Interaction-aware Drug Package Recommendation via Policy Gradient

ZHI ZHENG, CHAO WANG, TONG XU, DAZHONG SHEN, and PENGGANG QIN,
School of Computer Science and Technology, University of Science and Technology of China
XIANGYU ZHAO, City University of Hong Kong
BAOXING HUAI, Huawei Technologies
XIAN WU, Tencent
ENHONG CHEN, School of Computer Science and Technology, University of Science and Technology of China

3

Recent years have witnessed the rapid accumulation of massive electronic medical records, which highly support intelligent medical services such as drug recommendation. However, although there are multiple interaction types between drugs, e.g., synergism and antagonism, which can influence the effect of a drug package significantly, prior arts generally neglect the interaction between drugs or consider only a single type of interaction. Moreover, most existing studies generally formulate the problem of package recommendation as getting a personalized scoring function for users, despite the limits of discriminative models to achieve satisfactory performance in practical applications. To this end, in this article, we propose a novel end-to-end Drug Package Generation (DPG) framework, which develops a new generative model for drug package recommendation that considers the interaction effects between drugs that are affected by patient conditions. Specifically, we propose to formulate the drug package generation as a sequence generation process. Along this line, we first initialize the drug interaction graph based on medical records and domain knowledge. Then, we design a novel message-passing neural network to capture the drug interaction, as well as a drug package generator based on a recurrent neural network. In detail, a mask layer is utilized to capture the impact of patient condition, and the deep reinforcement learning technique is leveraged to reduce the dependence on the drug order. Finally, extensive experiments on a real-world dataset from a first-rate hospital demonstrate the effectiveness of our DPG framework compared with several competitive baseline methods.

CCS Concepts: • **Information systems** → **Data mining;**

Additional Key Words and Phrases: Drug recommendation, package recommendation, graph neural network, reinforcement learning

This is a substantially extended and revised version of Zheng et al. [77], which appears in Proceedings of the Web Conference 2021 (WWW'21). This research was partially supported by grants from the National Key Research and Development Program of China (Grant No. 2018YFB1402600), and the National Natural Science Foundation of China (Grant No. 62072423). Also, this research was partially supported by the Huawei-USTC Joint Innovation Program.

Authors' addresses: Z. Zheng, C. Wang, T. Xu (corresponding author), D. Shen, P. Qin, and E. Chen, School of Computer Science, University of Science and Technology of China; emails: {zhengzhi97, wdyx2012, sdz, qinpg}@mail.ustc.edu.cn, {tongxu, cheneh}@ustc.edu.cn; X. Zhao, City University of Hong Kong; email: xiangyuzhao1105@gmail.com; B. Huai, Huawei Technologies; email: huaibaoxing@huawei.com; X. Wu, Tencent; email: kevinxwu@tencent.com.

1046-8188/2023/01-ART3 $15.00
https://doi.org/10.1145/3511020

ACM Reference format:
Zhi Zheng, Chao Wang, Tong Xu, Dazhong Shen, Penggang Qin, Xiangyu Zhao, Baoxing Huai, Xian Wu, and Enhong Chen. 2023. Interaction-aware Drug Package Recommendation via Policy Gradient. *ACM Trans. Inf. Syst.* 41, 1, Article 3 (January 2023), 32 pages.
https://doi.org/10.1145/3511020

1 INTRODUCTION

With the growth of population and the intensification of population aging, people's demand for high-quality medical services continues to rise, and the pressure on the medical workers is increasing. Moreover, certain public health emergencies, such as the outbreak of COVID-19, will also have a significant impact on the medical system. Meanwhile, **artificial intelligence (AI)** technologies have shown enormous potential to reduce human labor. Therefore, if AI technologies could be effectively utilized to realize intelligent diagnosis and drug recommendation clinically, then it will greatly improve the overall quality of medical services.

