`GraphCare`: Enhancing Healthcare Predictions with Open-World Personalized Knowledge Graphs

Pengcheng Jiang[*] **Cao Xiao**[†] **Adam Cross**[‡] **Jimeng Sun**[*]
[*] University of Illinois Urbana-Champaign [†] Relativity [‡] OSF HealthCare
`{pj20, jimeng}@illinois.edu, danicaxiao@gmail.com`
`adam.r.cross@osfhealthcare.org`

Abstract

Clinical predictive models often rely on patients' electronic health records (EHR), but integrating medical knowledge to enhance predictions and decision-making is challenging. This is because personalized predictions require personalized knowledge graphs (KGs), which are difficult to generate from patient EHR data. To address this, we propose `GraphCare`, an open-world framework that leverages external KGs to improve EHR-based predictions. Our method extracts knowledge from large language models (LLMs) and external biomedical KGs to generate patient-specific KGs, which are then used to train our proposed Bi-attention AugmenTed (`BAT`) graph neural network (GNN) for healthcare predictions. We evaluate `GraphCare` on two public datasets: MIMIC-III and MIMIC-IV. Our method outperforms baseline models in four vital healthcare prediction tasks: mortality, readmission, length-of-stay, and drug recommendation, improving AUROC on MIMIC-III by average margins of 10.4%, 3.8%, 2.0%, and 1.5%, respectively. Notably, `GraphCare` demonstrates a substantial edge in scenarios with limited data availability. Our findings highlight the potential of using external KGs in healthcare prediction tasks and demonstrate the promise of `GraphCare` in generating personalized KGs for promoting personalized medicine.

1 Introduction

The digitization of healthcare systems has led to the accumulation of vast amounts of electronic health record (EHR) data that encode valuable information about patients, treatments, etc. Machine learning models have been developed on these data and demonstrated huge potential for enhancing patient care and resource allocation via predictive tasks, including mortality prediction [1, 2], length-of-stay estimation [3, 4], readmission prediction [5, 6], and drug recommendations [7, 8].

To improve predictive performance and integrate expert knowledge with data insights, clinical knowledge graphs (KGs) were adopted to complement EHR modeling [9, 10, 11]. These KGs represent medical concepts (e.g., diagnoses, procedures, drugs) and their relationships, enabling effective learning of patterns and dependencies. However, existing approaches mainly focus on simple hierarchical relations [12, 13, 10] rather than leveraging comprehensive relationships among biomedical entities despite incorporating valuable contextual information from established biomedical knowledge bases (e.g., UMLS [14]) could enhance predictions. Moreover, large language models (LLMs) such as GPT [15, 16, 17, 18] pre-trained on web-scale biomedical literature could serve as alternative resources for extracting clinical knowledge given their remarkable reasoning abilities on open-world data. There is a substantial body of existing research demonstrating their potential use as knowledge bases [19, 20, 21].

To fill the gap in personalized medical KGs, we propose to leverage the exceptional reasoning abilities of LLMs to extract and integrate personalized KG from open-world data. Our proposed method

Preprint. Under review.

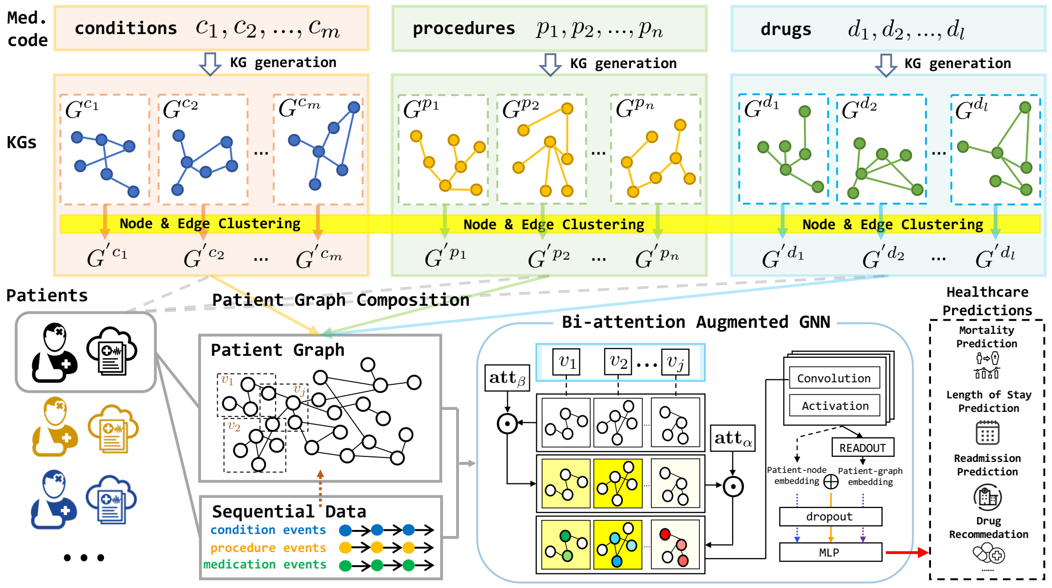


Figure 1: **Overview of** GraphCare. ***Above***: Given a patient record consisting of conditions, procedures and medications, we generate a concept-specific KG for each medical code, either by relational knowledge prompting from a large language model or by subgraph sampling from an existing KG; and we perform node and edge clustering among all graphs (§3.1). ***Below***: For each patient, we compose a patient-specific graph by combining the medical concept KGs associated with them and attach the sequential data labeling the temporal information across visits (§3.2). To utilize the patient graph for predictions, we employ a bi-attention augmented graph neural network (GNN) model, which highlights essential visits and nodes with attention weights (§3.3). With three types of patient representations (patient-node, patient-graph, and joint embeddings), GraphCare is able to handle a variety of healthcare predictions (§3.4).

GraphCare (Personalized **Graph**-based Health**Care** Prediction) is a framework designed to generate patient-specific KGs by effectively harnessing the wealth of clinical knowledge. As shown in Figure 1, our patient KG generation module first takes medical codes as input and generates code-wise KGs by prompting LLMs or retrieving subgraphs from existing graphs. It then clusters nodes and edges to create a more aggregated KG for each medical code. Next, it constructs a personalized KG for each patient by merging their associated medical code KGs and incorporating temporal information from their sequential visit data. These patient-specific graphs are then fed into our **Bi**-attention **A**ugmen**T**ed (BAT) graph neural network (GNN) for diverse downstream prediction tasks.

We evaluated the effectiveness of GraphCare using two widely-used EHR datasets, MIMIC-III [22] and MIMIC-IV [23]. Through extensive experimentation, we found that GraphCare outperforms several baselines, while BAT outperforms state-of-the-art GNN models [24, 25] on four common healthcare prediction tasks: mortality prediction, readmission prediction, LOS prediction, and drug recommendation. Our experimental results demonstrate that GraphCare, equipped with the BAT, achieves average AUROC improvements of 10.4%, 3.8%, 2.0%, 1.5% and 4.3%, 1.3%, 1.2%, 1.5% over all baselines on MIMIC-III and MIMIC-IV, respectively. Furthermore, our approach requires significantly fewer patient records to achieve comparable results, providing compelling evidence for the benefits of integrating open-world knowledge into healthcare predictions.

2 Related Works

Clinical Predictive Models. EHR data has become increasingly recognized as a valuable resource in the medical domain, with numerous predictive tasks utilizing this data [5, 7, 1, 3]. A multitude of deep learning models have been designed to cater to this specific type of data, leveraging its rich, structured nature to achieve enhanced performance [26, 27, 28, 29, 30, 8, 31, 10, 32, 33, 34, 35, 36]. Among these models, some employ a graph structure to improve prediction accuracy, effectively capturing underlying relationships among medical entities [10, 37, 38, 39, 40, 41, 36, 8]. However, a limitation of these existing works is that they do not link the local graph to an external knowledge base, which contains a large amount of valuable relational information [42, 43]. We propose to create a customized knowledge graph for each medical concept in an open-world setting by probing relational knowledge from either LLMs or KGs, enhancing its predictive capabilities for healthcare.

Personalized Knowledge Graphs. Personalized KGs have emerged as promising tools for enhancing clinical decision-making and healthcare predictions [44, 45, 46, 47, 48]. Previous approaches such as GRAM [12] and its successors [49, 50, 51, 52] incorporated external hierarchical graphs to improve predictions of deep learning-based models; however, they primarily focus on simple parent-child relationships, overlooking the rich complexities found in large knowledge bases. MedML [53] employs graph data for COVID-19 related prediction. However, the KG in this work has a limited scope and relies heavily on curated features. To address the aforementioned gaps, we propose two solutions for automatically constructing comprehensive personalized KGs from open sources: utilizing prompting techniques [54] derived from LLMs to generate KGs tailored to healthcare predictive tasks, or employing random sampling from existing KGs [14] to ensure a diverse and representative knowledge base.

Attention-augmented GNNs. Attention mechanisms [55] have been widely utilized in GNNs to capture the most relevant information from the graph structure for various tasks [24, 56, 57, 58, 59, 60]. The incorporation of attention mechanisms in GNNs allows for enhanced graph representation learning, which is particularly useful in the context of EHR data analysis [10]. In the `GraphCare` framework, we introduce a new GNN `BAT` which leverages both visit-level and node-level attention, along with edge weights, for predictions with personalized KGs in the healthcare domain.

3 Personalized Graph-based HealthCare Prediction (`GraphCare`)

In this section, we present `GraphCare`, a comprehensive framework designed to generate personalized KGs and utilize them for enhanced healthcare predictions.

Overview. The `GraphCare` framework operates through three general steps.

Step 1: we generate a concept-specific KG for each medical code in the dataset, either by prompting LLMs or by subsampling from existing KGs. Once gathered, we conduct node and edge clustering across these concept-specific KGs.

Step 2: we construct a personalized KG for each patient. This is done by merging the concept-specific KGs associated with their individual medical codes.

Step 3: we apply our novel Bi-attention Augmented (`BAT`) Graph Neural Network (GNN) to make predictions based on the personalized KGs.

3.1 Step 1: Concept-Specific Knowledge Graph Generation.

To create a KG for each unique medical code in the provided coding system, we employ one of two approaches: (1) We extract knowledge from LLMs using a prompting method, or (2) We sample a subgraph from an existing biomedical KG. Denote a medical code as $e \in \{\mathbf{c}, \mathbf{p}, \mathbf{d}\}$, where $\mathbf{c} = (c_1, c_2, ..., c_{|\mathbf{c}|})$, $\mathbf{p} = (p_1, p_2, ..., p_{|\mathbf{p}|})$, and $\mathbf{d} = (d_1, d_2, ..., d_{|\mathbf{d}|})$ correspond to sets of medical codes for conditions, procedures, and drugs, with sizes of $|\mathbf{c}|$, $|\mathbf{p}|$, and $|\mathbf{d}|$, respectively. For each medical code e, we extract a knowledge graph $G^e = (\mathcal{V}^e, \mathcal{E}^e)$, where $\mathcal{V}^e$ represents nodes, and $\mathcal{E}^e$ denotes edges in the graph.

(1) LLM-based KG extraction via prompting involves a template with three components: instruction, example, and prompt. For example, with an instruction "Given a prompt, extrapolate as many relationships as possible of it and provide a list of updates", an example "prompt: systemic lupus erythematosus. updates: [systemic lupus erythematosus is, treated with, steroids]..." and a prompt "prompt: tuberculosis updates:<place holder>", the LLM would respond with a list of KG triples such as "[tuberculosis, may be treated with, antibiotics], [tuberculosis, affects, lungs]..." where each triple contains a head entity, a relation, and a tail entity. We showcase our carefully designed prompts in Appendix C.1. For each medical code, we run χ times prompting and aggregate multiple outputs into one in order to compose a more comprehensive KG for each code, which is then parsed to construct $G^e_{\mathrm{LLM}(\chi)} = (\mathcal{V}^e_{\mathrm{LLM}(\chi)}, \mathcal{E}^e_{\mathrm{LLM}(\chi)})$.

(2) Subgraph sampling from existing KGs. To leverage existing biomedical knowledge graphs [61, 14, 62], we also extract the concept-specific graph for a medical code through subgraph sampling, which involves selecting a subset of nodes and edges from the original KG relevant to the medical code. To perform subgraph sampling, we first identify the entity in the existing biomedical KG corresponding to the medical code e. Then, we randomly sample a κ-hop subgraph originating from the entity. We describe more details of this in C.2. The output is $G^e_{\text{sub}(\kappa)} = (\mathcal{V}^e_{\text{sub}(\kappa)}, \mathcal{E}^e_{\text{sub}(\kappa)})$.

