialize the representations of medical codes. The results of Sherbet$_b$ and Sherbet$_c$ indicate that the hierarchical structure of medical codes is helpful to guide the prediction. We can also see that the performance of Sherbet$_c$ has a large improvement over Sherbet$_e$, which shows the importance of self-supervised learning. However, the performance improvement of Sherbet$_c$ over Sherbet$_d$ on HF prediction is not as significant as the improvement over Sherbet$_e$. We infer that the hyperbolic embedding is also effective without self-supervised learning when the number of predicted medical codes is small. In addition, after replacing the multilevel attention with T-LSTM, the performance of Sherbet$_f$ is not as good as Sherbet. Moreover, the result of Sherbet$_g$ shows that the proposed Sherbet graph and GNN are also important in predicting health events because the graph structure was utilized to capture hidden relations of diseases. Finally, even without self-supervised learning and hyperbolic embedding (Sherbet$_e$), our model still achieves top performance among baselines as shown in Tables III and IV, which demonstrates the effectiveness of the proposed framework.

C. Self-Supervised Learning Study

To further study the effectiveness and impact of self-supervised learning, we adopt different self-supervised learning epochs (SE). On both tasks, we select five different SE as [200, 400, 600, 800, 1000] to report the variation of w-F_1, R@10 of the diagnosis prediction, and AUC and F_1 of the HF

prediction on the validation data of both tasks in MIMIC-III during the fine-tuning phase.

Fig. 4 shows the validation results for fine-tuning of the diagnosis prediction and the HF prediction on different SE. The *x*-axis of each subplot denotes the fine-tuning epochs, and the *y*-axis means each evaluation metrics for fine-tuning. In Fig. 4(a) and (b) of the diagnosis prediction results, we can see that SE has a significant influence on the fine-tuning results. Larger SE not only produces higher w-F_1 and R@10 but also speeds up the convergence. As shown in Fig. 4(c) and (d) of the HF prediction, the final values of AUC and F_1 are not affected too much by SE compared to the diagnosis prediction. However, lower SE still gives worse results. Therefore, we can further conclude that self-supervised learning on the historical hierarchy prediction task helps in improving the prediction performance on different tasks and accelerates the fine-tuning of the model.

D. Case Studies for Model Interpretation

We interpret Sherbet by demonstrating the representation of some typical disease complications learned by Sherbet on MIMIC-III and visualizing the contributions introduced in Section III-F.

1) Representation of Disease Complication: In order to demonstrate if Sherbet successfully captures the disease complications, we adopt the embedding matrix **X**, that is, the output of the GNN, as the final representation learned by self-supervised learning. Then, we use *t*-SNE [48] to project **X** into two dimensions. In the next step, we select 15 types of HF,[3] three types of essential hypertension,[4] and seven types of acute rheumatic fever[5] that appear in MIMIC-III and plot them in Fig. 5(a). As shown in Fig. 5(a), they are mainly grouped into three clusters. In addition, hypertension locates near one cluster of HF, and acute rheumatic fever locates near the three clusters of HF. Given that hypertension and acute rheumatic fever are two common comorbidities of heart-related diseases, they are often diagnosed in the same set of patients. It indicates that the proposed method learns similar embeddings for similar diseases or complications. Next, we select two types of diabetes[6]: 1) type I and 2) type II, and plot their embeddings in Fig. 5(b). As shown in Fig. 5(b), there are no joint clusters between diabetes type I and diabetes type II. We infer that the complications of diabetes type I and diabetes type II are different to some extent. To summarize, Sherbet is able to capture the disease complications. It can also distinguish different types of diseases based on their different complications. Therefore, Sherbet can provide the general interpretability for diseases using the learned hidden representations of medical codes.