Fortunately, with the popularization of information technology in the medical industry, **electronic medical records (EMRs)** have been widely used in major hospitals, which powerfully support downstream intelligent applications like medical image analysis [16, 35], medical text analysis [2, 40], and drug recommendation [45, 46, 55, 77]. However, prior arts may still fail to recommend drugs accurately due to the following reasons. First, most patients have only been recorded once or several times in the EMR database, which makes it hard to utilize conventional personalized recommendation methods based on user preference analysis. Second, it is crucial for the recommender system to consider both drug effect and the interaction between drugs at the same time, and give the patient a suitable **drug package**, which contains multiple drugs. Furthermore, there are multiple interaction types between drugs, e.g., synergism and antagonism, and the interaction of drugs may have different effects on different patients. However, most existing studies generally neglect the interaction between drugs or consider only a single type of interaction [46, 55], and these methods cannot capture the personalized effect of drug interaction for different patients either. Third, most existing studies on package recommendation [9, 10, 77] generally formulate the problem as getting a personalized scoring function for users, i.e., getting a discriminative model. However, discriminative models can only select suitable packages that already exist within the EMR database, which may fail to meet the needs of new patients. Therefore, there are limits for existing methods to achieve satisfactory performance in practical applications.

To tackle the above challenges, in this article, we propose a novel drug package recommendation framework named **Drug Package Generation (DPG)**. Following References [65, 77], we formulate the drug package generation problem as a sequence generation process, and we capture the influence of drug interaction based on **Message-passing Neural Network (MPNN)** and mask vectors. The rationale behind capturing the drug interaction is that the interaction between drugs will influence the effect of the drug package, and the impact of drug interaction on drug effect will be further affected by patient conditions. We illustrate this by a patient with kidney disease as shown in Figure 1. The drug package for this patient contains three drugs, respectively, pyridoxine, aztreonam, and cefuroxime. Cefuroxime is synergistic with the other two drugs, which can improve the effectiveness of the drug package. Torasemide is antagonistic with pyridoxine, so it is not included in the package. Furthermore, the combination of cefuroxime and gentamicin has a synergistic antibacterial effect, but at the same time it may increase nephrotoxicity, so it is not suitable for this patient. Along this line, we first collect drug interaction data from a public online medical knowledge base and divide drug pairs into three categories with the help of domain experts, respectively, No Interaction, Synergism, and Antagonism. Based on the interaction data, we

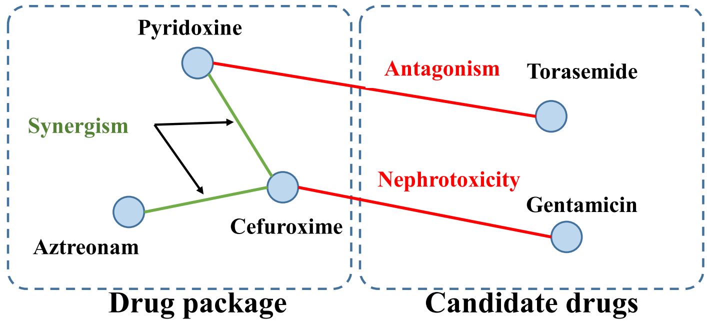


Fig. 1. An example for a patient with kidney disease. Blue nodes indicate drugs. Green edges connecting two blue nodes indicate synergism, while red edges connecting two blue nodes indicate antagonism or toxicity.

construct a drug interaction graph that contains all the drugs in the EMR dataset. After that, we formulate the drug interaction graph as an attributed graph and utilize the edge attribute vectors to describe the influence of drug interaction based on MPNN. Then, we propose a novel drug package generator based on **Recurrent Neural Network (RNN)**. In each step of the RNN generation, we exploit a mask layer to explicitly capture the patient condition's impact on the drug interaction. Furthermore, we utilize reinforcement learning to reduce the dependence on the drug order, and we propose a joint learning method to train the MPNN and RNN models simultaneously. Finally, extensive experiments on two real-world datasets demonstrate the effectiveness of our DPG framework compared with several competitive baseline methods. To the best of our knowledge, the contribution of this article can be summarized as follows:

- We develop a new end-to-end framework named DPG to generate drug packages based on a recurrent neural network, which can capture the effect of different types of drug interaction and the influence of the patient condition explicitly.
- We propose to construct a drug interaction graph based on the drug interaction data, and we further design a message-passing neural network to get the drug embedding and capture the interaction between drugs.
- We propose a hybrid loss function to learn the parameters of our DPG model. Furthermore, we propose training strategies to reduce the dependence on the drug order based on both maximum likelihood estimation and policy gradient and testing strategies to generate a candidate package set and provide the best drug package for a specific patient.
- We conduct extensive experiments on two real-world datasets, which validate the effectiveness of our DPG framework.

