From Static to Dynamic: GNNs-Driven Clinical Decision-Making Assistance

Shiyi Lin[1,2], Zirui Zhuang[1,2], Qi Qi[1,2], Jingyu Wang[1,2], Jianxin Liao[1,2],
Jiachang Hao[1,2], and Haifeng Sun[1,2](✉)

[1] Beijing University of Posts and Telecommunications, Beijing 100876, China
{041,zhuangzirui,qiqi8266,wangjingyu,liaojx,haojc,hfsun}@bupt.edu.cn
[2] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications

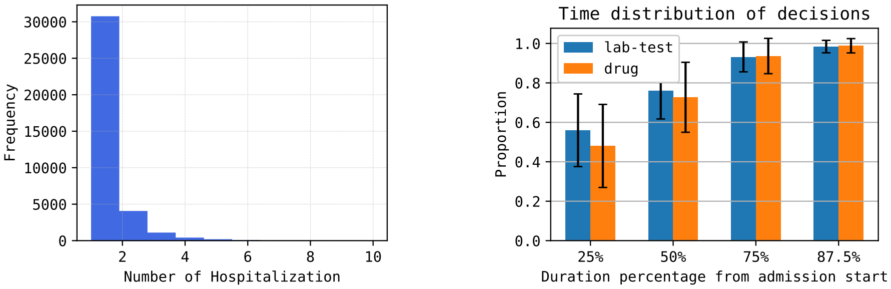
Abstract. In the realm of clinical decision-making, the complexity and variability inherent in patient care require advanced methods to assist healthcare professionals. This paper introduces a novel approach that leverages dynamic graph representation learning to enhance clinical decision-making assistance. By modeling EHRs data (Electronic Health Records) as discrete-time dynamic graphs and employing Graph Neural Networks (GNNs), our method captures the intricate and evolving interactions between patients and medical items. This reconceptualization of clinical decision-making as a recommendation system task aligns more closely with real-world scenarios, addressing limitations of previous methods such as limited patient coverage and delayed recommendations. Our experiments, conducted on two real-world clinical datasets, demonstrate the superior performance of our approach compared to traditional models, highlighting its practical utility and potential for future research and providing insight into the effective use of dynamic graphs in healthcare applications[3].

Keywords: Clinical Decision-Making · Graph Neural Networks · Dynamic Heterogeneous Graphs

1 Introduction

Clinical decision-making (including prescription decisions) involves the judicious use of evidence while considering both clinical expertise and individual patient needs and preferences [29]. This process takes into account various factors, including the patient's medical history, symptoms, and examination results, as well as the effectiveness, risks, and costs of treatment options. Due to variations in medical knowledge and clinical experience between different physicians, as well as the complexity of the clinical decision-making process, errors or omissions occasionally occur [6]. These can result in patients missing out on appropriate treatment opportunities[25]. Therefore, we hope to use some method to assist

[3] https://github.com/Nemo4110/GNNs-Driven-Clinical-Decision-Making-Assistance.git



(a) Distribution of Patient Hospitalization Frequencies

(b) Timing of Clinical Decisions During Hospitalization

Fig. 1: Statistical charts of MIMIC-III dataset.

decision-making and avoid errors or omissions as much as possible. In recent years, deep learning technology has achieved success in fields such as CV (Computer Vision), NLP (Natural Language Processing), and recommendation systems [28,1,44,7]. It has become a valuable research problem to try to use deep learning technology to learn the historical decision-making process of a large number of doctors to assist doctors in clinical decision-making.

Existing deep learning-based decision-support methods primarily focus on helping in prescription decisions, specifically the medication recommendation task[21,30,42,43,39]. These methods aim to suggest appropriate drugs based on a patient's condition by employing a framework akin to machine translation in NLP. Here, the source language input comprises the diagnosis and surgical procedure sequences from both current and previous hospitalizations, along with the medication sequence from the previous hospitalization. The target language output is the medication sequence for the current hospitalization. Through various carefully crafted mechanisms, the encoder extracts valuable representation information from the source input, enabling the decoder to generate the current hospitalization's medication sequence. This machine translation-like approach has yielded impressive results in medication recommendation tasks. However, analyzing the MIMIC-III [16] dataset used in these works, we identified two main issues: **Limited Patient Coverage** (Fig.1a): The method excludes patients with fewer than two hospitalizations, as it requires prior hospitalization sequences for input, thus covering only a small fraction of patients. **Timing of Recommendations** (Fig.1b): Recommendations are provided late in the hospitalization process, after most clinical decisions have been made, which contradicts the goal of proactively assisting decision-making.

We found that doctors make clinical decisions every day based on careful consideration of the historical interaction information between patients and various medical items (such as lab test items and drugs), thereby judging the patient's current condition and then determining which medical items the patient needs

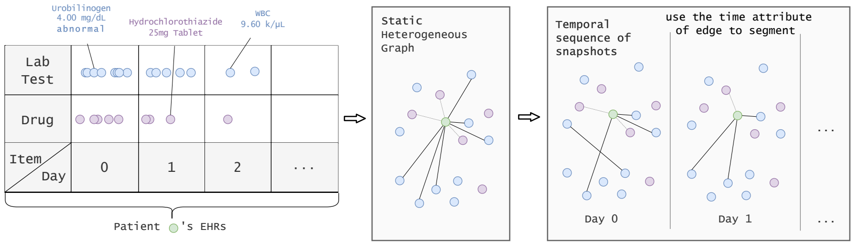


Fig. 2: Left to Right: the typical EHRs data; the static heterogeneous graph constructed from EHRs data; and the temporal sequence of snapshots obtained by further segmenting edges using their time attributes.

next, from the current set of available medical items. Hence, compared to machine translation, this process is more like the user-item recommendation task in the recommendation system than the machine translation task. Besides, by analyzing a patient's Electronic Health Records (EHRs), we can see that the historical interaction information is interconnected (Fig.2 left part). For example, if tests show an infection, appropriate drugs are prescribed; and if a drug has side effects, further tests monitor the patient's response. This indicates that EHRs data contains valuable, evolving, heterogeneous graph-structure semantic information, then, by effectively utilizing it, we can enhance clinical decision-making and improve recommendations for patient care [31,18].