Node and Edge Clustering. Next, we perform node and edge clustering based on the similarity of their word embeddings, which are retrieved from LLMs. The similarity between two nodes or edges is computed using the cosine similarity between their word embeddings. We apply the agglomerative clustering algorithm [63] on the consine similarity with a distance threshold δ, to group similar nodes and edges in the global graph $G = (G^{e_1}, G^{e_2}, ..., G^{e_{(|\mathbf{c}|+|\mathbf{p}|+|\mathbf{d}|)}})$ of all concepts. After the clustering process, we obtain $\mathcal{C}_{\mathcal{V}} : \mathcal{V} \to \mathcal{V}^{'}$ and $\mathcal{C}_{\mathcal{E}} : \mathcal{E} \to \mathcal{E}^{'}$ which map the nodes $\mathcal{V}$ and edges $\mathcal{E}$ in the original graph G to new nodes $\mathcal{V}^{'}$ and edges $\mathcal{E}^{'}$, respectively. With these two mappings, we obtain a new global graph $G^{'} = (\mathcal{V}^{'}, \mathcal{E}^{'})$, and we create a new graph $G^{'e} = (\mathcal{V}^{'e}, \mathcal{E}^{'e}) \subset G^{'}$ for each medical code. The node embedding $\mathbf{H}^{\mathcal{V}} \in \mathbb{R}^{|\mathcal{C}_{\mathcal{V}}| \times w}$ and the edge embedding $\mathbf{H}^{\mathcal{R}} \in \mathbb{R}^{|\mathcal{C}_{\mathcal{E}}| \times w}$ are initialized by the averaged word embedding in each cluster, where w denotes the dimension of the word embedding.

3.2 Step 2: Personalized Knowledge Graph Composition

For each patient, we compose their personalized KG by merging the clustered knowledge graphs of their medical codes. We create a patient node ($\mathcal{P}$) and connect it to its direct EHR nodes in the graph. The personalized KG for a patient can be represented as $G_{\text{pat}} = (\mathcal{V}_{\text{pat}}, \mathcal{E}_{\text{pat}})$, where $\mathcal{V}_{\text{pat}} = \mathcal{P} \cup \{\mathcal{V}^{'e_1}, \mathcal{V}^{'e_2}, ..., \mathcal{V}^{'e_\omega}\}$ and $\mathcal{E}_{\text{pat}} = \mathcal{E}^{\mathcal{P}} \cup \{\mathcal{E}^{'e_1}, \mathcal{E}^{'e_2}, ..., \mathcal{E}^{'e_\omega}\}$, with $\{e_1, e_2, ..., e_\omega\}$ being the medical codes directly associated with the patient with ω being the number of medical codes for this patient. Further, as a patient is represented as a sequence of J visits [29], the *visit-subgraphs* for patient i can be represented as $G_{\text{pat}(i)} = \{G_{i,1}, G_{i,2}, ..., G_{i,J}\} = \{(\mathcal{V}_{i,1}, \mathcal{E}_{i,1}), (\mathcal{V}_{i,2}, \mathcal{E}_{i,2}), ..., (\mathcal{V}_{i,J}, \mathcal{E}_{i,J})\}$ for visits $\{x_1, x_2, ..., x_J\}$ where $\mathcal{V}_{i,j} \subseteq \mathcal{V}_{\text{pat}(i)}$ and $\mathcal{E}_{i,j} \subseteq \mathcal{E}_{\text{pat}(i)}$ for $1 \leq j \leq J$.

3.3 Step 3: Bi-attention Augmented Graph Neural Network

Given that each patient's data encompasses multiple visit-subgraphs, it becomes imperative to devise a specialized model capable of managing this temporal graph data. Graph Neural Networks (GNNs), known for their proficiency in this domain, can be generalized as follows:

$$\mathbf{h}_k^{(l+1)} = \sigma\left(\mathbf{W}^{(l)} \text{AGGREGATE}^{(l)}\left(\mathbf{h}_{k'}^{(l)} | k' \in \mathcal{N}(k)\right) + \mathbf{b}^{(l)}\right), \tag{1}$$

where $\mathbf{h}_k^{(l+1)}$ represents the updated node representation of node k at the $(l+1)$-th layer of the GNN. The function AGGREGATE$^{(l)}$ aggregates the node representations of all neighbors $\mathcal{N}(k)$ of node k at the l-th layer. $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are the learnable weight matrix and bias vector at the l-th layer, respectively. σ denotes an activation function. Nonetheless, the conventional GNN approach overlooks the temporal characteristics of our patient-specific graphs and misses the intricacies of personalized healthcare. To address this, we propose a Bi-attention Augmented (`BAT`) GNN that better accommodates temporal graph data and offers more nuanced predictive healthcare insights.

Our model. In `GraphCare`, we incorporate attention mechanisms to effectively capture relevant information from the personalized KG. We first reduce the size of node and edge embedding from the word embedding to the hidden embedding to improve model's efficiency and handle the sparsity problem. The dimension-reduced hidden embeddings are computed as follows:

$$\mathbf{h}_{i,j,k} = \mathbf{W}_v \mathbf{h}^{\mathcal{V}}_{(i,j,k)} + \mathbf{b}_v \quad \mathbf{h}_{(i,j,k)\leftrightarrow(i,j',k')} = \mathbf{W}_r \mathbf{h}^{\mathcal{R}}_{(i,j,k)\leftrightarrow(i,j',k')} + \mathbf{b}_r \tag{2}$$

where $\mathbf{W}_v, \mathbf{W}_r \in \mathbb{R}^{w \times q}$, $\mathbf{b}_v, \mathbf{b}_r \in \mathbb{R}^q$ are learnable vectors, $\mathbf{h}^{\mathcal{V}}_{(i,j,k)}, \mathbf{h}^{\mathcal{R}}_{(i,j,k)\leftrightarrow(i,j',k')} \in \mathbb{R}^w$ are input embedding, $\mathbf{h}_{i,j,k}, \mathbf{h}_{(i,j,k)\leftrightarrow(i,j',k')} \in \mathbb{R}^q$ are hidden embedding of the k-th node in j-th visit-subgraph of patient, and the hidden embedding of the edge between the nodes $v_{i,j,k}$ and $v_{i,j'k'}$, respectively. q is the size of the hidden embedding.

Subsequently, we compute two sets of attention weights: one set corresponds to the subgraph associated with each visit, and the other pertains to the nodes within each subgraph. The node-level attention weight for the k-th node in the j-th visit-subgraph of patient i, denoted as $\alpha_{i,j,k}$, and the visit-level attention weight for the j-th visit of patient i, denoted as $\beta_{i,j}$, are shown as follows:

$$\alpha_{i,j,1}, ..., \alpha_{i,j,M} = \text{Softmax}(\mathbf{W}_\alpha \mathbf{g}_{i,j} + \mathbf{b}_\alpha),$$
$$\beta_{i,1}, ..., \beta_{i,N} = \boldsymbol{\lambda}^\top \text{Tanh}(\mathbf{w}_\beta^\top \mathbf{G}_i + \mathbf{b}_\beta), \quad \text{where} \quad \boldsymbol{\lambda} = [\lambda_1, ..., \lambda_N], \tag{3}$$

where $\mathbf{g}_{i,j} \in \mathbb{R}^M$ is a multi-hot vector representation of visit-subgraph $G_{i,j}$, indicating the nodes appeared for the j-th visit of patient i where $M = |C_\mathcal{V}|$ is the number of nodes in the global graph G'. $\mathbf{G}_i \in \mathbb{R}^{N \times M}$ represents the multi-hot matrix of patient i's graph G_i where N is the maximum visits across all patients. $\mathbf{W}_\alpha \in \mathbb{R}^{M \times M}$, $\mathbf{w}_\beta \in \mathbb{R}^M$, $\mathbf{b}_\alpha \in \mathbb{R}^M$ and $\mathbf{b}_\beta \in \mathbb{R}^N$ are learnable parameters. $\boldsymbol{\lambda} \in \mathbb{R}^N$ is the decay coefficient vector, J is the number of visits of patient i, $\lambda_j \in \boldsymbol{\lambda}$ where $\lambda_j = \exp(-\gamma(J - j))$ when $j \leq J$ and 0 otherwise, is coefficient for the visit j, with decay rate γ, initializing higher weights for more recent visits.

Attention initialization. To further incorporate prior knowledge from LLMs and help the model converge, we initialize $\mathbf{W}_\alpha$ for the node-level attention based on the cosine similarity between the node embedding and the word embedding $\mathbf{w}_{\text{task}}$ of a specific term for the prediction task at hand (e.g., "death" for mortality prediction). Formally, we first calculate the weights for the nodes in the global graph G' by $w_m = (\mathbf{h}_m \cdot \mathbf{w}_{\text{task}})/(||\mathbf{h}_m||_2 \cdot ||\mathbf{w}_{\text{task}}||_2)$ where $\mathbf{h}_m \in \mathbf{H}^\mathcal{V}$ is the input embedding of the m-th node in G', and w_m is the weight computed. We normalize the weights such that $0 \leq w^m \leq 1, \forall 1 \leq m \leq M$. We initialize $\mathbf{W}_\alpha = \text{diag}(w_1, ..., w_M)$ as a diagonal matrix.
Next, we update the node embedding by aggregating the neighboring nodes across all visit-subgraphs incorporating the attention weights for visits and nodes computed in Eq (3) and the weights for edges. Based on Eq (1), we design our convolutional layer `BAT` as follows:

$$\mathbf{h}_{i,j,k}^{(l+1)} = \sigma\left(\mathbf{W}^{(l)} \sum_{j' \in J, k' \in \mathcal{N}(k) \cup \{k\}} \left(\underbrace{\alpha_{i,j',k'}^{(l)} \beta_{i,j'}^{(l)} \mathbf{h}_{i,j',k'}^{(l)}}_{\text{Node aggregation term}} + \underbrace{\text{w}_{\mathcal{R}\langle k,k' \rangle}^{(l)} \mathbf{h}_{(i,j,k) \leftrightarrow (i,j',k')}}_{\text{Edge aggregation term}} \right) + \mathbf{b}^{(l)} \right), \tag{4}$$

where σ is the ReLU function, $\mathbf{W}^{(l)} \in \mathbb{R}^{q \times q}$, $\mathbf{b}^{(l)} \in \mathbb{R}^q$ are learnable parameters, $\mathbf{w}_\mathcal{R}^{(l)} \in \mathbb{R}^{|\mathcal{C}_\mathcal{E}|}$ is the edge weight vector at the layer l, and $\text{w}_{\mathcal{R}\langle k,k' \rangle}^{(l)} \in \mathbf{w}_\mathcal{R}^{(l)}$ is the scalar weight for the edge embedding $\mathbf{h}_{(i,j,k) \leftrightarrow (i,j',k')}^\mathcal{R}$. In Eq (4), the node aggregation term captures the contribution of the attention-weighted nodes, while the edge aggregation term represents the influence of the edges connecting the nodes. This convolutional layer integrates both node and edge features, allowing the model to learn a rich representation of the patient's EHR data. After several layers of convolution, we obtain the node embeddings $\mathbf{h}_{i,j,k}^{(L)}$ of the final layer (L), which are used for the predictions:

$$\mathbf{h}_i^{G_{\text{pat}}} = \text{MEAN}(\sum_{j=1}^{J} \sum_{k=1}^{K_j} \mathbf{h}_{i,j,k}^{(L)}), \quad \mathbf{h}_i^\mathcal{P} = \text{MEAN}(\sum_{j=1}^{J} \sum_{k=1}^{K_j} \mathbb{1}_{i,j,k}^\Delta \mathbf{h}_{i,j,k}^{(L)}),$$
$$\mathbf{z}_i^{\text{graph}} = \text{MLP}(\mathbf{h}_i^{G_{\text{pat}}}), \quad \mathbf{z}_i^{\text{node}} = \text{MLP}(\mathbf{h}_i^\mathcal{P}) \quad \mathbf{z}_i^{\text{joint}} = \text{MLP}(\mathbf{h}_i^{G_{\text{pat}}} \oplus \mathbf{h}_i^\mathcal{P}), \tag{5}$$

where J is the number of visits of patient i, K_j is the number of nodes in visit j, $\mathbf{h}_i^{G_{\text{pat}}}$ denotes the patient graph embedding obtained by averaging the embeddings of all nodes across visit-subgraphs and the various nodes within each subgraph for patient i. $\mathbf{h}_i^\mathcal{P}$ represents the patient node embedding computed by averaging node embeddings of the direct medical code linked to the patient node. $\mathbb{1}_{i,j,k}^\Delta \in \{0, 1\}$ is a binary label indicating whether a node $v_{i,j,k}$ corresponds to a direct medical code for patient i. Finally, we apply a multilayer perception (MLP) to either $\mathbf{h}_i^{G_{\text{pat}}}$, $\mathbf{h}_i^\mathcal{P}$, or the concatenated embedding $(\mathbf{h}_i^{G_{\text{pat}}} \oplus \mathbf{h}_i^\mathcal{P})$ to obtain logits $\mathbf{z}_i^{\text{graph}}$, $\mathbf{z}_i^{\text{node}}$ or $\mathbf{z}_i^{\text{joint}}$ respectively. We discuss more details of the patient representations in Appendix D.