Note that to solve the drug recommendation problem, we have done some preliminary work in Reference [77] and proposed a framework named DPR. However, there are many differences between DPR and DPG. The essential difference is that DPR is a discriminative model that can only select suitable packages that already exist within the data, while DPG is a generative model based on RNN, which can generate new drug packages for patients after the training process. Therefore, DPR is trained by BPR loss function, while DPG is trained by reinforcement learning. Furthermore, DPR utilizes the MPNN model only for the drugs in a specific drug package, while DPG utilizes the MPNN model on the graph that consists of all the drugs in the dataset to get the drug embedding and capture the interaction.

Overview. The rest of this article is organized as follows: In Section 2, we briefly introduce some related works of our study. In Section 3, we introduce the preliminaries and formally define the problem of drug package recommendation. Technical details of our Drug Package Generation framework will be introduced in Section 4. Then, we comprehensively evaluate the model performance in Section 5, with further discussions on the interpretability of results. In Section 6, we conclude the article.

2 RELATED WORK

In this section, we will briefly provide a comprehensive review of the relevant approaches. Specifically, we group the related works into four lines of literature: *Drug Recommendation System*, *Package Recommendation System*, *Graph Neural Networks*, and *Discrete Data Generation*.

2.1 Drug Recommendation System

Recommendation systems have been widely used in a variety of applications like social networking and e-commerce. The methods can be broadly classified into two categories, respectively, neighborhood-based collaborative filtering methods based on similar users or items [1], and model-based methods, particularly latent factor models that factorize the user-item matrix into user factors and item factors [27]. Current recommender systems have been further advanced by the significant contribution from deep learning [12, 21, 60, 64, 72], where user preferences and item characteristics can be learned in deep architectures. For example, by replacing the inner product in the matrix factorization methods with a neural architecture that can learn an arbitrary function from data, He et al. [21] present **Neural network-based Collaborative Filtering (NCF)**.

Furthermore, to provide more accurate, diverse, and explainable recommendations, many efforts have been made beyond modeling user-item interactions and taking side information into account. For example, Wang et al. [56] investigate the utility of **knowledge graph (KG)** and proposes a new method named **Knowledge Graph Attention Network (KGAT)**, which explicitly models the high-order connectivities in KG in an end-to-end fashion. Based on these technologies, some methods focusing on drug recommendation have been put forward. For example, Zheng et al. [78] introduce an LDA-based contextual collaborative model called Medicine-LDA to integrate the multi-source information. Zhang et al. [71] construct a heterogeneous graph that includes patients and drugs, and describes a novel recommendation system based on label propagation. In recent years, some researchers have further incorporated external knowledge into the design of their models. For example, Zhang et al. [74] utilize a recurrent decoder to model label dependencies and incorporates external knowledge into the design of the reinforcement reward. Shang et al. [46] propose to integrate the drug-drug interactions knowledge graph by a memory module implemented as a graph convolutional networks, and models longitudinal patient records as the query. Shang et al. [45] propose to utilize structural knowledge like clinical ontology to learn better representation called tree embedding by utilizing the ancestors' information. Wang et al. [54] propose to jointly embed diseases, drugs and patients into a shared lower-dimensional space, and decomposes the drug recommendation into a link prediction process. However, the studies on drug interaction are not thorough enough. Different from the prior arts, our method can capture the drug interaction explicitly and utilize the external knowledge personally.