2) Quantification of Multilevel Attention: To measure the contributions both at the code level and at the admission level explained in Section III-F, we visualize the code-level attention distribution α in code-level attention, and coefficient δ
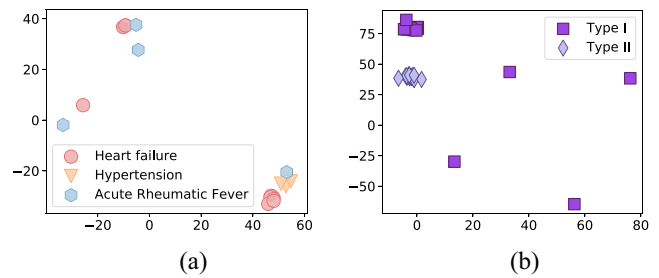
[3]www.icd9data.com/2015/Volume1/390-459/420-429/428/
[4]www.icd9data.com/2015/Volume1/390-459/401-405/401/
[5]www.icd9data.com/2015/Volume1/390-459/390-392/
[6]www.icd9data.com/2015/Volume1/240-279/249-259/250/

Fig. 5. Scatter plot of heart related conditions and different diabetes using t-SNE. (a) Complications between different types of HF, hypertension, and acute rheumatic fever (which is a disease that can affect the heart). (b) Diabetes type I and type II.
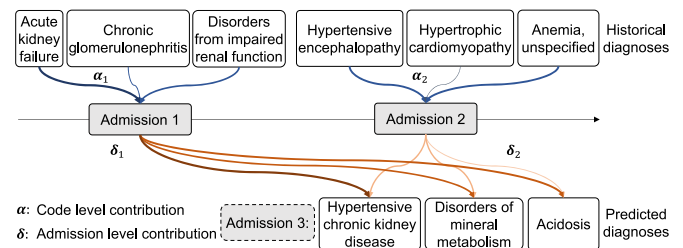

Fig. 6. Contribution of diagnoses to each admission, and admissions to each predicted diagnosis. In this case, the patient has two historical admissions. We use two admissions to predict diagnoses in the third admission.

in (15) for a given patient *u* on the diagnosis prediction task. Fig. 6 demonstrates the historical diagnoses, the admissions, and the predicted diagnoses for a patient in rectangles. The contributions α and δ are represented by blue and red lines, respectively. The thickness and darkness of the line denote different values of α and δ. Thicker and darker lines correspond to larger values of α and δ. In this case, there are six diagnoses in the first admission and ten diagnoses in the second admission, and we select three important diagnoses for both admissions. There are also six diagnoses of the third admission of this patient, that is, ground-truth diagnoses, and we show three correct predictions in Fig. 6.

In the first admission, the three diagnoses are all kidney-related diseases. Acute kidney failure and disorders resulting from impaired renal function mainly contribute to this admission. In the second admission, the three diagnoses belong to brain, heart, and blood-related diseases, respectively. Hypertensive encephalopathy and Anemia mainly contribute to this admission. In the third admission, we can see that the correct predictions are all metabolic diseases and kidney disease. Therefore, Sherbet predicts that the first admission contributes more than the second admission. More specifically, the first admission contributes mainly to hypertensive chronic kidney disease because the patient is diagnosed with acute kidney failure in the first admission. We can also infer that the kidney problems of this patient mainly cause the metabolic diseases in the third admission. For the second admission, we can discover that it mainly contributes to hypertensive chronic kidney disease. We believe that it is due to the fact this patient has hypertensive encephalopathy. High blood pressure causing hypertensive encephalopathy also causes hypertensive chronic

kidney disease in this admission. Besides, anemia in the second admission can also be related to the metabolic diseases in the third admission. In summary, Sherbet provides quantitative and personalized interpretability using the contributions α and δ learned from patients' historical diagnoses and admissions.