3.4 Training and Prediction

The model can be adapted for a variety of healthcare prediction tasks. Consider a set of samples $\{(x_1), (x_1, x_2), \ldots, (x_1, x_2, \ldots, x_t)\}$ for each patient with t visits, where each tuple represents a sample consisting of a sequence of consecutive visits.

Mortality (MT.) prediction predicts the mortality label of the subsequent visit for each sample, with the last sample dropped. Formally, $f : (x_1, x_2, \ldots, x_{t-1}) \rightarrow y[x_t]$ where $y[x_t] \in \{0, 1\}$ is a binary label indicating the patient's survival status recorded in visit x_t.

Readmission (RA.) prediction predicts if the patient will be readmitted into hospital within σ days. Formally, $f : (x_1, x_2, \ldots, x_{t-1}) \rightarrow y[\tau(x_t) - \tau(x_{t-1})], y \in \{0, 1\}$ where $\tau(x_t)$ denotes the encounter time of visit x_t. $y[\tau(x_t) - \tau(x_{t-1})]$ equals 1 if $\tau(x_t) - \tau(x_{t-1}) \leq \sigma$, and 0 otherwise. In our study, we set $\sigma = 15$ days.

Length-Of-Stay (LOS) prediction [64] predicts the length of ICU stays for each visit. Formally, $f : (x_1, x_2, \ldots, x_t) \rightarrow y[x_t]$ where $y[x_t] \in \mathbb{R}^{1\times C}$ is a one-hot vector indicating its class among C classes. We set 10 classes $[\mathbf{0}, \mathbf{1}, \ldots, \mathbf{7}, \mathbf{8}, \mathbf{9}]$, which signify the stays of length < 1 day ($\mathbf{0}$), within one week ($\mathbf{1}, \ldots, \mathbf{7}$), between one and two weeks ($\mathbf{8}$), and longer than two weeks ($\mathbf{9}$).

Drug recommendation predicts medication labels for each visit. Formally, $f : (x_1, x_2, \ldots, x_t) \rightarrow y[x_t]$ where $y[x_t] \in \mathbb{R}^{1\times|\mathbf{d}|}$ is a multi-hot vector where $|\mathbf{d}|$ denotes the number of all drug types.

The loss functions used to train the model are as follows:

$$\mathcal{L}_{bce} = -\frac{1}{P}\sum_{i=1}^{P}[y_i \log(\sigma(z_i)) + (1 - y_i)\log(1 - \sigma(z_i))],\ \mathcal{L}_{ce} = -\frac{1}{P}\sum_{i=1}^{P}\sum_{c=1}^{C} y_{i,c} \log(\text{softmax}(z_{i,c})), \tag{6}$$

where $\mathcal{L}_{bce}$ is the binary cross-entropy loss, $\mathcal{L}_{ce}$ is the cross-entropy loss, y_i is the ground truth label for patient i, P denotes the number of patients, C is the number of classes, and z_i and $z_{i,c}$ are logits obtained from Eq (5). The sigmoid function $\sigma(z_i) = \frac{1}{1+e^{-z_i}}$ is applied for binary (MT. and RA.) and multi-label (Drug.) classification tasks, while the softmax function, $\text{softmax}(z_{i,c}) = \frac{e^{z_{i,c}}}{\sum_{c'=1}^{C} e^{z_{i,c'}}}$, is used for multi-class (LOS) classification tasks.

4 Experiments

We evaluate the performance of `GraphCare` on four healthcare prediction tasks: mortality prediction, LOS prediction, readmission prediction, and drug recommendation.

4.1 Experimental Setting

Data. For the EHR data, we use the publicly available MIMIC-III [22] and MIMIC-IV [23] datasets. Table 1 presents statistics for these processed datasets. For the generation of relational knowledge and knowledge graph construction, we utilize GPT-4 [18] as our chosen LLM. Additionally, we utilize the UMLS-KG [14] as an existing large-scale biomedical KG, which contains 300K nodes and 1M edges. We retrieve EHR subgraphs from this KG. To retrieve the word embeddings of the entities and relations within the constructed KGs, we employ the second-generation GPT-3 embedding model.

Table 1: Statistics of pre-processed EHR datasets. "#": "the number of", "/patient": "per patient".

	#patients	#visits	#visits/patient	#conditions/patient	#procedures/patient	#drugs/patient
MIMIC-III	35,707	44,399	1.24	12.89	4.54	33.71
MIMIC-IV	123,488	232,263	1.88	21.74	4.70	43.89

Baselines. Our baselines include GRU [65], Transformer [66], RETAIN [28], GRAM [12], Deepr [67], StageNet [35], AdaCare [34], GRASP [68], SafeDrug [69], MICRON [31], GAMENet [8], and MoleRec [36]. AdaCare and GRASP are evaluated only on binary prediction tasks given their computational demands. For drug recommendation, we also consider task-specific models SafeDrug, MICRON, GAMENet, and MoleRec. Our `GraphCare` model's performance is examined under three different GNNs: GAT [24], GIN [25], and our `BAT`. We do not compare to models such as GCT [10] and CGL [41] as they incorporate lab results and clinical notes, which are not used in this study. Implementation details are discussed in Supplementary.

Evaluation Metrics. We consider the following metrics: (a) **Accuracy** - the proportion of correctly predicted instances out of the total instances; (b) **F1** - the harmonic mean of precision and recall; (c) **Jaccard score** - the ratio of the intersection to the union of predicted and true labels; (d) **AUPRC** - the area under the precision-recall curve, emphasizing the trade-off between precision and recall; (e) **AUROC** - the area under the receiver operating characteristic curve, capturing the trade-off between the true positive and the false positive rates. (f) **Cohen's Kappa** - measures inter-rater agreement for categorical items, adjusting for the expected level of agreement by chance in multi-class classification.

Table 2: **Performance comparison of four prediction tasks on MIMIC-III/MIMIC-IV.** We report the average performance and the standard deviation (in bracket) of each model over 100 runs for MIMIC-III and 50 runs for MIMIC-IV. The best results are **highlighted** for both datasets.

Model	Task 1: Mortality Prediction				Task 2: Readmission Prediction			
	MIMIC-III		MIMIC-IV		MIMIC-III		MIMIC-IV	
	AUPRC	**AUROC**	**AUPRC**	**AUROC**	**AUPRC**	**AUROC**	**AUPRC**	**AUROC**
GRU	$0.12_{(0.01)}$	$0.61_{(0.01)}$	$0.04_{(0.00)}$	$0.69_{(0.01)}$	$0.68_{(0.00)}$	$0.65_{(0.01)}$	$0.66_{(0.00)}$	$0.66_{(0.00)}$
Transformer	$0.10_{(0.01)}$	$0.57_{(0.01)}$	$0.03_{(0.00)}$	$0.65_{(0.01)}$	$0.67_{(0.01)}$	$0.64_{(0.01)}$	$0.66_{(0.00)}$	$0.65_{(0.00)}$
RETAIN	$0.10_{(0.01)}$	$0.59_{(0.01)}$	$0.04_{(0.00)}$	$0.65_{(0.02)}$	$0.65_{(0.01)}$	$0.64_{(0.01)}$	$0.66_{(0.00)}$	$0.66_{(0.00)}$
GRAM	$0.11_{(0.01)}$	$0.60_{(0.01)}$	$0.04_{(0.00)}$	$0.67_{(0.01)}$	$0.67_{(0.01)}$	$0.64_{(0.00)}$	$0.66_{(0.00)}$	$0.66_{(0.00)}$
Deepr	$0.13_{(0.01)}$	$0.61_{(0.00)}$	$0.04_{(0.00)}$	$0.69_{(0.01)}$	$0.69_{(0.01)}$	$0.66_{(0.00)}$	$0.66_{(0.00)}$	$0.65_{(0.00)}$
AdaCare	$0.11_{(0.00)}$	$0.58_{(0.01)}$	$\mathbf{0.05_{(0.00)}}$	$0.69_{(0.01)}$	$0.68_{(0.01)}$	$0.66_{(0.00)}$	$0.66_{(0.00)}$	$0.66_{(0.00)}$
GRASP	$0.10_{(0.01)}$	$0.59_{(0.01)}$	$\mathbf{0.05_{(0.00)}}$	$0.68_{(0.01)}$	$0.69_{(0.00)}$	$0.66_{(0.01)}$	$0.66_{(0.00)}$	$0.66_{(0.00)}$
StageNet	$0.12_{(0.00)}$	$0.62_{(0.01)}$	$0.04_{(0.00)}$	$0.70_{(0.01)}$	$0.69_{(0.01)}$	$0.67_{(0.00)}$	$0.66_{(0.00)}$	$0.66_{(0.00)}$
`GraphCare w/ GAT`	$0.14_{(0.01)}$	$0.67_{(0.01)}$	$0.05_{(0.00)}$	$0.71_{(0.01)}$	$0.71_{(0.01)}$	$0.67_{(0.01)}$	$0.67_{(0.00)}$	$\mathbf{0.67_{(0.00)}}$
`GraphCare w/ GIN`	$0.14_{(0.00)}$	$0.66_{(0.01)}$	$\mathbf{0.05_{(0.00)}}$	$0.71_{(0.01)}$	$0.71_{(0.01)}$	$0.68_{(0.00)}$	$\mathbf{0.68_{(0.00)}}$	$\mathbf{0.67_{(0.00)}}$
`GraphCare w/ BAT`	$\mathbf{0.16_{(0.00)}}$	$\mathbf{0.70_{(0.01)}}$	$\mathbf{0.05_{(0.00)}}$	$\mathbf{0.72_{(0.01)}}$	$\mathbf{0.73_{(0.01)}}$	$\mathbf{0.69_{(0.00)}}$	$\mathbf{0.68_{(0.00)}}$	$\mathbf{0.67_{(0.00)}}$