2.2 Package Recommendation System

Most recommendation research concentrates on recommending one item to one user at a time. However, in many real-world scenarios, the platform needs to show users a set of items, in other words, a package (or a bundle). For example, modern e-commerce websites and online service businesses, e.g., Amazon, Taobao, Steam, and Netflix, develop new applications [3, 42, 49, 63], which recommend and sell a list of packages rather than a list of items. Bai et al. [3] propose that a package recommendation system is beneficial to both customers and sellers. For customers, high-quality packages broaden their interests and indicate the complementary products directly. For sellers, they are bundling increases per customer transaction. Therefore, several efforts have been made to solve this problem. Some studies turn this problem into optimization problems like 0-1 Knapsack problem, and provide some approximate solutions due to the NP-Hardness [15, 28, 41, 79]. Liu et al. [36] put forward a Tourist-Area-Season topic model and proposes a cocktail approach on personalized travel package recommendation. Bai et al. [3] propose a bundle generation network that decomposes the problem by determinantal point processes. Pathak et al. [42] develop a model that utilizes the trained features of an item recommendation model to learn the personalized ranking over bundles. Chen et al. [10] contribute a neural network solution based on factorized attention network to aggregate the item embeddings in a package. Chang et al. [9] propose a model based on graph neural network, which explicitly models the interaction and affiliation between users, bundles, and items by unifying them into a heterogeneous graph. However, these models neglect the different types of interactions between items, and most of these models are discriminative models, which prevents them from capturing satisfactory performance for drug package recommendation.

2.3 Graph Neural Networks

As shown in Reference [62], deep learning has revolutionized many machine learning tasks in recent years, ranging from image classification and video processing to speech recognition and natural language understanding. The data in these tasks are typically represented in the Euclidean space. However, there is an increasing number of applications where data are generated from non-Euclidean domains and represented as graphs with complex relationships and interdependencies between objects. Recently, many studies on extending deep learning approaches for graph data have emerged [11, 30–32, 61, 67, 70, 73]. Unlike standard neural networks, GNNs retain a state representing information from its neighborhood with arbitrary depth. For example, Kipf et al. [26] present **graph convolutional network (GCN)** for semi-supervised learning on graph data via an approximation of spectral graph convolutions. Li et al. [34] propose **Gated Graph Neural Networks (GG-NNs)**, which is an adaptation of GNNs that is suitable for both non-sequential and sequential outputs. Hamilton et al. [19] present GraphSAGE to generate node embeddings by sampling and aggregating features from the local neighborhoods of nodes. Velivckovic et al. [51] present **graph attention networks (GATs)** that leverage masked self-attentional layers to address the shortcomings of methods based on graph convolutions. Gilmer et al. [17] further present that the essence of existing GNNs is to learn a message-passing algorithm and an aggregation procedure to compute a function of the entire input graph, and reformulate existing models into a single common framework called MPNNs. With the strong power of the learning structure, GNNs have been widely applied in many fields. For example, Zhang et al. and Li et al. [33, 69] utilize graph data and graph neural networks for competitive analysis. Liu et al. [38] propose a deep model to integrate structural and temporal social contexts to address the dynamic social-aware recommendation task. Wang et al. [57] exploit the user-item graph structure by propagating embeddings, which leads to the expressive modeling of high-order connectivity.