Based on these insights, we explicitly build the patient's hospitalization process recorded by corresponding EHRs to heterogeneous dynamic graph-structured data, which is represented as a series of static graph snapshots (Fig.2 right part, like the frames in an old film reel). These graph snapshots are called discrete-time dynamic graphs, a form of dynamic graph [18]. Compared with static graphs, dynamic graphs can better reflect the dynamic change process of the system; this fits the characteristics of the patient's hospitalization process and is suitable for modeling the patient's condition representation and its development process over different days.

For effectively uncovering and utilizing the rich semantic information in constructed discrete-time dynamic graphs, we used GNNs (graph neural networks) to process each static graph snapshot. GNNs can learn the relationships between entities while taking into account the attribute information of the entities [5,40]. Through multi-layer GNNs, we fuse various information in the graph to generate vectors representing the patient's condition on different days. Then, we use the additive attention mechanism [2] to interact dependencies between historical condition representations and target medical items. Finally, the prediction layer calculates the similarity between the paid-attention condition representation vector and the target item representation vector to infer the score of the relationship edge (the score at which decision-making occurs), providing references for doctors to assist clinical decision-making.

Experiments demonstrate the effectiveness of the dynamic graph representation learning method we proposed, offering significant reference value and practical utility. Moreover, this work provides a new perspective for future research on methods aimed at assisting clinical decision-making. We highlight our contribution as follows:

- **Reconceptualization of Clinical Decision-Making Assist Recommendation**: we redefine the clinical decision-making assistance problem as a recommendation system task rather than a machine translation task. This new perspective aligns more closely with the real-world scenario of dynamic interactions between patients and various medical items, addressing limitations such as limited patient coverage and delayed recommendation timing in previous works.
- **Innovative Use of Dynamic Heterogeneous Graphs**: we propose modeling Electronic Health Records (EHRs) as discrete-time dynamic graph structures and utilize Graph Neural Networks (GNNs) to process this data. This approach effectively captures the complex and evolving interactions between patients and various medical items, providing a more accurate and detailed representation of patient condition changes to enhance clinical decision-making assistance.

2 Related Works

2.1 Graph-Structured Data and Graph Neural Networks

Research on graph-structured data has traditionally focused on homogeneous graphs and has produced influential algorithms like PageRank [3], Graph Kernels [35], Random-Walk [26,8]. However, real-world graphs are typically heterogeneous, with different types of nodes and relations [31]. For example, in a healthcare network, nodes can be patients, doctors, laboratory tests, diseases, drugs, treatments, and various types of relations (edges) between them. Currently, there is a transition from static graphs to dynamic graphs in the research trend of graph-structured data, as many real-world applications involve dynamic graphics [18].

Graph Neural Networks (GNNs) have become popular for modeling graph-structured data due to their inherent flexibility to model the underlying systems better [5]. Although originally designed for homogeneous graphs, modern GNNs like GCN [19], GAT [34], GraphSAGE [10], GIN [40] can be adapted for use with heterogeneous graphs through slight modifications to their neighborhood aggregation mechanisms[4]. With GNNs, we can efficiently construct node embeddings to uncover rich knowledge within heterogeneous graphs.

[4] Perform through the `to_hetero` function in `PyG` library.

2.2 Graph Methods for Recommendation

In recent years, graph-based methods have gained significant traction in the field of recommendation systems due to their ability to naturally model complex relationships and interactions among users, items, and additional contextual information. Traditional recommendation approaches, such as collaborative filtering and matrix factorization, often struggle to capture higher-order connectivity and intricate dependencies present in real-world data. Graph-based methods, on the other hand, leverage the power of graph structures to represent these multifaceted relationships, enabling more accurate and personalized recommendations. Works like GC-MC [33], NGCF [37], LightGCN [11], and SGL [38] adapt GCN [19] to the user-item interaction graph and capture CF signals in high-hop neighbors for recommendation. Our approach shares this spirit of using Graph Neural Networks (GNNs) to aggregate multi-hop neighborhood information and distill semantic insights from user-item interactions. However, it takes a critical step further by decomposing the interaction graph along the temporal dimension. Rather than merging interactions from various time points into a single, static heterogeneous graph, our method constructs a sequence of graph snapshots that more precisely captures changes over time. This discrete-time dynamic graph representation lets us model how user preferences, item characteristics, and their relationships evolve, enabling the system to adapt to shifting behaviors and trends. Through this temporal decomposition, our model preserves important chronological cues that might otherwise be lost, offering greater flexibility and finer-grained insights compared to traditional GNNs-based recommendation algorithms.