VI. Conclusion

In this article, we proposed Sherbet, a self-supervised graph learning framework with hyperbolic embeddings for medical codes to predict temporal health events. We first took advantage of the hierarchical structure of medical codes to pretrain a hyperbolic embedding for diseases. Then, we adopted a GNN with a weighted and directed graph of medical codes to learn disease complications in EHR data. With the specially designed code-level and admission-level attention mechanism, Sherbet is able to simultaneously provide generic interpretability for medical concepts and personalized interpretability for patients. In addition, we also designed a self-supervised proxy task to predict the historical hierarchy of diagnoses in patients' admission records by further utilizing the hierarchical structure of medical codes. This task is able to leverage more data by incorporating single admission records and the final admissions of multiple admission records. Our experimental results showed the improved performance of Sherbet over state-of-the-art methods. We also illustrated the generic and personalized interpretability of Sherbet using case studies. One shortcoming of Sherbet is that it only utilizes disease codes in patients' admission records. In the future, we will explore the relationships of more features in EHR data, such as procedures, medicines, and lab results.

References

[1] E. Choi, M. T. Bahadori, A. Schuetz, W. F. Stewart, and J. Sun, "Doctor AI: Predicting clinical events via recurrent neural networks," in *Proc. Mach. Learn. Healthcare Conf.*, 2016, pp. 301–318.

[2] N. Zheng *et al.*, "Predicting COVID-19 in China using hybrid AI model," *IEEE Trans. Cybern.*, vol. 50, no. 7, pp. 2891–2904, Jul. 2020.

[3] E. Choi, M. T. Bahadori, L. Song, W. F. Stewart, and J. Sun, "GRAM: Graph-based attention model for healthcare representation learning," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2017, pp. 787–795.

[4] T. Bai, S. Zhang, B. L. Egleston, and S. Vucetic, "Interpretable representation learning for healthcare via capturing disease progression through time," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2018, pp. 43–51.

[5] Z. Xu, J. Zhang, Q. Zhang, Q. Xuan, and P. S. F. Yip, "A comorbidity knowledge-aware model for disease prognostic prediction," *IEEE Trans. Cybern.*, early access, May 7, 2021, doi: 10.1109/TCYB.2021.3070227.

[6] D. Nguyen, W. Luo, S. Venkatesh, and D. Phung, "Effective identification of similar patients through sequential matching over ICD code embedding," *J. Med. Syst.*, vol. 42, no. 5, p. 94, 2018.

[7] B. Yang, M. Ye, Q. Tan, and P. C. Yuen, "Cross-domain missingness-aware time-series adaptation with similarity distillation in medical applications," *IEEE Trans. Cybern.*, early access, Aug. 14, 2020, doi: 10.1109/TCYB.2020.3011934.

[8] S. Darabi, M. Kachuee, S. Fazeli, and M. Sarrafzadeh, "TAPER: Time-aware patient EHR representation," *IEEE J. Biomed. Health Inform.*, vol. 24, no. 11, pp. 3268–3275, Nov. 2020.

[9] A. W. Mulyadi, E. Jun, and H.-I. Suk, "Uncertainty-aware variational-recurrent imputation network for clinical time series," *IEEE Trans. Cybern.*, early access, Mar. 4, 2017, doi: 10.1109/TCYB.2021.3053599.

[10] P. Nguyen, T. Tran, N. Wickramasinghe, and S. Venkatesh, "`Deepr`: A convolutional net for medical records," *IEEE J. Biomed. Health Inform.*, vol. 21, no. 1, pp. 22–30, Jan. 2017.

[11] Z. Huang, W. Dong, H. Duan, and J. Liu, "A regularized deep learning approach for clinical risk prediction of acute coronary syndrome using electronic health records," *IEEE Trans. Biomed. Eng.*, vol. 65, no. 5, pp. 956–968, May 2018.

[12] 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 *Proc. IEEE Int. Conf. Data Min. (ICDM)*, 2017, pp. 787–792.

[13] C. Wang, N. Lu, Y. Cheng, and B. Jiang, "A data-driven aero-engine degradation prognostic strategy," *IEEE Trans. Cybern.*, vol. 51, no. 3, pp. 1531–1541, Mar. 2021.