2.4 Discrete Data Generation

Deep generative models have recently drawn significant attention, and the ability to learn over large-scale data endows them with more potential and vitality [5]. Salakhutdinov et al. [44] contribute an efficient learning procedure for fully general Boltzmann machines. Bengio et al. [6] develop a denoising autoencoder that learns the data distribution in a supervised learning fashion. Recently, the most popular generative models are **Generative Adversarial Nets (GANs)** [18, 73] and **Variational AutoEncoder (VAE)** [25], where Goodfellow et al. [18] propose to train a generative model and a discriminative model by a min-max game and Kingma et al. [25] combine deep learning with stochastic variational inference. Both GANs and VAE have gained striking successes in continuous data generation, e.g., natural image generation[50, 58]. However, most of these generative models are designed to adjust the output continuously, which does not work on discrete data generation, e.g., text and set generation. To solve this problem, RNN-based models like LSTM [22] and GRU [13] are most commonly used, where the models generate one word or item at one time step [47, 66]. Yu et al. [68] propose that the discrete outputs from the generative model in GANs make it challenging to pass the gradient update from the discriminative model to the generative model. Furthermore, Yu et al. [68] solve this problem by developing SeqGAN to model the data generator as a stochastic policy in **reinforcement learning (RL)** and using Monte Carlo search to calculate the reward for the intermediate state-action steps. Inspired by this, Liu et al. [37] improve the previous image captioning methods by using Monte Carlo rollouts instead of mixing MLE training with **policy gradient (PG)**. Dai et al. [14] further contribute an image description method based on conditional GAN, which encourages not only fidelity but also naturalness and diversity. Yang et al. [65] propose that the RNN models are not suitable for the set generation task, since RNN models are trained with the **maximum likelihood estimation (MLE)** method and the cross-entropy loss function, which relies on strict order. Therefore, Yang et al. [65] develop a novel set generation model based on RL, which not only captures the correlations between items but also reduces the dependence on the item order. Zhao et al. [75, 76] propose to utilize reinforcement learning for video question answering.

In this article, we follow some outstanding ideas in the above works according to the properties of the drug package recommendation task. Along this line, we develop a new end-to-end framework named DPG to generate drug packages based on RNN and RL. Along this line, we further utilize MPNN to integrate drug interaction information and a mask layer to capture the impact of the patient's condition on the drug package generation process. Therefore, DPG can not only capture the drug interaction information in the package generation process but also overcome the shortness of discriminative models.

3 PRELIMINARIES

In this section, we first introduce the two datasets used in our study, respectively, our private dataset named APH, which comes from a first-rate hospital in China, and a public dataset named MIMIC-III [23]. Then, we propose the problem formulation of drug package recommendation.

3.1 Data Description and Preprocessing

3.1.1 The APH Dataset. Our private dataset named APH used in this article comes from the electronic medical record database of a first-rate hospital in China. As shown in Figure 2, each medical record contains the following information:

- **Demographics.** Demographics are formatted data including basic patient information, such as patient's gender, age, type of medical insurance, whether surgery has been performed, and so on. This information provides guidance for doctors to prescribe, for example, some drugs are not suitable for children, while some drugs are only covered by certain medical insurance, and so on.
- **Laboratory results.** A laboratory test is a procedure in which the hospital takes a sample of the patient's body fluid or tissue to get information on the patient's health. The laboratory results are shown as the patient's values and normal values for laboratory items. For example, "glucose value: 77 mg/dL, normal value: 65-99 mg/dL."

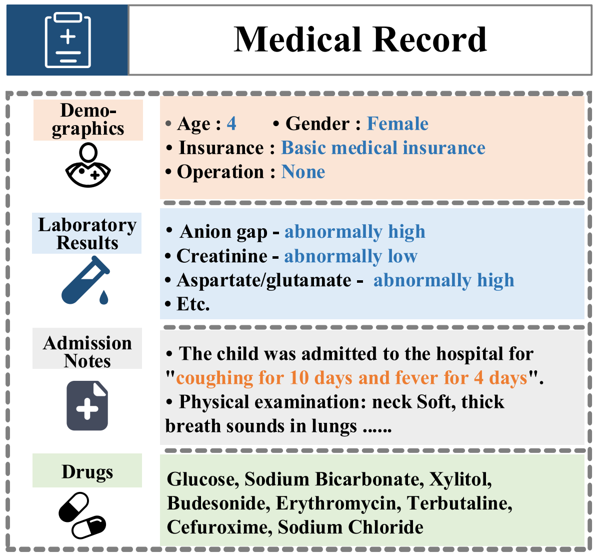


Fig. 2. An example of the medical record in our private dataset. Each medical record contains demographics, laboratory results, admission notes, and drugs corresponding to a specific patient.