3 Method

3.1 Modeling Hospitalization Process to Heterogeneous Dynamic Graph-Structured Data

Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ denote the static heterogeneous graph of EHRs data (see mid part of Fig.2), where $\mathcal{V}$ is the nodes set of all types of nodes, and $\mathcal{E}$ is the edges set of all the existent relations between these typed nodes in $\mathcal{V}$. $\mathcal{V}$ and $\mathcal{E}$ can be divided into multiple subsets. For example, the $\mathcal{V}$ can be divided into $\{\mathcal{V}^{\mathcal{P}}, \mathcal{V}^{\mathcal{T}}, \mathcal{V}^{\mathcal{D}}\}$[5], representing patient, laboratory test, and drug nodes set respectively[6]; the $\mathcal{E}$ can be divided into $\{\mathcal{E}^{\mathcal{P}-\mathcal{D}}, \mathcal{E}^{\mathcal{P}-\mathcal{T}}\}$, representing `patient-took-drug`, `patient-did-test` relations respectively. Then, every node v in $\mathcal{V}$ has node features f^v; also, every edge e in $\mathcal{E}$ has multiple kinds of edge features f^e (e.g. **the time attribute that reflects when this relation occurred**, attributes that reflect the severity of lab test results or drug dosage). Through segmenting the edges by

[5] We use a single patient node and nodes for all other types of medical items (drugs, laboratory tests) to construct a heterogeneous graph $\mathcal{G}$ representing the patient's hospitalization process.

[6] Each hospitalization is treated as an independent patient.

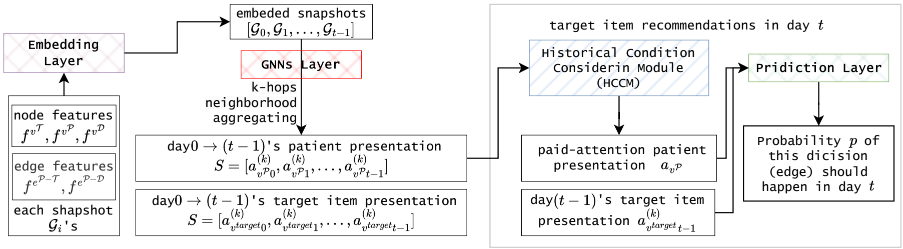

Embedding Layer
node features $f^{v^{\mathcal{T}}}, f^{v^{\mathcal{P}}}, f^{v^{\mathcal{D}}}$
edge features $f^{e^{\mathcal{P}-\mathcal{T}}}, f^{e^{\mathcal{P}-\mathcal{D}}}$
each shapshot $\mathcal{G}_i$'s
embeded snapshots $[\mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_{t-1}]$
GNNs Layer
k-hops neighborhood aggregating
day0 $\rightarrow (t-1)$'s patient presentation $S = [a^{(k)}_{v^{\mathcal{P}}0}, a^{(k)}_{v^{\mathcal{P}}1}, \ldots, a^{(k)}_{v^{\mathcal{P}}t-1}]$
day0 $\rightarrow (t-1)$'s target item presentation $S = [a^{(k)}_{v_{target_0}}, a^{(k)}_{v_{target_1}}, \ldots, a^{(k)}_{v_{target_{t-1}}}]$
target item recommendations in day t
Historical Condition Considerin Module (HCCM)
paid-attention patient presentation $a_{v^{\mathcal{P}}}$
day$(t-1)$'s target item presentation $a^{(k)}_{v_{target}t-1}$
Pridiction Layer
Probability p of this dicision (edge) should happen in day t

Fig. 3: An illustration of our dynamic graph representation learning framework. The embedding layer transforms original features for each graph snapshot. The GNNs layer, via message passing, aggregates neighborhoods and updates embeddings. For day t, the target item's embedding is taken from the $\mathcal{G}_{t-1}$, and attention mechanism captures its temporal dependencies with the patient's historical embeddings. Finally, the prediction layer computes an inner product of these embeddings and applies a sigmoid to yield the probability.

the time attribute, each static heterogeneous graph $\mathcal{G}$ is divided into a temporal snapshots sequence $[\mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_{t-1}]$. Thus the hospitalizations of patients are modeled on a day-by-day basis.

3.2 Model Architecture

Embedding Layer Since the original data of each type of node / edge features f^v / f^e are composed of many high-dimensional token-type variables and float-type variables, we use the embedding layer to convert them into low-dimensional dense vector representations. Specifically, we collect the dictionary dimensions of each token type variable column into a list $D = [\mathrm{d}_0, \mathrm{d}_1, \ldots, \mathrm{d}_{n-1}]$, then calculate the offset list $O = [o_0, o_1, \ldots, o_{n-1}] = [0, \sum_{i=0}^{0} \mathrm{d}_i, \sum_{i=0}^{1} \mathrm{d}_i, \ldots, \sum_{i=0}^{n-2} \mathrm{d}_i]$, and then use a unified Embedding table $ET \in \mathbb{R}^{\sum D \times h_e}$ to provide conversions for these token type variable columns (h_e is embedding size); when looking up, the original value x of the i-th column will be added with the corresponding offset o_i to obtain the embedding vector $e \in \mathbb{R}^{1 \times h_e}$ of the $(x + o_i)$-th row of the Embedding table ET. If there are float type columns, they are fed into a fully connected layer so that the final dimension becomes h_e; then, a new dimension will be inserted before the last dimension to facilitate subsequent concatenation with the sparse embedding representation obtained from the token type variable columns to get the final embedding representation.