Model	Task 3: Length of Stay Prediction							
	MIMIC-III				MIMIC-IV			
	AUROC	**Kappa**	**Accuracy**	**F1**	**AUROC**	**Kappa**	**Accuracy**	**F1**
GRU	$0.78_{(0.00)}$	$0.26_{(0.00)}$	$0.40_{(0.00)}$	$0.35_{(0.01)}$	$0.79_{(0.00)}$	$0.26_{(0.00)}$	$0.35_{(0.00)}$	$0.32_{(0.00)}$
Transformer	$0.78_{(0.00)}$	$0.25_{(0.00)}$	$0.40_{(0.00)}$	$0.35_{(0.00)}$	$0.78_{(0.00)}$	$0.25_{(0.00)}$	$0.34_{(0.00)}$	$0.31_{(0.00)}$
RETAIN	$0.78_{(0.00)}$	$0.26_{(0.00)}$	$0.41_{(0.00)}$	$0.35_{(0.00)}$	$0.79_{(0.00)}$	$0.26_{(0.00)}$	$\mathbf{0.36_{(0.00)}}$	$0.32_{(0.00)}$
GRAM	$0.78_{(0.00)}$	$0.26_{(0.00)}$	$0.40_{(0.00)}$	$0.34_{(0.00)}$	$0.79_{(0.00)}$	$0.26_{(0.00)}$	$0.35_{(0.00)}$	$0.32_{(0.00)}$
Deepr	$0.78_{(0.00)}$	$0.25_{(0.00)}$	$0.40_{(0.01)}$	$0.35_{(0.00)}$	$0.79_{(0.00)}$	$0.26_{(0.00)}$	$\mathbf{0.36_{(0.00)}}$	$0.32_{(0.00)}$
StageNet	$0.78_{(0.00)}$	$0.25_{(0.00)}$	$0.40_{(0.00)}$	$0.34_{(0.00)}$	$0.79_{(0.00)}$	$0.26_{(0.00)}$	$0.35_{(0.00)}$	$0.31_{(0.00)}$
`GraphCare w/ GAT`	$0.79_{(0.00)}$	$0.27_{(0.00)}$	$0.41_{(0.00)}$	$0.35_{(0.00)}$	$0.79_{(0.00)}$	$0.26_{(0.00)}$	$\mathbf{0.36_{(0.00)}}$	$0.32_{(0.00)}$
`GraphCare w/ GIN`	$0.79_{(0.00)}$	$0.27_{(0.00)}$	$0.41_{(0.00)}$	$0.35_{(0.00)}$	$\mathbf{0.80_{(0.00)}}$	$0.26_{(0.00)}$	$\mathbf{0.36_{(0.00)}}$	$0.32_{(0.00)}$
`GraphCare w/ BAT`	$\mathbf{0.80_{(0.00)}}$	$\mathbf{0.29_{(0.00)}}$	$\mathbf{0.42_{(0.00)}}$	$\mathbf{0.37_{(0.00)}}$	$\mathbf{0.80_{(0.00)}}$	$\mathbf{0.27_{(0.00)}}$	$\mathbf{0.36_{(0.00)}}$	$\mathbf{0.33_{(0.00)}}$

Model	Task 4: Drug Recommendation							
	MIMIC-III				MIMIC-IV			
	AUPRC	**AUROC**	**F1**	**Jaccard**	**AUPRC**	**AUROC**	**F1**	**Jaccard**
GRU	$0.77_{(0.00)}$	$0.94_{(0.00)}$	$0.62_{(0.00)}$	$0.48_{(0.00)}$	$0.74_{(0.00)}$	$0.94_{(0.00)}$	$0.60_{(0.00)}$	$0.44_{(0.00)}$
Transformer	$0.76_{(0.00)}$	$0.94_{(0.00)}$	$0.62_{(0.00)}$	$0.47_{(0.00)}$	$0.71_{(0.00)}$	$0.93_{(0.00)}$	$0.56_{(0.00)}$	$0.40_{(0.00)}$
RETAIN	$0.77_{(0.00)}$	$0.94_{(0.00)}$	$0.64_{(0.00)}$	$\mathbf{0.49_{(0.00)}}$	$0.74_{(0.00)}$	$0.94_{(0.00)}$	$0.60_{(0.00)}$	$0.45_{(0.00)}$
GRAM	$0.77_{(0.00)}$	$0.94_{(0.00)}$	$0.63_{(0.00)}$	$0.48_{(0.00)}$	$0.74_{(0.00)}$	$0.94_{(0.00)}$	$0.60_{(0.00)}$	$0.45_{(0.00)}$
Deepr	$0.74_{(0.00)}$	$0.94_{(0.00)}$	$0.60_{(0.00)}$	$0.45_{(0.00)}$	$0.74_{(0.00)}$	$0.94_{(0.00)}$	$0.59_{(0.00)}$	$0.44_{(0.00)}$
StageNet	$0.74_{(0.00)}$	$0.93_{(0.00)}$	$0.61_{(0.00)}$	$0.46_{(0.00)}$	$0.74_{(0.00)}$	$0.94_{(0.00)}$	$0.60_{(0.00)}$	$0.45_{(0.00)}$
SafeDrug	$0.68_{(0.00)}$	$0.91_{(0.00)}$	$0.47_{(0.00)}$	$0.32_{(0.00)}$	$0.66_{(0.01)}$	$0.92_{(0.00)}$	$0.56_{(0.00)}$	$0.44_{(0.00)}$
MICRON	$0.77_{(0.00)}$	$\mathbf{0.95_{(0.00)}}$	$0.63_{(0.00)}$	$0.48_{(0.00)}$	$0.74_{(0.00)}$	$0.94_{(0.00)}$	$0.59_{(0.00)}$	$0.44_{(0.00)}$
GAMENet	$0.76_{(0.00)}$	$0.94_{(0.00)}$	$0.62_{(0.00)}$	$0.47_{(0.00)}$	$0.74_{(0.00)}$	$0.94_{(0.00)}$	$0.60_{(0.00)}$	$0.45_{(0.00)}$
MoleRec	$0.70_{(0.00)}$	$0.92_{(0.00)}$	$0.58_{(0.00)}$	$0.43_{(0.00)}$	$0.69_{(0.00)}$	$0.92_{(0.00)}$	$0.56_{(0.00)}$	$0.41_{(0.00)}$
`GraphCare w/ GAT`	$0.78_{(0.00)}$	$\mathbf{0.95_{(0.00)}}$	$0.64_{(0.00)}$	$0.48_{(0.00)}$	$0.75_{(0.01)}$	$0.94_{(0.00)}$	$0.60_{(0.00)}$	$\mathbf{0.46_{(0.00)}}$
`GraphCare w/ GIN`	$0.78_{(0.00)}$	$\mathbf{0.95_{(0.00)}}$	$0.64_{(0.00)}$	$0.48_{(0.00)}$	$\mathbf{0.75_{(0.00)}}$	$\mathbf{0.95_{(0.00)}}$	$0.61_{(0.00)}$	$\mathbf{0.46_{(0.00)}}$
`GraphCare w/ BAT`	$\mathbf{0.79_{(0.00)}}$	$\mathbf{0.95_{(0.00)}}$	$\mathbf{0.65_{(0.00)}}$	$\mathbf{0.49_{(0.00)}}$	$\mathbf{0.75_{(0.00)}}$	$\mathbf{0.95_{(0.00)}}$	$\mathbf{0.62_{(0.00)}}$	$\mathbf{0.46_{(0.00)}}$

4.2 Experimental Results

As demonstrated in Table 2, the `GraphCare` framework consistently outperforms existing baselines across all prediction tasks for both MIMIC-III and MIMIC-IV datasets. For example, when combined with `BAT`, `GraphCare` exceeds StageNet's best result by $+8\%$ in AUROC for mortality prediction on MIMIC-III. Within our `GraphCare` framework, our proposed `BAT` GNN consistently outperforms GAT and GIN, underlining the effectiveness of the bi-attention mechanism. The improvement is less pronounced on MIMIC-IV than on MIMIC-III (e.g., +2.0% versus +5.3% average AUPRC improvement for the readmission prediction). This smaller margin may be due to MIMIC-IV's richer dataset providing the baselines with more patients and more features per patient for learning, as shown in Table 1. In the following, we will analyze the effects of incorporating the personalized KG and our proposed `BAT` in detail. The ablation study of bi-attention is placed in Appendix E.

4.2.1 Effect of Personalized Knowledge Graph

Effect of data size. To examine the impact of training data volume on model performance, we conduct a comprehensive experiment where the size of the training data fluctuates between 0.1% and 100% of the original training set, while the validation/testing data remain constant. Performance metrics are averaged over 10 runs, each initiated with a different random seed. The results, depicted in Figure 2, suggest that `GraphCare` exhibits a considerable edge over other models when confronted with scarce training data. For instance, `GraphCare`, despite being trained on a mere 0.1% of the training data (equivalent to 36 patient samples), accomplished an LOS prediction accuracy comparable to the best baseline StageNet that trained on 0.9% of the training data (about 320 patient samples). A similar pattern appears in drug recommendation tasks. Notably, both GAMENet and GRAM also show a certain level of resilience against data limitations, likely due to their use of internal EHR graphs or external ontologies.

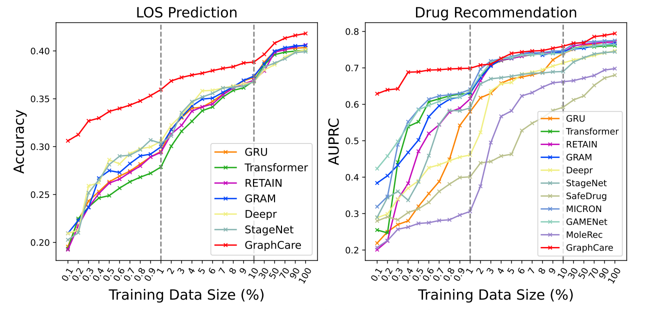


Figure 2: **Performance by the sizes of training data**. Values on the x-axis indicate % of the entire training data. The dotted lines separate three ranges: [0.1, 1], [1, 10] and [10, 100] (%).

Effect of knowledge graph sizes. To understand how the size of the KG affects the performance of `GraphCare`, we randomly sample 10 sub-KGs from the clustered global knowledge graph (i.e. G' in §3.1) with different random seeds for each ratio in range $[0.0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0]$ and report the results in Table 3. In all cases, the trend of improved performance with larger KG size is clear, highlighting the value of a more comprehensive knowledge graph in making more accurate predictions across various healthcare tasks. Furthermore, it indicates that our method is scalable and can evolve in tandem with the rapid growth of LLMs. It is also interesting to note that different tasks demonstrate different degrees of improvement from the KG; readmission prediction improvements are more modest than those observed with mortality prediction despite using the same sub-KGs.

Table 3: **The performance of `GraphCare` with different KG sizes.** All KGs are randomly sampled from the global KG while keeping the nodes of medical codes fixed. For each ratio, we report the average nodes/edges of the patient's personalized KG as well as the performance across four tasks on MIMIC-III.

Feature: Condition + Procedure + Drug								
KG Ratio		**0**	**0.1**	**0.3**	**0.5**	**0.7**	**0.9**	**1.0**
Nodes/pat.		242	362	618	883	1113	1359	1479
Edges/pat.		288	485	886	1296	1690	2170	2374
MT.	AUPRC	0.13	0.13	0.14	0.15	0.15	0.15	**0.16**
	AUROC	0.64	0.65	0.67	0.68	0.68	0.69	**0.70**
RA.	AUPRC	0.70	0.71	0.71	0.71	0.72	0.72	**0.73**
	AUROC	0.67	0.68	0.68	0.68	**0.69**	**0.69**	**0.69**

Feature: Condition + Procedure								
KG Ratio		**0**	**0.1**	**0.3**	**0.5**	**0.7**	**0.9**	**1.0**
Nodes/pat.		15	47	139	221	317	401	447
Edges/pat.		6	25	140	261	398	599	666
LOS	AUROC	0.77	0.78	0.78	0.79	0.79	**0.80**	**0.80**
	F1	0.34	0.35	0.35	0.35	0.36	0.36	**0.37**
Drug.	AUROC	0.94	0.94	0.94	**0.95**	**0.95**	**0.95**	**0.95**
	F1	0.61	0.62	0.63	0.64	0.64	**0.65**	**0.65**

4.2.2 Patient Representation Learning

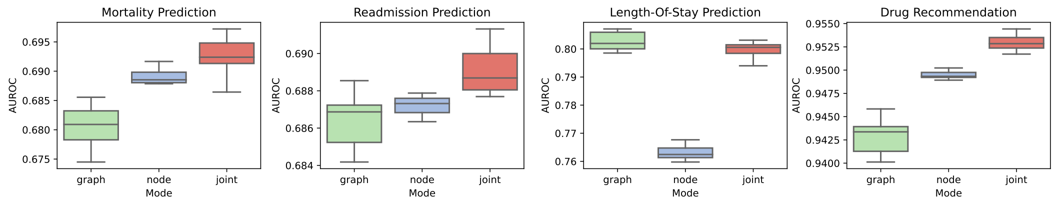


Figure 3: **Performance of healthcare predictions with three types of patient representations** (§3.3): (1) **graph** - patient graph embedding obtained through mean pooling of node embedding; (2) **node** - patient node embedding connected to the direct EHR node; (3) **joint** - embedding concatenated by (1) and (2).