- **Admission notes.** An admission note is part of a medical record that documents the patient's status, including physical examination findings, reasons why the patient is being admitted for inpatient care to a hospital, and the initial instructions for the patient's care.
- **Drugs.** This information includes all of the drugs used during the patient's hospital stay.

To integrate and utilize the above multi-source heterogeneous data, we conduct the following preprocessing steps. First, for the demographics, since each type of information may correspond to too many values, e.g., the age information may correspond to more than one hundred values, so we propose some classification methods as shown in Table 1. Based on these classification methods, we convert demographics into documents, e.g., "Gender: Male, Age: Agedness." Second, for the laboratory results, we divide the results into three levels, respectively, normal, abnormally high and abnormally low, according to the given typical values. We then extract all abnormal test results (abnormally high and low) and convert them into documents, e.g., "glucose value: abnormally high, lipid panel: abnormally high." After that, we merge the demographic documents and laboratory result documents, namely, disease documents. Finally, we remove all the punctuation and meaningless characters for the admission notes and adjust all of the admission notes in the dataset to the same length by padding and cut-off.

To study the interaction between drugs, we collect data from two large online pharmaceutical knowledge bases, i.e., DrugBank[1] and YaoZhi,[2] where users can check drug properties and drug-drug interaction. The drug interaction information in these two databases is stored in text format based on certain templates. We further classify the templates into three categories with the help of domain experts, respectively, *No Interaction*, *Synergism*, and *Antagonism*. Specifically, *No Interaction* means there is no interaction between two drugs, *Synergism* means combining two drugs can lead to enhanced drug effect, and *Antagonism* is the opposite. Table 2 shows some examples of

[1]https://go.drugbank.com/releases/latest.
[2]https://db.yaozh.com/interaction.

Table 1. Different Classification Methods for Different Types of Demographics Information

Information type	Classification method
Gender	Male, Female
Age	0~3 years old as infant, 4~12 as early youth, 13~45 as youth, 46~59 as middle age, 60+ as agedness
Insurance	High quality, Low quality, Self-paying, Unknown
Operation	Yes, No
Anesthesia	General, Local, No

Table 2. Examples of Drug Interaction Labeling

Drug A	Drug B	Description	Classification	Direction
Amoxicillin	Oseltamivir	No Interaction	No Interaction	Bidirection
Dipyridamole	Valsartan	Dipyridamole may increase the antihypertensive activities of Valsartan.	Synergism	A to B
Repaglinide	Doxepin	Doxepin may decreas the hypoglycemic activities of Repaglinide.	Antagonism	B to A

There are three types of interaction and the interaction can be directed.

different drug interactions. Note that the interaction can be directed, for example, if drug A can increase the effectiveness of drug B, then the direction is from A to B. Moreover, for most drug pairs, we cannot confirm whether there are any interactions between them, so we leave them unlabeled.

3.1.2 The MIMIC-III Dataset. The MIMIC-III dataset is a publicly available dataset consisting of medical records of 40K **intensive care unit (ICU)** patients over 11 years. Following References [46, 55], we use the diagnose codes and procedure codes to reflect the condition of patients, and the dataset was preprocessed similarly to Reference [55]. Note that there are no admission notes in the MIMIC-III dataset, so we only converted the diagnosis codes and procedure codes into disease documents. We also utilize the interaction data proposed in Section 3.1.1 for capturing the drug-drug interaction in MIMIC-III.

3.2 Problem Formulation

Based on the above EMR and drug interaction data, here we introduce the problem formulation of drug package recommendation. For facilitating illustration, Table 3 lists some important mathematical notations used throughout this article.