Graph Semantic Information Aggregating Module Specifically, for the temporal snapshot sequences $[\mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_{t-1}]$ that represent a single hospitalization process, modern GNNs follow a neighborhood aggregation strategy, where we iteratively update the representation of a node by aggregating representations of its neighbors[40]. After k iterations of aggregation, a node's representation captures the structural information within its k-hop network neighborhood. Formally, the k-th layer of a GNNs is:

$$a_v^{(k)} = f_{\mathrm{AGG}}^{(k)}\big(\{h_u^{(k-1)}, f^{e^{u-v}} : u \in \mathcal{N}(v)\}\big) \tag{1}$$

$$h_v^{(k)} = f_{\mathrm{CMB}}^{(k)}\big(h_v^{(k-1)}, a_v^{(k)}\big) \tag{2}$$

where $h_v^{(k)}$ is the feature vector of node v at the k-th iteration/layer, $f^{e^{u-v}}$ is the feature vector of edge between node u and v, $h_v^{(0)}$ is initial feature vector of corresponding node feature f^v. The choice of the aggregate function $f_{\mathrm{AGG}}^{(k)}$ and the combine function $f_{\mathrm{COM}}^{(k)}$ in GNNs is crucial and diverse, and the different choices composing to different specific GNNs. The updated representation of node $a_v^{(k)}$ is where the rich semantics lie in each snapshot.

It is worth mentioning that, prior to the aforementioned neighborhood aggregation process, we stack the adjacency matrices of each snapshot $\mathcal{G}_i$ in a diagonal fashion (creating a giant graph that holds multiple isolated subgraphs), and their node and edge features are concatenated accordingly[7].

Historical Condition Considering Module Prior to determining the probability of whether a specific medical item (drug or lab test) should be recommended, we aim for the model to emulate a physician-like synthesis of the patient's condition progression and the characteristics of the target item (for instance, determining the necessity of prescribing a medication on a particular day might require consideration of certain laboratory test results from previous days). Therefore, we employ an additive attention mechanism [2,45] between the sequence of historical patient condition representations $S = [a^{(k)}_{v^{\mathcal{P}}{}_0}, a^{(k)}_{v^{\mathcal{P}}{}_1}, \dots, a^{(k)}_{v^{\mathcal{P}}{}_{t-1}}]$ obtained via a Graph Neural Networks (GNNs) and the representation of the target item $a^{(k)}_{v^{\mathcal{D}}{}_{t-1}}|a^{(k)}_{v^{\mathcal{T}}{}_{t-1}}$ pending inference on that day. By doing so, the model can apply different attention weights to the patient's historical condition representation sequence according to the characteristics of the target medical item (drug or laboratory test item) that needs to be judged in the current clinical decision. Formally, the query $q \in \mathbb{R}^{1\times h_e}$ is $a^{(k)}_{v^{\mathcal{D}}{}_{t-1}}|a^{(k)}_{v^{\mathcal{T}}{}_{t-1}}$, the key-value pairs $k = v = S \in \mathbb{R}^{t\times h_e}$, the attention score a_{ij} of i-th query q_i and j-th key k_j is calculated by:

$$e_{ij} = v^T \cdot \tanh(W_q \cdot q_i + W_k \cdot k_j) \tag{3}$$

$$a_{ij} = \mathrm{softmax}(e_{ij}) = \frac{\exp(e_{ij})}{\sum_k \exp(e_{ik})} \tag{4}$$

where W_q and W_k are learnable weight matrices for linear transformations, tanh is the hyperbolic tangent activation function used to introduce nonlinearity. Then, apply these attention score to value v for obtaining paid-attention pa-

[7] https://pytorch-geometric.readthedocs.io/en/latest/advanced/batching.html

tient condition represent $a_{v^{\mathcal{P}}}$ by:

$$a_{v^{\mathcal{P}}} \in \mathbb{R}^{1 \times h_e} = \text{Attention}(q_i, k, v) = \sum_j a_{ij}(W_v \cdot v_j) \tag{5}$$

Prediction Layer To calculate the probability of a clinical decision occurring on day t (graph snapshot $\mathcal{G}_t$), we perform a dot product between the node representation of the target medical item $a_v^{(k)}{}_{t-1}$ from $\mathcal{G}_{t-1}$, and the patient condition representation vector $a_{v^{\mathcal{P}}}$ from $[\mathcal{G}_0, \mathcal{G}_1, \dots, \mathcal{G}_{t-1}]$, which is weighted by attention specific to the target medical item. This approach aligns with the common practice in recommendation models, where the similarity between two vectors is computed, then passed through a `sigmoid` layer to map the result into a probability range of $[0, 1]$. In other words, the probability $p \in [0, 1]$ is calculated using the following formula:

$$p = \texttt{sigmoid}(a_v^{(k)}{}_{t-1} \cdot a_{v^{\mathcal{P}}}) = \frac{1}{1 + e^{(a_v^{(k)}{}_{t-1} \cdot a_{v^{\mathcal{P}}})}} \tag{6}$$

4 Experiments

4.1 Experimental Settings

Table 1: Statistics of the experimented data.

MIMIC	Item	Total	#Interactions	Avg. Interactions	Sparsity
III	drug	4294	10,189,840	205.49 per patient	95.21%
	lab test	753	15,186,986	261.25 per patient	63.20%
IV	drug	5,967	34,631,582	94.78 per patient	98.41%
	lab test	861	45,161,804	132.87 per patient	84.57%

#Interactions is the number of total records between patients and current item.
Sparsity $= 1 - \frac{\#\text{Interactions}}{\#\text{user} \times \#\text{item}}$

Dataset We use the MIMIC-III (Medical Information Mart for Intensive Care) dataset, which is a large, single-center database comprising information relating to patients admitted to critical care units at a large tertiary care hospital[16]; and MIMIC-IV dataset, which is a relational database containing real hospital stays for patients admitted to a tertiary academic medical center, containing comprehensive information for each patient while they were in the hospital: laboratory measurements, medications administered, vital signs documented, and so on[15]. MIMIC-IV builds upon the success of MIMIC-III, and incorporates numerous improvements over MIMIC-III. We conducted a statistical analysis for the expariment data processed from these two datasets, and the results are presented in Table 1.