[14] L. Yao, Y. Zhang, B. Wei, W. Zhang, and Z. Jin, "A topic modeling approach for traditional Chinese medicine prescriptions," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 6, pp. 1007–1021, Jun. 2018.

[15] J. Shang, T. Ma, C. Xiao, and J. Sun, "Pre-training of graph augmented transformers for medication recommendation," in *Proc. 28th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2019, pp. 5953–5959.

[16] A. E. Johnson *et al.*, "MIMIC-III, a freely accessible critical care database," *Sci. Data*, vol. 3, 2016, Art. no. 160035.

[17] CDC. (Nov. 2015). *ICD-9-CM—International Classification of Diseases, Ninth Revision, Clinical Modification*. Accessed: Oct. 5, 2020. [Online]. Available: https://www.cdc.gov/nchs/icd/icd9cm.htm

[18] F. H. Messerli, S. F. Rimoldi, and S. Bangalore, "The transition from hypertension to heart failure: Contemporary update," *JACC Heart Failure*, vol. 5, no. 8, pp. 543–551, 2017.

[19] 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 *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3504–3512.

[20] 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 *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2017, pp. 1903–1911.

[21] K. Cho *et al.*, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Nat. Lang. Process. (EMNLP)*, 2014, pp. 1724–1734.

[22] J. Luo, M. Ye, C. Xiao, and F. Ma, "HiTaNet: Hierarchical time-aware attention networks for risk prediction on electronic health records," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2020, pp. 647–656.

[23] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, May 4, 2020, doi: 10.1109/TPAMI.2020.2992393.

[24] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–9. [Online]. Available: https://openreview.net/forum?id=S1v4N2l0-

[25] X. Xu, H. Lu, J. Song, Y. Yang, H. T. Shen, and X. Li, "Ternary adversarial networks with self-supervision for zero-shot cross-modal retrieval," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2400–2413, Jun. 2020.

[26] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Assoc. Comput. Linguist. Human Lang. Technol.*, Jun. 2019, pp. 4171–4186.

[28] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *Proc. Conf. Empirical Methods Nat. Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.

[29] J. Yu, D. Tao, and M. Wang, "Adaptive hypergraph learning and its application in image classification," *IEEE Trans. Image Process.*, vol. 21, no. 7, pp. 3262–3272, Jul. 2012.

[30] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017, pp. 1–6.

[31] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6338–6347.

[32] I. Chami, A. Wolf, D.-C. Juan, F. Sala, S. Ravi, and C. Ré, "Low-dimensional hyperbolic knowledge graph embeddings," 2020. [Online]. Available: arXiv:2005.00545.

[33] N. Choudhary, N. Rao, S. Katariya, K. Subbian, and C. K. Reddy, "Self-supervised hyperboloid representations from logical queries over knowledge graphs," in *Proc. Web Conf.*, 2021, pp. 1373–1384.

[34] E. Choi *et al.*, "Learning the graphical structure of electronic health records with graph convolutional transformer," in *Proc. 34th Conf. Assoc. Adv. Artif. Intell.*, 2020, pp. 606–613.

[35] C. Lu, C. K. Reddy, P. Chakraborty, S. Kleinberg, and Y. Ning, "Collaborative graph learning with auxiliary text for temporal event prediction in healthcare," in *Proc. 13th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2021, pp. 3529–3535.

[36] CDC. (Feb. 2020). *ICD-10—International Classification of Diseases, Tenth Revision*. Accessed: May 10, 2020. [Online]. Available: https://www.cdc.gov/nchs/icd/icd10cm.htm

[37] B. Dhingra, C. Shallue, M. Norouzi, A. Dai, and G. Dahl, "Embedding text in hyperbolic spaces," in *Proc. 12th Workshop Graph Based Methods Nat. Lang. Process. (TextGraphs)*, 2018, pp. 59–69.

[38] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proc. Conf. Empirical Methods Nat. Lang. Process.*, Sep. 2015, pp. 1412–1421.