Suppose there are N patients and M drugs in the training set. Based on the above preprocessing method, for patient i, we can construct the disease document and turn it into one-hot encoding form as $\mathcal{W}_i = \{w_{i,1}, w_{i,2}, \ldots, w_{i,p}\}$, where $w_{i,\cdot}$ is the 0/1 indicator value for a demographic feature or a lab result. In addition, we can formulate the admission note as $\mathcal{T}_i = \{t_{i,1}, t_{i,2}, \ldots, t_{i,q}\}$, where $t_{i,\cdot}$ is a word in the processed admission note. In this way, the patient i can be expressed as a patient description $\mathcal{U}_i = \{\mathcal{W}_i, \mathcal{T}_i\}$. Note that for the MIMIC-III dataset, there are no admission notes and the patient description $\mathcal{U}_i$ is just equal to $\mathcal{W}_i$. We also have the drug package $\mathcal{P}_i = \{d_{i,1}, d_{i_2}, \ldots, d_{i,s}\}$, where $d_{i,\cdot}$ is a drug that patient i used. Moreover, based on the labeled drug interaction data, we can construct the drug relation matrix $\mathcal{R} \in \mathbb{R}^{M \times M}$, where $\mathcal{R}_{ij}$ represents the

Table 3. Mathematical Notations

Symbol	Description
N, M	Number of patients and the number of drugs
K	Average size of drug packages
L	Layer number of MPNN
$\mathcal{S}$	Set of all the drugs appeared in the dataset
$\mathcal{G}$	Drug interaction graph of set $\mathcal{S}$
$\mathcal{R}$	Drug relation matrix
C	Candidate drug package set
$\mathcal{P}_i$	Drug package of patient i
$\mathcal{W}_i$	Disease document of patient i
$\mathcal{T}_i$	Admission note of patient i
$\mathcal{U}_i$	Patient description of patient i
Θ	Model Parameters
n	Size of the candidate drug package set
d_i	ith drug in the entire drug set
$d_{i,\cdot}$	Drug in the drug package of patient i
$w_{i,\cdot}$	Indicator value in the disease document of patient i
$t_{i,\cdot}$	Word in the admission note of patient i
$\mathbf{d}$	Drug embedding for the corresponding drug
$\mathbf{r}_{ij}$	Representation for the interaction between d_i and d_j
$[\cdot\|\|\cdot]$	Concatenation of two vectors
$MLP(\cdot)$	Multilayer Perceptron with ReLU Activation Function

interaction between d_i and d_j. $\mathcal{R}_{ij}$ is initialized as follows:

$$\mathcal{R}_{ij} = \begin{cases} 0 & no\ interaction\ between\ d_i\ and\ d_j, \\ 1 & the\ interaction\ between\ d_i\ and\ d_j\ is\ Synergism, \\ 2 & the\ interaction\ between\ d_i\ and\ d_j\ is\ Antagonism, \\ -1 & the\ interaction\ between\ d_i\ and\ d_j\ is\ Unknown. \end{cases} \tag{1}$$

Note that the direction is from d_i to d_j. Along this line, the problem of drug package recommendation can be formulated as:

Definition 1 (Drug Package Recommendation). Given the patient descriptions $\{\mathcal{U}_1, \ldots, \mathcal{U}_N\}$ with the corresponding drug packages $\{\mathcal{P}_1, \ldots, \mathcal{P}_N\}$, and the drug relation matrix $\mathcal{R}$, the goal of drug package recommendation is to get a personalized **generator** g, which can generate a candidate drug package set $C = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_n\}$ and pick out the most suitable drug package $\mathcal{P} \in C$ based on each patient description $\mathcal{U}$.

In this way, we formulate the drug package recommendation system as a generative model, which makes the problem essentially different from our preliminary work [77]. Moreover, instead of just generating one package for one patient, we force g to generate a candidate drug package set C at first. The rationale behind this is that in the real-world clinical treatment process, the doctors may not only hope the drug recommendation system can indicate the drugs that are most likely to be used but also provide a variety of possible treatment plans, especially those that include some unpopular drugs. To provide the doctor with the most excellent help, the candidate drug packages should have the following merits: (1) **Accuracy**, which means the most suitable drug package