Model Implement and Training we use the `PyG` and `PyTorch` library to implement our model, which is trained using 2:1[8] negative sample strategy. We use AdamW[23] optimizer with 0.0003 initial learning rate decay by `CosineAnnealingLR` schedule, to optimize the `BCEWithLogitsLoss` loss on training set for at most 10 epochs[9]:

$$L = \frac{1}{N} \sum_{i} -\big[y_i \cdot \log(x_i) + (1 - y_i) \cdot \log(1 - x_i)\big] \tag{7}$$

where N represents the number of samples, y_i denotes the ground true class of the i-th sample, x_i represents the probability that the model predicts the i-th sample as belonging to the positive class. Both our models and all compared models' embedding dimensions and hidden dimensions are kept the same by 10 and 256, respectively. We use `ecs.gn7i-c8g1.2xlarge (8 vCPU, 30 GiB, NVIDIA A10 * 1)` instances of Alibaba Cloud's artificial intelligence platform (PAI) and the AutoDL computing power rental platform's instance with specifications of `(16 vCPU, 120 GiB RAM, NVIDIA RTX 4090 * 1)` to conduct experiments.

Besides, whenever our model processes 10% of the samples in the training set, we switch to validation mode to calculate the model's loss performance on the validation set. If a lower validation loss is observed, we save the current parameter state of the model as a checkpoint. Subsequently, when evaluating metrics on the test set, we use the model checkpoint corresponding to the minimum validation loss.

Compared Methods To evaluate the performance level of our proposed model, we selected the following recommendation models which are commonly used in the industry [10], categorized into 4 types, as baselines for comparison:

General recommendation methods primarily follow the collaborative filtering (CF) paradigm to model user-item interactions. NeuMF [12] combines neural networks with matrix factorization, using neural embeddings to enhance recommendation performance. SimpleX [24] applies a cosine contrastive loss with extensive negative sampling to improve both training efficiency and performance. DiffRec [36] uses denoising principles from diffusion models to capture more complex user-item interaction signals, extending traditional CF approaches to account for higher degrees of uncertainty in user preferences.

Context-aware recommendation methods incorporate various mechanisms to perceive and capture the contextual relationships between user features and item features. DSSM [14] employs a deep neural network to project queries and documents into the same low-dimensional space, effectively modeling semantic relevance. DeepFM [9] extends Factorization Machines [27] by including a

[8] 2:1 means that, for each positive sample (real edge), we randomly generate 2 negative samples (fake edges).

[9] Empirical tests show that 10 epochs are enough for each model's parameters to converge, and the training time isn't too long to be unbearable.

[10] we utilize `RecBole` [41] library to run experiments https://github.com/RUCAIBox/RecBole

Table 2: Performance of different models on MIMIC-III and IV dataset.

Model	AUC		AP		Accuracy		Precision		Recall		F1-score	
	III	IV	III	IV	III	IV	III	IV	III	IV	III	IV
NeuMF	0.8839	0.9304	0.8290	0.8823	0.7999	0.8248	0.9293	0.9282	0.4325	0.5142	0.5903	0.6618
SimpleX	0.9742	0.9758	0.9468	0.9477	0.9256	0.9267	0.8691	0.8686	0.9144	0.9191	0.8912	0.8931
DiffRec	0.9794	0.9777	0.9728	0.9712	0.7575	0.7446	0.9998	0.9998	0.2725	0.2338	0.4283	0.3790
DSSM	0.9401	0.9418	0.8722	0.8699	0.8746	0.8797	0.7720	0.7787	0.8851	0.8926	0.8247	0.8318
DeepFM	0.9562	0.9605	0.9211	0.9261	0.8888	0.8891	0.7952	0.7928	0.8976	0.9033	0.8433	0.8445
EulerNet	0.9578	0.9583	0.9216	0.9221	0.8872	0.8824	0.7927	0.7780	0.8957	0.9056	0.8411	0.8370
DIN	0.9865	0.9865	0.9729	0.9706	0.9483	0.9449	0.9218	0.9096	0.9231	0.9269	0.9224	0.9181
SASRec	0.9859	0.9867	0.9718	0.9711	0.9306	0.9408	0.8445	0.8748	0.9706	0.9598	0.9032	0.9153
CORE	0.9507	0.9463	0.8991	0.8892	0.5421	0.5828	0.4211	0.4441	0.9982	0.9988	0.5924	0.6148
NGCF	0.9837	0.9854	0.9652	0.9677	0.8669	0.9014	0.7162	0.7751	0.9951	0.9922	0.8329	0.8703
LightGCN	0.9756	0.9815	0.9509	0.9606	0.7587	0.7544	0.5804	0.5759	0.9966	0.9983	0.7336	0.7304
SGL	0.9777	0.9740	0.9566	0.9482	0.8987	0.8940	0.7817	0.7738	0.9658	0.9638	0.8641	0.8584
Ours	0.9954	0.9943	0.9917	0.9897	0.9703	0.9646	0.9677	0.9531	0.9410	0.9392	0.9542	0.9461

deep neural network that learns both manually crafted cross features and high-order nonlinear features automatically. EulerNet [32] leverages Euler's formula for adaptive feature interaction learning.