We further discuss the performance of different patient representations in `GraphCare`, as depicted in Figure 3. We calculate the average over 20 independent runs for each type of patient representation and for each task. Our observations reveal that the patient node embedding presents more stability as it is computed by averaging the direct EHR nodes. These nodes are rich in precise information, thereby reducing noise, but they offer limited global information across the graph. On the other hand, patient graph embedding consistently exhibits the most significant variance, with the largest distance observed between the maximum and minimum outliers. Despite capturing a broader scope of information, the graph embedding performs less effectively due to the increased noise. This is attributed to its derivation method that averages all node embeddings within a patient's personalized KG, inherently incorporating a more diverse and complex set of information. The joint embedding operates as a balanced compromise between the node and graph embeddings. It allows `GraphCare` to learn from both local and global information. Despite the increased noise, the joint embedding provides an enriched context that improves the model's understanding and prediction capabilities.

4.2.3 Interpretability of `GraphCare`

Figure 4 showcases an example of a personalized KG for mortality prediction tied to a specific patient (predicted mortality 1), who was accurately predicted only by our `GraphCare` method, while other baselines incorrectly estimated the outcome.

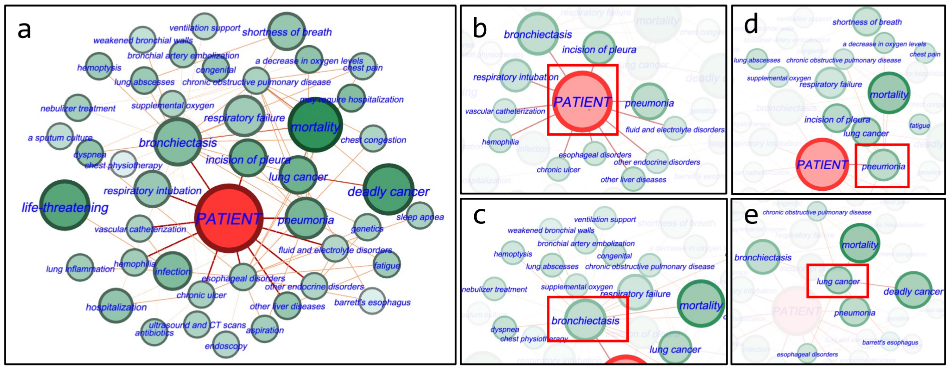


Figure 4: Example showing a patient's personalized KG with importance scores (Appendix F) visualized. For better presentation, we hide the nodes of drugs. The red node represents the patient node. Nodes with higher scores are darker and larger. Edges with higher scores are darker and thicker. Subgraph (a) shows a central area overview of this personalized KG, and other subgraphs show more details with a focused node highlighted.

In Figure 4(a), important nodes and edges contributing to mortality prediction, such as "*deadly cancer*", are emphasized with higher importance scores. This demonstrates the effectiveness of our `BAT` model in identifying relevant nodes and edges. Additionally, Figure 4(b) shows the direct Electronic Health Record (EHR) nodes connected to the patient node, enhancing interpretability of predictions using patient node embedding. Figure 4(c) and (d) show KG triples linked to the direct EHR nodes "*bronchiectasis*" and "*pneumonia*". These nodes are connected to important nodes like "*mortality*", "*respiratory failure*", "*lung cancer*", and "*shortness of breath*", indicating their higher weights. In Figure 4(e), the "*lung cancer*" node serves as a common connector for "*bronchiectasis*" and "*pneumonia*". It is linked to both "*mortality*" and "*deadly cancer*", highlighting its significance. Removing this node had a noticeable impact on the model's performance, indicating its pivotal role in accurate predictions. This emphasizes the value of comprehensive health data and considering all potential health factors, no matter how indirectly connected they may seem.

5 Conclusion

We introduced `GraphCare`, a novel framework that enriches healthcare predictions by leveraging open-world knowledge graphs (KGs) to generate personalized knowledge graphs. The primary innovation of `GraphCare` lies in its ability to import external relational knowledge linked with medical codes found in a patient's electronic health record (EHR). This information is processed by our proposed bi-attention augmented (`BAT`) Graph Neural Network (GNN), which captures crucial information from temporal personalized KG to aid healthcare predictions. Our empirical results demonstrate `GraphCare`'s superior performance, outperforming baselines across four prediction tasks on two real-world datasets. `GraphCare` proves robust with limited training data, highlighting its scalability and real-world applicability. The effectiveness of `GraphCare` scales with the size of the used KGs, indicating its capabilities will grow with advancements in large language models, our principal source for KG triple retrieval. We therefore assert that `GraphCare` holds immense potential for widespread applications in the medical domain.

References

[1] Mathias Carl Blom, Awais Ashfaq, Anita Sant'Anna, Philip D Anderson, and Markus Lingman. Training machine learning models to predict 30-day mortality in patients discharged from the emergency department: a retrospective, population-based registry study. *BMJ open*, 9(8):e028015, 2019.

[2] Katherine R Courtright, Corey Chivers, Michael Becker, Susan H Regli, Linnea C Pepper, Michael E Draugelis, and Nina R O'Connor. Electronic health record mortality prediction model for targeted palliative care among hospitalized medical patients: a pilot quasi-experimental study. *Journal of general internal medicine*, 34:1841–1847, 2019.

[3] Xiongcai Cai, Oscar Perez-Concha, Enrico Coiera, Fernando Martin-Sanchez, Richard Day, David Roffe, and Blanca Gallego. Real-time prediction of mortality, readmission, and length of stay using electronic health record data. *Journal of the American Medical Informatics Association*, 23(3):553–561, 09 2015.

[4] Scott Levin, Sean Barnes, Matthew Toerper, Arnaud Debraine, Anthony DeAngelo, Eric Hamrock, Jeremiah Hinson, Erik Hoyer, Trushar Dungarani, and Eric Howell. Machine-learning-based hospital discharge predictions can support multidisciplinary rounds and decrease hospital length-of-stay. *BMJ Innovations*, 7(2), 2021.

[5] Awais Ashfaq, Anita Sant'Anna, Markus Lingman, and Sławomir Nowaczyk. Readmission prediction using deep learning on electronic health records. *Journal of biomedical informatics*, 97:103256, 2019.

[6] Cao Xiao, Tengfei Ma, Adji B Dieng, David M Blei, and Fei Wang. Readmission prediction via deep contextual embedding of clinical concepts. *PloS one*, 13(4):e0195024, 2018.

[7] Suman Bhoi, Mong Li Lee, Wynne Hsu, Hao Sen Andrew Fang, and Ngiap Chuan Tan. Personalizing medication recommendation with a graph-based approach. *ACM Transactions on Information Systems (TOIS)*, 40(3):1–23, 2021.

[8] Junyuan Shang, Cao Xiao, Tengfei Ma, Hongyan Li, and Jimeng Sun. Gamenet: Graph augmented memory networks for recommending medication combination. In *proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1126–1133, 2019.

[9] Irene Y Chen, Monica Agrawal, Steven Horng, and David Sontag. Robustly extracting medical knowledge from ehrs: a case study of learning a health knowledge graph. In *PACIFIC SYMPOSIUM ON BIOCOMPUTING 2020*, pages 19–30. World Scientific, 2019.

[10] Edward Choi, Zhen Xu, Yujia Li, Michael Dusenberry, Gerardo Flores, Emily Xue, and Andrew Dai. Learning the graphical structure of electronic health records with graph convolutional transformer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 606–613, 2020.

[11] Maya Rotmensch, Yoni Halpern, Abdulhakim Tlimat, Steven Horng, and David Sontag. Learning a health knowledge graph from electronic medical records. *Scientific reports*, 7(1):1–11, 2017.

[12] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F Stewart, and Jimeng Sun. Gram: graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 787–795, 2017.

[13] Edward Choi, Cao Xiao, Walter Stewart, and Jimeng Sun. Mime: Multilevel medical embedding of electronic health records for predictive healthcare. *Advances in neural information processing systems*, 31, 2018.

[14] Olivier Bodenreider. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270, 2004.

[15] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[16] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[17] Renqian Luo, Liai Sun, Yingce Xia, Tao Qin, Sheng Zhang, Hoifung Poon, and Tie-Yan Liu. BioGPT: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics*, 23(6), sep 2022.

[18] OpenAI. Gpt-4 technical report, 2023.

[19] Xin Lv, Yankai Lin, Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Peng Li, and Jie Zhou. Do pre-trained models benefit knowledge graph completion? a reliable evaluation and a reasonable approach. Association for Computational Linguistics, 2022.

[20] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

[21] Badr AlKhamissi, Millicent Li, Asli Celikyilmaz, Mona Diab, and Marjan Ghazvininejad. A review on language models as knowledge bases, 2022.

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

[23] Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. Mimic-iv. *PhysioNet. Available online at: https://physionet. org/content/mimiciv/1.0/(accessed August 23, 2021)*, 2020.

[24] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[25] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[26] Benjamin Shickel, Patrick James Tighe, Azra Bihorac, and Parisa Rashidi. Deep ehr: a survey of recent advances in deep learning techniques for electronic health record (ehr) analysis. *IEEE journal of biomedical and health informatics*, 22(5):1589–1604, 2017.

[27] Riccardo Miotto, Li Li, Brian A Kidd, and Joel T Dudley. Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, 6(1):1–10, 2016.

[28] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. *Advances in neural information processing systems*, 29, 2016.

[29] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine learning for healthcare conference*, pages 301–318. PMLR, 2016.

[30] Edward Choi, Mohammad Taha Bahadori, Elizabeth Searles, Catherine Coffey, Michael Thompson, James Bost, Javier Tejedor-Sojo, and Jimeng Sun. Multi-layer representation learning for medical concepts. In *proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1495–1504, 2016.

[31] Chaoqi Yang, Cao Xiao, Lucas Glass, and Jimeng Sun. Change matters: Medication change prediction with recurrent residual networks. *arXiv preprint arXiv:2105.01876*, 2021.

[32] Tianran Zhang, Muhao Chen, and Alex AT Bui. Diagnostic prediction with sequence-of-sets representation learning for clinical events. In *Artificial Intelligence in Medicine: 18th International Conference on Artificial Intelligence in Medicine, AIME 2020, Minneapolis, MN, USA, August 25–28, 2020, Proceedings 18*, pages 348–358. Springer, 2020.

[33] Liantao Ma, Chaohe Zhang, Yasha Wang, Wenjie Ruan, Jiangtao Wang, Wen Tang, Xinyu Ma, Xin Gao, and Junyi Gao. Concare: Personalized clinical feature embedding via capturing the healthcare context. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 833–840, 2020.

[34] Liantao Ma, Junyi Gao, Yasha Wang, Chaohe Zhang, Jiangtao Wang, Wenjie Ruan, Wen Tang, Xin Gao, and Xinyu Ma. Adacare: Explainable clinical health status representation learning via scale-adaptive feature extraction and recalibration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 825–832, 2020.

[35] Junyi Gao, Cao Xiao, Yasha Wang, Wen Tang, Lucas M Glass, and Jimeng Sun. Stagenet: Stage-aware neural networks for health risk prediction. In *Proceedings of The Web Conference 2020*, pages 530–540, 2020.

[36] Nianzu Yang, Kaipeng Zeng, Qitian Wu, and Junchi Yan. Molerec: Combinatorial drug recommendation with substructure-aware molecular representation learning. In *Proceedings of the ACM Web Conference 2023*, pages 4075–4085, 2023.

[37] Chenhao Su, Sheng Gao, and Si Li. Gate: graph-attention augmented temporal neural network for medication recommendation. *IEEE Access*, 8:125447–125458, 2020.

[38] Weicheng Zhu and Narges Razavian. Variationally regularized graph-based representation learning for electronic health records. In *Proceedings of the Conference on Health, Inference, and Learning*, pages 1–13, 2021.

[39] Yang Li, Buyue Qian, Xianli Zhang, and Hui Liu. Graph neural network-based diagnosis prediction. *Big Data*, 8(5):379–390, 2020.

[40] Xiancheng Xie, Yun Xiong, Philip S Yu, and Yangyong Zhu. Ehr coding with multi-scale feature attention and structured knowledge graph propagation. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 649–658, 2019.

[41] Chang Lu, Chandan K Reddy, Prithwish Chakraborty, Samantha Kleinberg, and Yue Ning. Collaborative graph learning with auxiliary text for temporal event prediction in healthcare. *arXiv preprint arXiv:2105.07542*, 2021.