[39] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," 2015. [Online]. Available: arXiv:1505.00853.

[40] C. M. Bishop, *Pattern Recognition and Machine Learning*. Heidelberg, Germany: Springer, 2006, pp. 359–361.

[41] T. J. Pollard, A. E. Johnson, J. D. Raffa, L. A. Celi, R. G. Mark, and O. Badawi, "The eICU collaborative research database, a freely available multi-center database for critical care research," *Sci. Data*, vol. 5, Sep. 2018, Art. no. 180178.

[42] E. Choi, C. Xiao, W. Stewart, and J. Sun, "MIME: Multilevel medical embedding of electronic health records for predictive healthcare," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 4547–4557.

[43] S. Chowdhury, C. Zhang, P. S. Yu, and Y. Luo, "Mixed pooling multi-view attention autoencoder for representation learning in healthcare," 2019. [Online]. Available: arXiv:1910.06456.

[44] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[45] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, May 2015, pp. 1–6.

[46] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6A overview of mini-batch gradient descent," *Cited On*, vol. 14, no. 8, p. 2, 2012.

[47] I. M. Baytas, C. Xiao, X. Zhang, F. Wang, A. K. Jain, and J. Zhou, "Patient subtyping via time-aware lstm networks," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2017, pp. 65–74.

[48] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

Chandan K. Reddy (Senior Member, IEEE) received the M.S. degree from Michigan State University, East Lansing, MI, USA, and the Ph.D. degree from Cornell University, Ithaca, NY, USA, in 2003 and 2007, respectively.

He is a Professor with the Department of Computer Science, Virginia Tech, Arlington, VA, USA. He has published over 130 peer-reviewed articles in leading conferences and journals. His primary research interests are machine learning with applications to various real-world domains, including healthcare, transportation, social networks, and e-commerce.

Prof. Reddy received the several awards for his research work, including the Best Application Paper Award at ACM SIGKDD Conference in 2010, the Best Poster Award at IEEE VAST conference in 2014, the Best Student Paper Award at IEEE ICDM Conference in 2016, and was a finalist of the INFORMS Franz Edelman Award Competition in 2011. He is serving on the editorial boards of *ACM Transactions on Knowledge Discovery From Data*, IEEE BIG DATA, and *ACM Transactions on Intelligent Systems and Technology*. He is a Distinguished Member of the ACM.

Chang Lu received the B.S. and M.S. degrees from the School of Computer Science, Fudan University, Shanghai, China, in 2016 and 2019, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science, Stevens Institute of Technology, Hoboken, NJ, USA, advised by Dr. Y. Ning.

His research interests include big data, deep learning, and interpretable predictive models in healthcare domain.

Yue Ning (Member, IEEE) received the Ph.D. degree in computer science from Virginia Tech, Arlington, VA, USA, in 2018.

She is an Assistant Professor with the Department of Computer Science, Stevens Institute of Technology, Hoboken, NJ, USA, where she is also affiliated with the Stevens Institute for Artificial Intelligence. Specifically, she is focused on developing predictive methods to capture spatiotemporal, dynamic, and interpretable patterns in large-scale data with applications in computational social science and health informatics. She has published over 20 peer-reviewed papers in prestigious journals and conferences, including *ACM Transactions on Knowledge Discovery From Data*, KDD, AAAI, CIKM, ICWSM, and SDM. Her research is supported by Stevens Institute of Technology, Nvidia, Santa Clara, CA, USA, and the National Science Foundation, Alexandria, VA, USA. Her research interests are in the general areas of machine learning, data analytics, and social media analysis.

Dr. Ning has served on program committees of KDD, AAAI, SDM, ICML, IEEE BigData, ASONAM, and ICLR, and as a journal reviewer for *ACM Transactions on Knowledge Discovery From Data*, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, *Data Mining and Knowledge Discovery*, and *Pattern Recognition*.