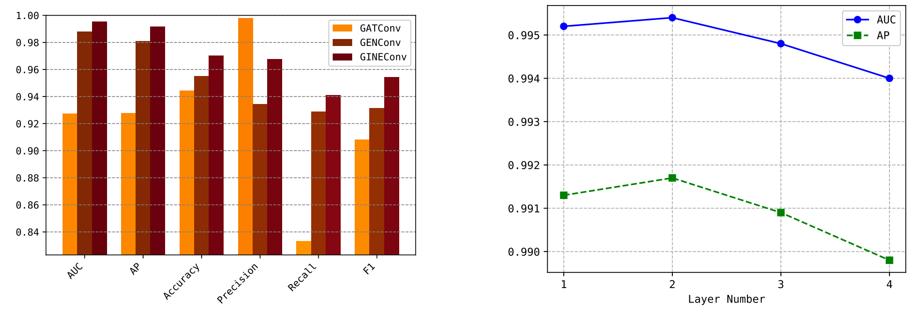
Sequential recommendation methods typically infer a user's temporal preferences by analyzing sequences of recently interacted items. DIN [45] uses local activation units to adaptively learn user-interest representations for a given advertisement, effectively capturing diverse user behaviors. SASRec [17] employs a self-attention mechanism to model long-term dependencies in user behavior. CORE [13] further addresses inconsistent predictions in session-based scenarios by unifying the representation space for both encoding and decoding.

GNNs-based recommendation methods use graph neural networks (GNNs) to exploit the structure of user-item graphs and extract high-order relationships. NGCF [37] constructs a bipartite graph of users and items, repeatedly propagating embeddings through a GNNs to capture in-depth adjacency and collaborative signals. LightGCN [11] implements a simpler GCN approach by retaining only the neighborhood aggregation process while removing feature transformations and nonlinear activations, thereby improving training efficiency without compromising performance. SGL [38] integrates self-supervised learning tasks to reduce the impact of noisy signals around high-degree nodes, enhancing the model's ability to represent genuine user preferences more accurately.

4.2 Performance Comparison

We evaluate performance of all models on the testing set using metrics including `AUC(ROCAUC)`, `AP`, `Accuracy`, `Precision`, `Recall`, `F1` [11]. Table 2 shows the results of medication recommendation task on MIMIC-III and MIMIC-IV dataset. As `AUC` can be interpreted as the probability that a randomly chosen positive instance is ranked higher by the model than a randomly chosen negative instance,

[11] For calculating `Accuracy`, `Precision`, `Recall`, `F1`, we fix probability threshold as 0.5.



(a) Performance comparison of different GNNs implementations.

(b) Performance comparison chart of GNNs with different numbers of layers.

Fig. 4: Results of parameter experiments.

and it is proven as an unbiased and stable metric under different negative sampling settings [20], so we mainly focus model performance on the `AUC` metric.

Results in Table 2 obviously shows that our dynamic graph representation learning method has better performance than other categories of baselines. We believe that, by capturing the dynamic evolution between snapshots, we refined the graph modeling, reducing noise across different days, facilitated the GNNs' learning of useful semantic information within each snapshot, resulting in improved performance. Besides, among the general recommendation models, SimpleX and DiffRec significantly outperform the traditional neural collaborative filtering method NeuMF, due to the use of cosine contrastive loss and diffusion denoising design, respectively. All context-aware models act well by incorporating the contextual relationships between user features and item features. Sequential recommendation models are further improved by leveraging the relationship between the current and historical drug decision sequence. GNNs-based models effectively aggregate historical interactions within patient-drug graphs, also achieving strong performance.

4.3 Parameter Experiments

We conducted parameter experiments on the GNNs module responsible for extracting semantic information from dynamic graph snapshots. Experiments involved examining different neighborhood aggregation methods and the impact of varying the number of GNNs layers on performance, and the results are presented in Fig.4. In terms of the primary metric, `AUC`, we observe from Fig.4a that the performance ranks as `GATConv` $<$ `GENConv` $<$ `GINEConv`, with `GATConv` performing notably worse while the latter two show better and more similar

Table 3: Experiment results of ablation study

GNNs	$\mathcal{E}^{\mathcal{P}-\mathcal{D}}$	$\mathcal{E}^{\mathcal{P}-\mathcal{T}}$	HCCM	AUC	AP	ACC	Prec	Recall	F1
`GATConv`	✓	✓	✓	0.9273	0.9277	0.9445	0.9981	0.8330	0.9081
	✓		✓	0.9258	0.9267	0.9445	0.9981	0.8330	0.9081
	✓	✓		0.9262	0.9270	0.9445	0.9981	0.8330	0.9081
`GENConv`	✓	✓	✓	0.9878	0.9811	0.9551	0.9343	0.9289	0.9316
	✓		✓	0.9867	0.9794	0.9443	0.8925	0.9443	0.9176
	✓	✓		0.9862	0.9799	0.9485	0.9119	0.9337	0.9227
`GINEConv`	✓	✓	✓	0.9954	0.9917	0.9703	0.9677	0.9410	0.9542
	✓		✓	0.9951	0.9913	0.9691	0.9610	0.9444	0.9526
	✓	✓		0.9956	0.9920	0.9707	0.9670	0.9431	0.9549

$\mathcal{E}^{\mathcal{P}-\mathcal{D}}$ and $\mathcal{E}^{\mathcal{P}-\mathcal{T}}$ represent `patient-took-drug`, `patient-did-test` relations respectively
HCCM stands for Historical Condition Considering Module.

performance [12] for details. We believe that the reason for these results lies in the differences in how these three GNNs implementations aggregate neighborhood information. `GATConv` stacks masked self-attention layers, allowing nodes to apply different levels of attention to their neighborhood information [34]. Both `GENConv` and `GINEConv` use an MLP to map the aggregated node features, but they differ in their aggregation strategies: `GENConv` uses a softmax function [22], while `GINEConv` directly sums the features [40], leading to performance differences.