[42] Kelsey S Lau-Min, Stephanie Byers Asher, Jessica Chen, Susan M Domchek, Michael Feldman, Steven Joffe, Jeffrey Landgraf, Virginia Speare, Lisa A Varughese, Sony Tuteja, et al. Real-world integration of genomic data into the electronic health record: the pennchart genomics initiative. *Genetics in Medicine*, 23(4):603–605, 2021.

[43] Xuequn Pan and James J Cimino. Locating relevant patient information in electronic health record data using representations of clinical concepts and database structures. In *AMIA Annual Symposium Proceedings*, volume 2014, page 969. American Medical Informatics Association, 2014.

[44] Peipei Ping, Karol Watson, Jiawei Han, and Alex Bui. Individualized knowledge graph: a viable informatics path to precision medicine. *Circulation research*, 120(7):1078–1080, 2017.

[45] Amelie Gyrard, Manas Gaur, Saeedeh Shekarpour, Krishnaprasad Thirunarayan, and Amit Sheth. Personalized health knowledge graph. In *CEUR workshop proceedings*, volume 2317. NIH Public Access, 2018.

[46] Sola Shirai, Oshani Seneviratne, and Deborah L McGuinness. Applying personal knowledge graphs to health. *arXiv preprint arXiv:2104.07587*, 2021.

[47] Nidhi Rastogi and Mohammed J Zaki. Personal health knowledge graphs for patients. *arXiv preprint arXiv:2004.00071*, 2020.

[48] Michelle M Li, Kexin Huang, and Marinka Zitnik. Graph representation learning in biomedicine and healthcare. *Nature Biomedical Engineering*, pages 1–17, 2022.

[49] Fenglong Ma, Quanzeng You, Houping Xiao, Radha Chitta, Jing Zhou, and Jing Gao. Kame: Knowledge-based attention model for diagnosis prediction in healthcare. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 743–752, 2018.

[50] Junyuan Shang, Tengfei Ma, Cao Xiao, and Jimeng Sun. Pre-training of graph augmented transformers for medication recommendation. *arXiv preprint arXiv:1906.00346*, 2019.

[51] Changchang Yin, Rongjian Zhao, Buyue Qian, Xin Lv, and Ping Zhang. Domain knowledge guided deep learning with electronic health records. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 738–747. IEEE, 2019.

[52] Cecilia Panigutti, Alan Perotti, and Dino Pedreschi. Doctor xai: an ontology-based approach to black-box sequential data classification explanations. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 629–639, 2020.

[53] Junyi Gao, Chaoqi Yang, Joerg Heintz, Scott Barrows, Elise Albers, Mary Stapel, Sara Warfield, Adam Cross, Jimeng Sun, et al. Medml: Fusing medical knowledge and machine learning models for early pediatric covid-19 hospitalization and severity prediction. *Iscience*, 25(9):104970, 2022.

[54] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[55] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[56] John Boaz Lee, Ryan Rossi, and Xiangnan Kong. Graph classification using structural attention. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1666–1674, 2018.

[57] Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. Gaan: Gated attention networks for learning on large and spatiotemporal graphs. *arXiv preprint arXiv:1803.07294*, 2018.

[58] Guangtao Wang, Rex Ying, Jing Huang, and Jure Leskovec. Multi-hop attention graph neural network. *arXiv preprint arXiv:2009.14332*, 2020.

[59] Bingfeng Zhang, Jimin Xiao, Jianbo Jiao, Yunchao Wei, and Yao Zhao. Affinity attention graph neural network for weakly supervised semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):8082–8096, 2021.

[60] Boris Knyazev, Graham W Taylor, and Mohamed Amer. Understanding attention and generalization in graph neural networks. *Advances in neural information processing systems*, 32, 2019.

[61] François Belleau, Marc-Alexandre Nolin, Nicole Tourigny, Philippe Rigault, and Jean Morissette. Bio2rdf: towards a mashup to build bioinformatics knowledge systems. *Journal of biomedical informatics*, 41(5):706–716, 2008.

[62] Kevin Donnelly et al. Snomed-ct: The advanced terminology and coding system for ehealth. *Studies in health technology and informatics*, 121:279, 2006.

[63] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.

[64] Hrayr Harutyunyan, Hrant Khachatrian, David C Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific data*, 6(1):96, 2019.

[65] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[66] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[67] Phuoc Nguyen, Truyen Tran, Nilmini Wickramasinghe, and Svetha Venkatesh. Deepr: a convolutional net for medical records. *IEEE journal of biomedical and health informatics*, 21(1):22–30, 2016.

[68] Chaohe Zhang, Xin Gao, Liantao Ma, Yasha Wang, Jiangtao Wang, and Wen Tang. Grasp: generic framework for health status representation learning based on incorporating knowledge from similar patients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 715–723, 2021.

[69] Chaoqi Yang, Cao Xiao, Fenglong Ma, Lucas Glass, and Jimeng Sun. Safedrug: Dual molecular graph encoders for recommending effective and safe drug combinations. *arXiv preprint arXiv:2105.02711*, 2021.

[70] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[71] Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

[72] Chaoqi Yang, Zhenbang Wu, Patrick Jiang, Zhen Lin, and Jimeng Sun. PyHealth: A deep learning toolkit for healthcare predictive modeling, 09 2022.

[73] Palmer L Elixhauser A, Steiner C. Clinical classifications software (ccs). 03 2016.

[74] Natalya F Noy, Nigam H Shah, Patricia L Whetzel, Benjamin Dai, Michael Dorf, Nicholas Griffith, Clement Jonquet, Daniel L Rubin, Margaret-Anne Storey, Christopher G Chute, et al. Bioportal: ontologies and integrated data resources at the click of a mouse. *Nucleic acids research*, 37(suppl_2):W170–W173, 2009.

[75] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

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

[77] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi: an open source software for exploring and manipulating networks. In *Proceedings of the international AAAI conference on web and social media*, volume 3, pages 361–362, 2009.

[78] Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. Towards controllable biases in language generation. *arXiv preprint arXiv:2005.00268*, 2020.

[79] Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*, 2021.

[80] Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*, 2020.

A Implementation

We implement `GraphCare` using Python 3.8.13, PyTorch 1.12.0 [70], and PyTorch Geometric 2.3.0 [71]. We employ PyHealth [72] to pre-process the EHR data, which is illustrated in Appendix B. We utilize medical code mappings from ICD-(9/10) to CCS for conditions and procedures, from NDC to ATC (level-3) for drugs, and from ICD to UMLS-CUI for subgraph retrieval. The mapping files are provided by AHRQ [73] and BioPortal [74]. For medical code KG retrieval (§3.1), we set $\chi = 3$ and $\kappa = 2$. The word embeddings retrieved from second-generation GPT-3 are 1536-dimension vectors. We use Scikit-learn 1.2.1 [75] for agglomerative clustering, with the distance threshold $\delta = 0.15$. For the model training, we split the dataset by 8:1:1 for training/validation/testing data, and we use Adam [76] as the optimizer with learning rate 1e-5 and weight decay 1e-5. We apply batch size 4 and hidden dimension 128. We take conditions and procedures as the features for the length-of-stay prediction and drug recommendation and additionally take drugs as features for mortality prediction and readmission prediction. All models are trained via 50 epochs over all patient samples, and the early stop strategy monitored by AUROC with 10 epochs is applied. We detail the hyper-parameter tuning in Appendix G. All experiments are conducted on a machine equipped with two AMD EPYC 7513 32-Core Processors, 528GB RAM, eight NVIDIA RTX A6000 GPUs, and CUDA 11.7. We use Gephi [77] for knowledge graph visualization.

B Dataset Processing

We employ PyHealth [72] to process the MIMIC-III and MIMIC-IV datasets. PyHealth has an EHR dataset pre-processing pipeline that standardizes the datasets, organizing each patient's data into several visits, where each visit contains unique and specified feature lists. For our experiments, we create feature lists composed of conditions and procedures for Length of Stay (LOS) prediction and drug recommendations. For the prediction tasks of mortality and readmission, we include the medication (drug) list in addition to the condition and procedure lists.

Subsequent to the parsing of the datasets, PyHealth also enables the mapping of medical codes across various coding systems using the provided code maps. The involved coding systems in this process are ICD-9[1], ICD-10[2,3], CCS[4], NDC[5] and ATC[6]. In our experiment, we convert 11,736 ICD-9-CM codes and 72,446 ICD-10-CM codes into 285 CCS-CM codes to capture condition concepts. Similarly, we map 4,670 ICD-9-PROC codes (a part of ICD-9-CM for procedure coding) and 79,758 ICD-10-PCS codes to 231 CCS-PROC codes for procedure concepts. For drug concepts, we convert 1,143,020 NDC codes into 269 level-3 ATC codes. This mapping process enhances the training speed and the predictive performance of the model by reducing the granularity of the medical concepts.

C Knowledge Graph Construction

In this section, we illustrate our solution to construct a biomedical knowledge graph (KG) for each medical concept by prompting from a large language model (LLM) and sampling a subgraph from a well-established KG.

C.1 Prompting KG from Large Language Model

Figure 5 showcases a carefully designed prompt for the retrieval of a biomedical KG from a LLM. The main goal of this approach is to leverage the extensive knowledge embedded in the LLM to extract meaningful triples consisting of two entities and a relationship.

In our strategy, we begin with a prompt related to a medical condition, a procedure, or a drug. The LLM is then tasked with generating a list of updates that extrapolate as many relationships as possible from this prompt. Each update is a triple following the format `[ENTITY 1, RELATIONSHIP,`

[1] `https://www.cdc.gov/nchs/icd/icd9cm.htm`
[2] `https://www.cms.gov/medicare/icd-10/2023-icd-10-cm`
[3] `https://www.cms.gov/medicare/icd-10/2023-icd-10-pcs`
[4] `https://www.hcup-us.ahrq.gov/toolssoftware/ccs/ccs.jsp`
[5] `https://www.accessdata.fda.gov/scripts/cder/ndc/index.cfm`
[6] `https://www.who.int/tools/atc-ddd-toolkit/atc-classification`

```
def ehr_kg_prompting(term, category):

•   if category == "condition":
•   example = \
•   """
•   Example:
•   prompt: systemic lupus erythematosus
•   updates: [[systemic lupus erythematosus, is an, autoimmune condition], [systemic
    lupus erythematosus, may cause, nephritis], [anti-nuclear antigen, is a test
    for, systemic lupus erythematosus], [systemic lupus erythematosus, is treated
    with, steroids], [methylprednisolone, is a, steroid]]
•   """
•   elif category == "procedure":
•   example = \
•   """
•   Example:
•   prompt: endoscopy
•   updates: [[endoscopy, is a, medical procedure], [endoscopy, used for,
    diagnosis], [endoscopic biopsy, is a type of, endoscopy], [endoscopic biopsy,
    can detect, ulcers]]
•   """
•   elif category == "drug":
•   example = \
•   """
•   Example:
•   prompt: iobenzamic acid
•   updates: [[iobenzamic acid, is a, drug], [iobenzamic acid, may have, side
    effects], [side effects, can include, nausea], [iobenzamic acid, used as, X-ray
    contrast agent], [iobenzamic acid, formula, C16H13I3N2O3]]
•   """


•   gpt_4 = GPT4()
•   response = gpt_4.chat(
•   f"""
•   Given a prompt (a medical condition/procedure/drug), extrapolate as many
    relationships as possible of it and provide a list of updates.
•   The relationships should be helpful for healthcare prediction (e.g., drug
    recommendation, mortality prediction, readmission prediction …)
•   Each update should be exactly in format of [ENTITY 1, RELATIONSHIP, ENTITY 2].
    The relationship is directed, so the order matters.
•   Both ENTITY 1 and ENTITY 2 should be noun.
•   Any element in [ENTITY 1, RELATIONSHIP, ENTITY 2] should be conclusive, make it
    as short as possible.
•   Do this in both breadth and depth. Expand [ENTITY 1, RELATIONSHIP, ENTITY 2]
    until the size reaches 100.
•
    {example}
•
    prompt: {term}
•   updates:

•   """
•   )

•   # Process the response to triples
•   triples = parse(response)

return triples
```

Figure 5: **Prompting knowledge graphs for EHR terms from GPT-4**.