Moreover, we also explored the impact of different numbers of layers, or different distances of neighborhood aggregation (k-hops), on the performance when using `GINEConv`. As shown in Fig.4b, the best results are achieved with 2 layers, while performance declines with more layers. This outcome is related to the core mechanism of k-layer GNNs aggregating k-hops neighborhood information. In our constructed dynamic graph snapshots, since the patient-drug $\mathcal{E}^{\mathcal{P}-\mathcal{D}}$ and patient-lab test $\mathcal{E}^{\mathcal{P}-\mathcal{T}}$ relationship edges are single-hop, excessive layers can lead to an over-smoothing problem [4] (akin to the famous 'Six Degrees of Separation'), resulting in decreased performance.

4.4 Ablation Study

We conducted an ablation study on how two implementation details/modules in our method affect performance. The first factor is the use of multiple patient-medical item relationship edges (corresponding to clinical decisions, prescribing drugs, or ordering tests) within dynamic graph snapshots. The second factor is the use of the historical condition considering module (HCCM). As shown in Table 3, we can see that incorporating patient-lab test relationship edges $\mathcal{E}^{\mathcal{P}-\mathcal{T}}$

[12] `GATConv`, `GENConv`, `GINEConv` are three different GNNs implements that we choose using `PyG` library's GNNs cheatsheet, see https://pytorch-geometric.readthedocs.io/en/latest/cheatsheet/gnn_cheatsheet.html

alongside patient-drug relationship edges $\mathcal{E}^{\mathcal{P}-\mathcal{D}}$ in the dynamic snapshot graph enhances performance. Moreover, our method outperforms baselines even when only using patient-drug interactions.

An interesting phenomenon shown in Table 3 is that while the use of HCCM benefits performance for GNNs implementations using `GATConv` and `GENConv`, it slightly decreases performance with `GINEConv`. We believe one possible reason is that the latest dynamic snapshot graph already captures the dynamic changes in the patient's condition (medications used and tests ordered are based on the historical changes in the patient's condition up to that day), then `GINEConv`'s aggregation mechanism could effectively model this already, and the introduction of HCCM adds model complexity, potentially leading to over-fitting and slightly reduced performance on the test set.

5 Conclusion

In this study, we introduced a novel approach to assist clinical decision-making that leverages dynamic graph representation learning, by reconceptualizing the clinical decision-making process as a recommendation system task that mirrors real-world interactions between patients and medical items. When performing a specific task such as drug recommendation for a patient, the physician can first follow the "recall" stage in recommendation system by applying various rules (for instance, selecting frequently used drugs, similar drugs, drugs used in the same department, or drugs prescribed for the same condition) to generate a candidate list of drugs that require decision support. This list is then fed into the model and model uses the dynamic graph of the patient's hospitalization process so far to extract node embeddings that represent the patient's current condition, and it also retrieves the corresponding node embeddings for each candidate medication. Based on these embeddings, the model computes a probability indicating whether the medication is needed at this point in time. Finally, the model sorts all candidate drugs in descending order of this probability and presents the list to the physician to assist in drug-related decision-making.

Our experimental results demonstrate the superior performance of our approach compared to traditional methods, highlighting the significant reference value and practical utility of dynamic graph representation learning in clinical settings. Parameters and ablation studies further investigate the contributions and importance of various components of the method to performance. This work not only offers a fresh perspective on clinical decision-making but also opens new avenues for research in dynamic graph modeling, promising more accurate, timely recommendations that enhance patient outcomes and healthcare efficiency.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China under Grants (62201072, 62471055, 62321001, 62101064, U23B2001, 62171057), the Ministry of Education and China Mobile Joint Fund (MCM20200202, MCM20180101), the Fundamental Research Funds for the Central Universities (2024PTB-004).

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. Bahdanau, D.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
3. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer networks and ISDN systems **30**(1-7), 107–117 (1998)
4. Chen, D., Lin, Y., Li, W., Li, P., Zhou, J., Sun, X.: Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 3438–3445 (2020)
5. Daigavane, A., Ravindran, B., Aggarwal, G.: Understanding convolutions on graphs. Distill **6**(9), e32 (2021)
6. Doherty, T.S., Carroll, A.E.: Believing in overcoming cognitive biases. AMA Journal of Ethics **22**(9), 773–778 (2020)
7. Feng, Y., Lv, F., Shen, W., Wang, M., Sun, F., Zhu, Y., Yang, K.: Deep session interest network for click-through rate prediction. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence. p. 2301–2307. IJCAI'19, AAAI Press (2019)
8. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864 (2016)
9. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: Deepfm: a factorization-machine based neural network for ctr prediction. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. p. 1725–1731. IJCAI'17, AAAI Press (2017)
10. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in neural information processing systems **30** (2017)
11. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M.: Lightgcn: Simplifying and powering graph convolution network for recommendation. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 639–648. SIGIR '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3397271.3401063, https://doi.org/10.1145/3397271.3401063
12. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web. p. 173–182. WWW '17, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE (2017). https://doi.org/10.1145/3038912.3052569, https://doi.org/10.1145/3038912.3052569
13. Hou, Y., Hu, B., Zhang, Z., Zhao, W.X.: Core: Simple and effective session-based recommendation within consistent representation space (2022), https://arxiv.org/abs/2204.11067
14. Huang, P.S., He, X., Gao, J., Deng, L., Acero, A., Heck, L.: Learning deep structured semantic models for web search using clickthrough data. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management. p. 2333–2338. CIKM '13, Association for Computing Machinery, New York, NY, USA (2013). https://doi.org/10.1145/2505515.2505665, https://doi.org/10.1145/2505515.2505665

15. Johnson, A.E., Bulgarelli, L., Shen, L., Gayles, A., Shammout, A., Horng, S., Pollard, T.J., Hao, S., Moody, B., Gow, B., et al.: Mimic-iv, a freely accessible electronic health record dataset. Scientific data **10**(1), 1 (2023)
16. Johnson, A.E., Pollard, T.J., Shen, L., wei H. Lehman, L., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L.A., Mark, R.G.: Mimic-iii, a freely accessible critical care database. Scientific Data **3** (2016), https://doi.org/10.1038/sdata.2016.35
17. Kang, W.C., McAuley, J.: Self-attentive sequential recommendation. In: 2018 IEEE International Conference on Data Mining (ICDM). pp. 197–206 (2018). https://doi.org/10.1109/ICDM.2018.00035
18. Kazemi, S.M., Goel, R., Jain, K., Kobyzev, I., Sethi, A., Forsyth, P., Poupart, P.: Representation learning for dynamic graphs: A survey. The Journal of Machine Learning Research **21**(1), 2648–2720 (2020)
19. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
20. Krichene, W., Rendle, S.: On sampled metrics for item recommendation. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 1748–1757 (2020)
21. Le, H., Tran, T., Venkatesh, S.: Dual memory neural computer for asynchronous two-view sequential learning. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 1637–1645 (2018)
22. Li, G., Xiong, C., Thabet, A., Ghanem, B.: Deepergcn: All you need to train deeper gcns (2020), https://arxiv.org/abs/2006.07739
23. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
24. Mao, K., Zhu, J., Wang, J., Dai, Q., Dong, Z., Xiao, X., He, X.: Simplex: A simple and strong baseline for collaborative filtering. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. p. 1243–1252. CIKM '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3459637.3482297, https://doi.org/10.1145/3459637.3482297
25. Neale, G., Hogan, H., Sevdalis, N.: Misdiagnosis: analysis based on case record review with proposals aimed to improve diagnostic processes. Clinical Medicine **11**(4), 317 (2011)
26. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 701–710 (2014)
27. Rendle, S.: Factorization machines. In: 2010 IEEE International Conference on Data Mining. pp. 995–1000 (2010). https://doi.org/10.1109/ICDM.2010.127
28. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)
29. Sackett, D.L., Rosenberg, W.M., Gray, J.M., Haynes, R.B., Richardson, W.S.: Evidence based medicine: what it is and what it isn't (1996)
30. Shang, J., Xiao, C., Ma, T., Li, H., Sun, J.: Gamenet: Graph augmented memory networks for recommending medication combination. In: proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 1126–1133 (2019)
31. Sun, Y., Han, J.: Mining heterogeneous information networks: a structural analysis approach. Acm Sigkdd Explorations Newsletter **14**(2), 20–28 (2013)
32. Tian, Z., Bai, T., Zhao, W.X., Wen, J.R., Cao, Z.: Eulernet: Adaptive feature interaction learning via euler's formula for ctr prediction. In: Proceedings of the

46th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 1376–1385. SIGIR '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3539618.3591681, https://doi.org/10.1145/3539618.3591681

33. Van Den Berg, R., Thomas, N.K., Welling, M.: Graph convolutional matrix completion. arXiv preprint arXiv:1706.02263 **2**(8), 9 (2017)
34. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
35. Vishwanathan, S.V.N., Schraudolph, N.N., Kondor, R., Borgwardt, K.M.: Graph kernels. Journal of Machine Learning Research **11**, 1201–1242 (2010)
36. Wang, W., Xu, Y., Feng, F., Lin, X., He, X., Chua, T.S.: Diffusion recommender model. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 832–841. SIGIR '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3539618.3591663, https://doi.org/10.1145/3539618.3591663
37. Wang, X., He, X., Wang, M., Feng, F., Chua, T.S.: Neural graph collaborative filtering. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 165–174. SIGIR'19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3331184.3331267, https://doi.org/10.1145/3331184.3331267
38. Wu, J., Wang, X., Feng, F., He, X., Chen, L., Lian, J., Xie, X.: Self-supervised graph learning for recommendation. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 726–735. SIGIR '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3404835.3462862, https://doi.org/10.1145/3404835.3462862
39. Wu, R., Qiu, Z., Jiang, J., Qi, G., Wu, X.: Conditional generation net for medication recommendation. In: Proceedings of the ACM Web Conference 2022. pp. 935–945 (2022)
40. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)
41. Xu, L., Tian, Z., Zhang, G., Zhang, J., Wang, L., Zheng, B., Li, Y., Tang, J., Zhang, Z., Hou, Y., Pan, X., Zhao, W.X., Chen, X., Wen, J.: Towards a more user-friendly and easy-to-use benchmark library for recommender systems. In: SIGIR. pp. 2837–2847. ACM (2023)
42. Yang, C., Xiao, C., Glass, L., Sun, J.: Change matters: Medication change prediction with recurrent residual networks. In: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021 (2021)
43. Yang, C., Xiao, C., Ma, F., Glass, L., Sun, J.: Safedrug: Dual molecular graph encoders for safe drug recommendations. In: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021 (2021)
44. Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., Yan, Y., Jin, J., Li, H., Gai, K.: Deep interest network for click-through rate prediction. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 1059–1068 (2018)
45. Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., Yan, Y., Jin, J., Li, H., Gai, K.: Deep interest network for click-through rate prediction. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. p. 1059–1068. KDD '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3219819.3219823, https://doi.org/10.1145/3219819.3219823