ENTITY 2] where ENTITY 1 and ENTITY 2 should be nouns. Our goal is to generate these triples in both breadth (a wide variety of entities and relationships related to the initial term) and depth (following chains of relationships to discover new entities and relationships). The process continues until we obtain a list with 100 KG triples. This prompting-based approach provides a structured, interconnected knowledge graph from the unstructured knowledge embedded in the LLM, proving especially beneficial for personalized KG generation.

In the experiment, we iterate through the vocabulary of conditions, procedures and drugs contained in CCS and ATC (level-3) with their code-name mappings[7,8]: $M : e^{\triangle} \leftarrow e$ where $e^{\triangle}$ is the corresponding name for the medical code e. For each term $e^{\triangle}$, we input it with its category (either "condition", "procedure" or "drug") to the function `ehr_kg_prompting()` shown in Figure 5 χ times and compose the graphs of all runs into one, i.e., $G^e = (G^e_1 \cup G^e_2 \cup ... \cup G^e_{\chi})$, to obtain more comprehensive graphs. As a result, we obtained 85,387 non-redundant KG triples with 42,056 unique entities and 9,404 unique relations when we set $\chi = 3$. For future work, we will explore to use this prompting-based method to construct more task-specific KGs, aiming at providing more relevant triples especially beneficial to a certain prediction.

C.2 Sampling Subgraph from Existing KG

Algorithm 1 Subgraph Sampling

```
1: procedure SUBGRAPHSAMPLING(EHR code $e$, KG $\mathcal{G}$, hop limit $\kappa$, window size $\epsilon$)
2:     Initialize an empty list $Q$ and an empty graph $G^e_{\text{sub}(\kappa)} = (\mathcal{V}^e_{\text{sub}(\kappa)}, \mathcal{E}^e_{\text{sub}(\kappa)})$
3:     Add $e$ to $Q$
4:     $\mathcal{V}^e_{\text{sub}(\kappa)} \leftarrow \mathcal{V}^e_{\text{sub}(\kappa)} \cup \{e\}$
5:     for $i = 1$ to $\kappa$ do
6:         Initialize an empty list $Q_{\text{next}}$
7:         for all $\text{ent} \in Q$ do
8:             if $i = 1$ then
9:                 Retrieve all triples $(\text{ent}, \text{rel}, \text{ent}')$ or $(\text{ent}', \text{rel}, \text{ent})$ from $\mathcal{G}$
10:            else
11:                Randomly retrieve $\epsilon$ triples $(\text{ent}, \text{rel}, \text{ent}')$ or $(\text{ent}', \text{rel}, \text{ent})$ from $\mathcal{G}$
12:            end if
13:            Add retrieved triples to $\mathcal{E}^e_{\text{sub}(\kappa)}$
14:            $\mathcal{V}^e_{\text{sub}(\kappa)} \leftarrow \mathcal{V}^e_{\text{sub}(\kappa)} \cup \{\text{ent}'\}$
15:            Add $\text{ent}'$ to $Q_{\text{next}}$
16:        end for
17:        $Q \leftarrow Q_{\text{next}}$
18:    end for
19:    return $G^e_{\text{sub}(\kappa)}$
20: end procedure
```

We illustrate how we extract subgraphs for medical codes from existing well-established biomedical KG like UMLS [14] in Algorithm 1. We have four arguments - medical code e, source KG $\mathcal{G}$, hop limit κ and window size ϵ. In brief, we search all the triples containing e for the first hop and search ϵ triples containing the other entity for each first-hop triple. For entity mappings across CCS, ATC and UMLS CUI, we use the ontology files provided by BioPortal[9]. When setting $\kappa = 2$ and $\epsilon = 5$, we obtain 247,069 non-redundant KG triples with 82,628 unique entities and 80 unique relations.

C.3 Comparison of KGs

Given the abundance of similar entities and relations retrieved in previous sections, it's essential to cluster these elements to streamline the model's learning process from the KGs. After an extensive hyper-parameter tuning, detailed in Appendix G.1, we selected $\delta = 0.15$ as the distance threshold for

[7] https://www.hcup-us.ahrq.gov/toolssoftware/ccs
[8] https://bioportal.bioontology.org/ontologies/ATC
[9] https://bioportal.bioontology.org/ontologies

the agglomerative clustering algorithm. We evaluate three distinct KGs: (1) GPT-KG, the KG we obtain from Large Language Models (LLMs) as discussed in Section §C.1; (2) UMLS-KG, the KG subsampled from the UMLS as outlined in Section §C.2; and (3) GPT-UMLS-KG, a combination of both GPT-KG and UMLS-KG, and report the results in Table 4.

Table 4: `GraphCare`'s performance by different KGs on MIMIC-III.

KG	Mortality		Readmission		LOS		Drug Rec.	
	AUPRC	AUROC	AUPRC	AUROC	F1-score	AUROC	F1-score	AUROC
GPT-KG	**15.7**	**69.6**	72.6	**68.9**	**36.6**	**80.2**	**65.2**	**95.1**
UMLS-KG	14.1	66.6	70.1	66.3	35.3	78.8	63.5	94.3
GPT-UMLS-KG	15.4	69.3	**72.7**	68.8	**36.6**	79.9	64.9	95.0

Comparing the results from Table 4 and Table 2, it's evident that incorporating KGs boosts performance in healthcare predictions. Moreover, the results indicate that GPT-KG consistently outperforms UMLS-KG across all tasks. The weaker performance of UMLS-KG might be attributed to its sparsity, resulting from its random sampling from the expansive UMLS database. Such sparse data might hinder UMLS-KG from capturing the intricate relationships crucial for effective predictions. On the other hand, GPT-KG, which is derived from large language models, presents a more comprehensive and context-aware representation of knowledge, leading to superior predictive performance. Interestingly, the combined GPT-UMLS-KG produces results akin to those using GPT-KG alone, suggesting that the addition of UMLS-KG doesn't noticeably improve outcomes, likely due to its sparse data content. In the future, we aim to devise more efficient methods for extracting beneficial knowledge graphs from large biomedical KGs like UMLS. This endeavor could further enrich our model and contribute to improved prediction performance.

D "Patient As a Graph" and "Patient As a Node"

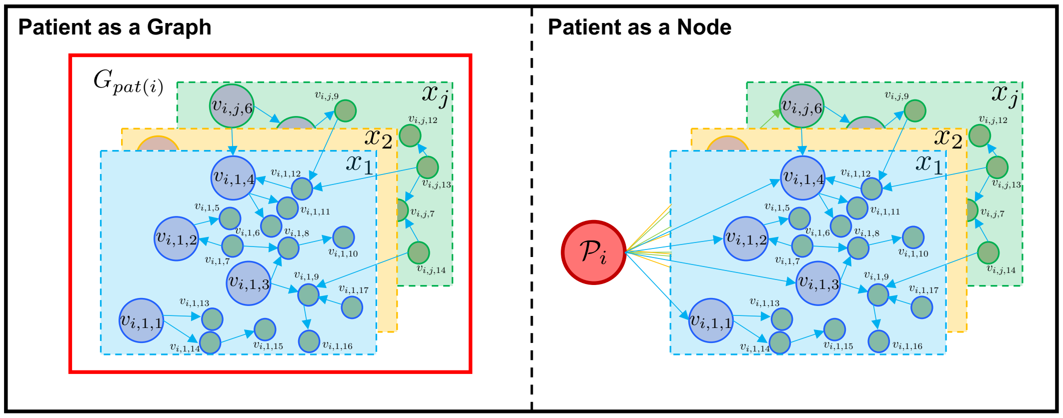


Figure 6: **Comparison of two patient representations in `GraphCare`.** ***Left***: patient as a graph covering the information in all nodes. ***Right***: patient as a node only connected to the nodes of the direct medical codes (the larger ones) recorded in the EHR dataset. x_j denotes the j-th for patient i. $v_{i,j,k}$ denotes the k-th node of the j-th visit for patient i. The connections among nodes are either inner-visit or across-visit.

Figure 6 presents two different patient representations. When viewed as a graph, the patient representation aims to encapsulate a comprehensive summary of all nodes, thus providing a broad overview of information. However, this approach may also include more noise due to its extensive scope. In contrast, when a patient is represented as a node, the information is aggregated solely from directly corresponding EHR nodes. This approach ensures a precise match with the patient's EHR data, offering a more accurate, albeit narrower, representation. Although this method provides a more focused insight, it also inevitably discards information from all other nodes, thus potentially losing broader contextual data. Therefore, the choice between these two representations hinges on the balance between precision and the extent of information required. In our experiment, we introduce a joint embedding composed by concatenating those two embeddings, as a balanced solution.

E Effect of Bi-attention Mechanism in BAT

We explore the component-wise effectiveness of BAT in Table 5. Comparing #(1,2,3) and #(4,5), we find that node-level attention is more crucial than visit-level attention. Also, by comparing #(2,4,6), we ascertain that visit-level attention gains more significance in the MIMIC-IV dataset due to its higher number of visits per patient. Comparing #7 with the other configurations shows BAT achieves better performance when both types of attentions and edge weights are employed concurrently.

Table 5: **Ablation study**. We measure the AUROC (%) for each task. α, β and $W_{\mathcal{R}}$ are node-level, visit-level attention, and edge weight, respectively. We highlight the best results on both **MIMIC-III** and **MIMIC-IV**.

#	α	β	$W_{\mathcal{R}}$	**Mortality**	**Readmission**	**LOS**	**Drug Rec.**
1	✗	✗	✗	66.9/70.5	66.7/66.2	79.2/79.0	94.5/94.1
2	✓	✗	✗	69.1/71.4	68.2/66.9	80.0/79.7	94.9/94.3
3	✗	✓	✗	67.3/71.0	67.0/66.7	79.4/79.2	94.8/94.2
4	✓	✗	✓	69.4/71.4	68.3/66.8	**80.3**/79.6	**95.1**/94.4
5	✗	✓	✓	68.0/71.3	67.3/66.3	79.5/79.3	94.8/94.2
6	✓	✓	✗	69.1/71.7	68.4/**67.2**	80.0/**80.4**	95.0/94.5
7	✓	✓	✓	**69.6**/**71.8**	**68.9**/67.1	80.2/80.1	**95.1**/**94.6**

F Importance Score

To provide insights into the GraphCare's decision-making process, we propose an interpretation method that computes the importance scores for the entities and relationships in the personalized knowledge graph. We first compute the entity importance scores as the sum of the product of node-level attention weights $\alpha_{i,j,k}$ and visit-level attention weights $\beta_{i,j}$ (obtained by Eq (3)) over all visits, and relationship importance scores as the edge weights $\mathrm{w}^{(l)}_{\mathcal{R}\langle k,k'\rangle}$ summed over all GNN layers:

$$\mathrm{I}^{\text{ent}}_{i,k} = \sum_{l=1}^{L-1}\sum_{j=1}^{J} \beta^{(l)}_{i,j}\alpha^{(l)}_{i,j,k}, \quad \mathrm{I}^{\text{rel}}_{i,k,k'} = \sum_{l=1}^{L-1} \mathrm{w}^{(l)}_{\mathcal{R}\langle k,k'\rangle}, \tag{7}$$

where $\mathrm{I}^{\text{ent}}_{i,k}$ is the importance score of entity k and $\mathrm{I}^{\text{rel}}_{i,k,k'}$ is the importance score of the relationship between entities k and k'. To identify the most crucial entities and relationships, we can also compute the top K entities and relationships with the highest importance scores, denoted as s in descending order of their importance:

$$\mathcal{T}^{\text{ent}}_{i,K} = \{s \mid s \in \mathrm{I}^{\text{ent}}_{i}, s \geq \mathrm{I}^{\text{ent}}_{i,(K)}\}, \quad \mathcal{T}^{\text{rel}}_{i,K} = \{s \mid s \in \mathrm{I}^{\text{rel}}_{i}, s \geq \mathrm{I}^{\text{rel}}_{i,(K)}\}, \tag{8}$$

where $\mathrm{I}^{\text{ent}}_{i,(K)}$ and $\mathrm{I}^{\text{rel}}_{i,(K)}$ are the K-th highest importance scores for entities and relationships, respectively, $\mathcal{T}^{\text{ent}}_{i,K}$ and $\mathcal{T}^{\text{rel}}_{i,K}$ represent the top K entities and relationships for patient i, respectively. By analyzing the top entities and relationships, we can gain a better understanding of the model's decision-making process and identify the most influential factors in its predictions.

G Hyper-parameter Tuning

Given that our GraphCare utilizes personalized knowledge graphs (KGs) as inputs for healthcare predictions, the representativeness of the constructed graphs becomes critical in the prediction process. The quality and structure of these KGs can significantly influence the performance of our predictive model, underlining the importance of thoroughly investigating the hyperparameters involved in their construction and subsequent analysis via the BAT GNN model. Therefore, we meticulously examine both the hyper-parameters for KG node/edge clustering and those for our proposed BAT Graph Neural Network (GNN) model. The validation set is used for the tuning.

G.1 Hyper-parameters for Clustering

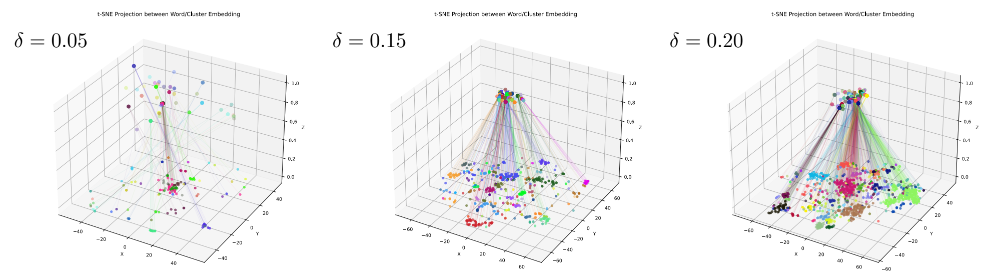
Table 6 presents the performance of GraphCare across four tasks on the MIMIC-III dataset, with varying agglomerative clustering distance thresholds $\delta \in \{0.05, 0.1, 0.15, 0.2\}$. We evaluate the performance with the GPT-KG. The results reveal that the model achieves optimal performance when $\delta = 0.15$. This outcome can be attributed to the following reasons: when δ is small, nodes of high

Table 6: Clustering hyperparameter tuning.

Threshold δ	# Cluster	**Mortality**		**Readmission**	
		AUPRC	AUROC	AUPRC	AUROC
0.05	29681	12.2	61.3	65.5	63.5
0.1	14662	13.3	65.2	70.0	67.4
0.15	4599	**15.7**	**69.6**	**72.6**	**68.9**
0.2	883	13.9	67.8	67.8	66.7

Threshold δ	# Cluster	**LOS**		**Drug Rec.**	
		F1-score	AUROC	F1-score	AUROC
0.05	15094	32.8	77.4	62.1	94.2
0.1	7941	34.7	79.7	64.8	94.5
0.15	2755	**36.6**	**80.2**	**65.2**	**95.1**
0.2	589	34.1	77.9	63.8	94.3

similarity may be incorrectly classified as distinct, complicating the learning process for the model. Conversely, if δ is large, dissimilar nodes could be inaccurately clustered together, which further challenges the training process. Examples in Figure 7 further demonstrate our findings.



Example 1: "fever and night sweats":

$\delta = 0.05$ (size:2)	fever and night sweats, night sweats
$\delta = 0.15$ (size:16)	fever and night sweats, night sweats, sweat test, hyperhidrosis and hypohidrosis, decreased sweating, hyperhidrosis, treat excessive sweating, excessive sweating, anhidrosis, sweating disorders, treat sweating disorders, treat hyperhidrosis, heat and sweating, sweating, diarrhea or excessive sweating
$\delta = 0.20$ (size:59)	fever and night sweats, night sweats, excessive moisture, dry mouth and mouth diseases, Xerostomia (dry mouth), high humidity, oral dryness, cold air or humidity, cold air and humidity, humidification, sweat test, drying out, dry eye symptoms, combined with a humidifier, drooling, dry mouth if overused, used for dry mouth, relieve dry skin, hyperhidrosis and hypohidrosis, add moisture to the air, mouth dryness, moisture, humidifier, dry mouth and thirst, the skin dry, dry mouth and throat, decreased sweating, hyperhidrosis, treat excessive sweating, xerostomia, in dry skin treatments, a humidifier, used for dry eye treatment, excessive sweating, retain moisture, dry skin, throat dryness, reduce dry mouth symptoms, dryness, dry mouth, anhidrosis, dry skin treatments, skin dryness, sweating disorders, for dry mouth, treat drooling, treat sweating disorders, treat hyperhidrosis, dry eyes, treat dry mouth, heat and sweating, dry the skin, sweating, to prevent excess moisture in affected areas, dryness or flaking of the skin, dryness in eyes, diarrhea or excessive sweating, humidity and temperature

Example 2: "brain tumors":

$\delta = 0.05$ (size:3)	brain tumors, brain tumor, brain cancer
$\delta = 0.15$ (size:21)	brain tumors, brain tumor, glioma, meningioma, oligodendroglioma, brain cancer, nerve tumor, nephroblastoma, medulloblastoma, neurofibromatosis, of developing brain tumors, ependymoma, glioblastoma, astrocytoma, especially glioblastoma, brain cancer originating in ependymal cells, brain cancer originating in star-shaped cells, brain cancer originating in oligodendrocytes, tumors in the nervous system, neuroblastoma
$\delta = 0.20$ (size:624)	brain tumors, bone metastasis from sarcoma, ductal carcinoma in situ, bone metastasis from prostate cancer, malignant transformation, prevention of uterine cancer, bone metastasis from gastrointestinal stromal tumors (GISTs), secondary malignancy treatment, risk factors for lung cancer, nonmalignant breast conditions, cancer of kidney, breast cancer risk assessment, benign neoplasm of uterus, noncancerous growth, schwannoma, cancer patients and survivors, ureter cancer, malignant tumors, ... spread of cancer to bones, risk of endometrial cancer, treat breast cancer, cure cancer, certain cancer treatments, metastases, killing cancer cells, an increased risk of prostate cancer, treating cancer, breast cancer prevention, detection of ovarian tumors, cancer management

Figure 7: **Comparison between the node clustering over GPT-KG and UMLS-KG.** ***Above***: we random sample 80 clusters with each distance threshold δ applied. Each figure visually represents the clustering of words, with color consistency denoting membership to the same cluster. ***Below***: we provide two examples of the clusters with different δ's for the given words ("fever and night sweats" and "brain tumors").

The examples presented in Figure 7 illustrate the significant influence of the distance threshold δ on the semantic coherence of the clusters. When $\delta = 0.20$, the clusters tend to incorporate words that aren't strongly semantically related to the given word. For instance, "humidity" is inappropriately grouped with "fever and night sweats", and "breast cancer" is incorrectly associated with "brain tumors". Conversely, when $\delta = 0.05$, the restrictive threshold fails to capture several words closely related to the given word, such as the absence of "heat and sweating" in the cluster for "fever and night sweats", and "glioma" for "brain tumors".

Striking an optimal balance, when $\delta = 0.15$, the resulting clusters exhibit a desirable semantic coherence. Most words within these clusters are meaningfully related to the given word. This observation underlines the importance of selecting an appropriate δ value to ensure the extraction of semantically consistent and comprehensive clusters. This is a pivotal step, as the quality of these clusters has a direct impact on subsequent healthcare prediction tasks, which rely on the KG constructed through this process.

G.2 Hyper-parameters for the Bi-attention GNN

For our proposed BAT GNN we tune the following hyper-parameters: batch size in {4, 16, 32, 64}, hidden dimension in {128, 256, 512}, learning rate in {1e-3, 1e-4, 1e-5}, weight decay in {1e-3, 1e-4, 1e-5}, decay rate γ in {0.01, 0.02, 0.03, 0.04} and number of layers L in {1, 2, 3, 4}. We show the tuning detail in Figure 8.

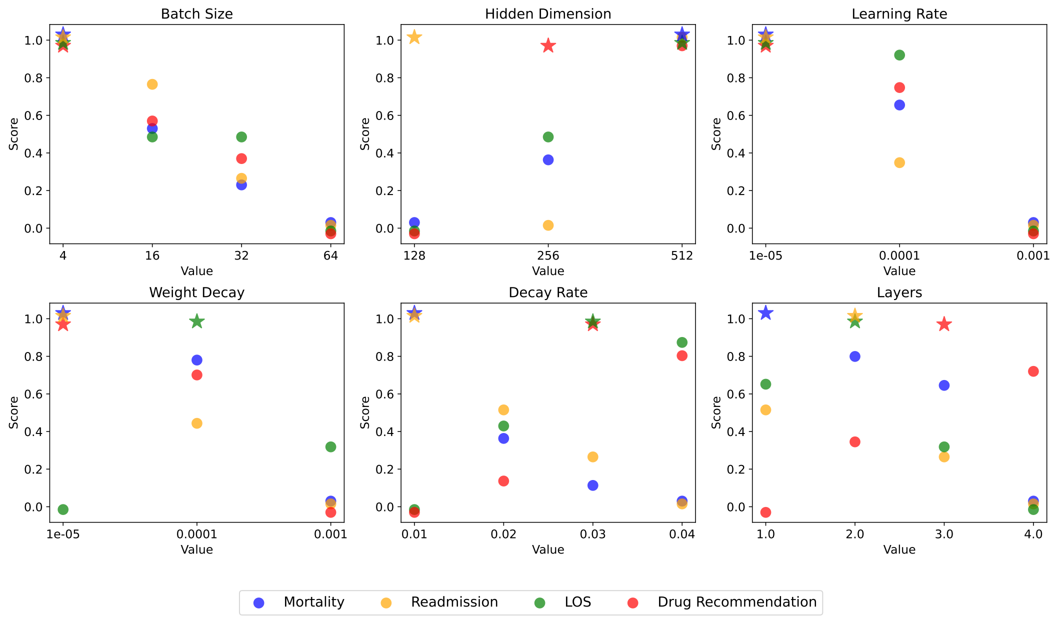


Figure 8: **Hyper-parameter tuning.** We tune each parameter while keeping other hyperparameters fixed for each task. *Score* denotes the normalized AUROC. The best value of each hyperparameter for each task is highlighted with a star.

The hyper-parameters employed throughout the experiments presented in this research are consolidated and presented in Table 7. For the sake of maintaining a fair and balanced comparison, we align the batch size, hidden dimension, learning rate, and weight decay with those of the baseline models.

Task	Batch Size	Hidden Dimension	Learning Rate	Weight Decay	Decay Rate	Layers
Mortality	4	128	1e-5	1e-5	0.01	1
Readmission	4	128	1e-5	1e-5	0.01	2
Length-Of-Stay (LOS)	4	128	1e-5	1e-5	0.03	2
Drug Recommendation	4	128	1e-5	1e-5	0.03	3

Table 7: Hyper-parameters for the BAT GNN model for different tasks.

H Ethics and Risks

In this study, we introduce a novel framework, GraphCare, which generates knowledge graphs (KGs) by leveraging relational knowledge from large language models (LLMs) and extracting information from existing KGs. This methodology is designed to provide an advanced tool for healthcare prediction tasks, enhancing their accuracy and interpretability. However, the ethical considerations associated with our approach warrant careful attention. Previous research has shown that LLMs may encode biases related to race, gender, and other demographic attributes [78, 79]. Furthermore, they may potentially generate toxic outputs [80]. Such biases and toxicity could inadvertently influence the content of the knowledge graphs generated by our proposed `GraphCare`, which relies on these LLMs for information extraction. Nevertheless, we have implemented stringent measures to mitigate these risks in our approach. First and foremost, `GraphCare` is designed to only retrieve knowledge related to medical concepts, thus significantly limiting the potential for inheriting broader social biases or toxic behaviors from the source LLMs. Moreover, we ensure that no patient information is input into any open-source software, safeguarding patient privacy and preventing the introduction of personal biases into the knowledge graphs.

I Limitations

Our proposed framework, `GraphCare`, assumes the factual accuracy of knowledge graph triples retrieved from large language models (LLMs). However, this assumption may not hold for certain pre-existing pre-trained language models. Therefore, selecting a reliable LLM becomes crucial for implementing `GraphCare` effectively. To address this issue, our framework incorporates node/edge clustering, which aggregates similar nodes and edges, facilitating pattern identification and validation of factual correctness in knowledge graph triples.

When scaling `GraphCare` for industrial applications, it is important to consider the inclusion of a human-in-the-loop module for quality control. This additional layer of scrutiny ensures the accuracy of the retrieved information and compensates for any limitations in automated processes.