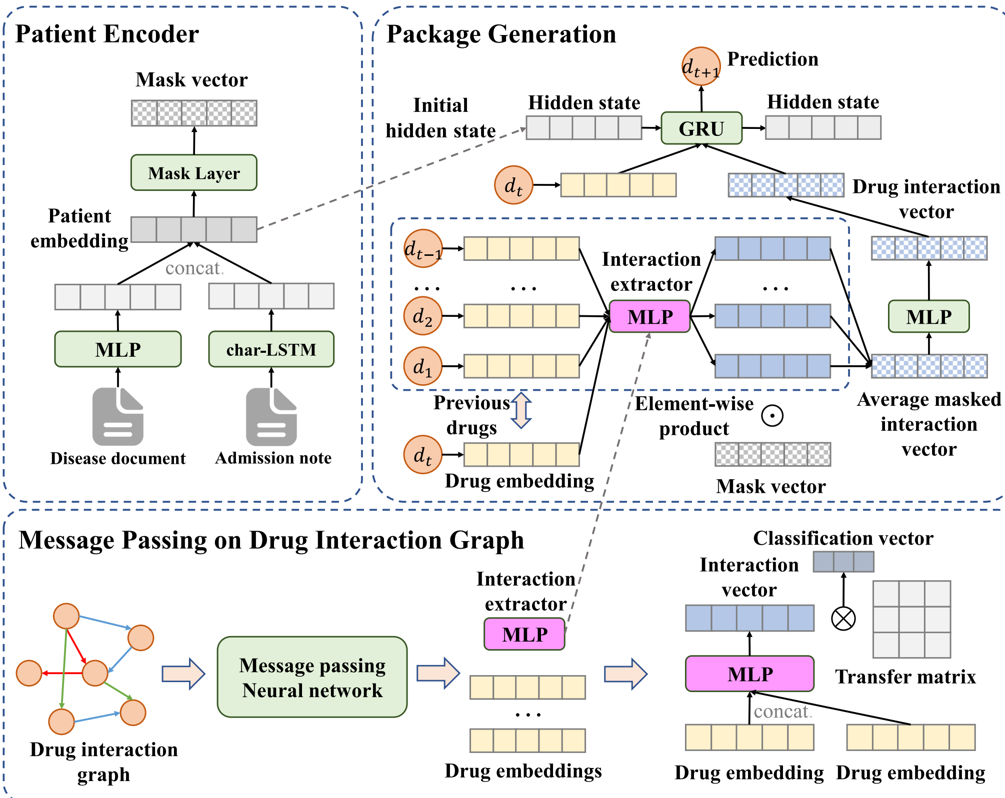


Fig. 3. A framework overview of the drug package generation model. Orange nodes indicate drugs. Green boxes indicate neural network models. Pink box indicates the interaction extractor. Yellow rectangles indicate drug embedding. Dark grey rectangles indicate patient embedding and the dark grey rectangles with dots indicate the corresponding mask vectors.

$\mathcal{P} \in C$ should be as accurate as possible. (2) **Comprehensiveness**, which means all the drugs contain in C should cover the drugs that are actually used as much as possible. (3) **Diversity**, which means the packages in C should be as diverse as possible to provide more possible options for the doctors to consider. We will discuss more about how to generate the candidate set C and how to evaluate the generation result in the following sections.

4 TECHNICAL DETAILS

In this section, we will introduce the framework of our model in detail. As shown in Figure 3, our framework mainly consists of three components, i.e., message passing on drug interaction graph, patient encoder, and drug package generation. Specifically, we first construct the drug interaction graph based on the drug relation matrix $\mathcal{R}$ and design a message-passing neural network to get the drug embedding and capture the interaction between drugs. Then, we get the embedding of the patient descriptions. Finally, we propose a novel drug package generation model, which explicitly captures the effect of drug interaction and the influence of patient condition. Besides, we also propose the training and testing methods of our DPG model based on maximum likelihood estimation and policy gradient.

4.1 Message Passing on Drug Interaction Graph

Compared with traditional item recommendation, the core problem of drug package recommendation is how to capture the interaction between drugs. Therefore, in this section, we propose to utilize graph models to solve this problem. To be specific, we first present a method to construct a drug interaction graph. Then, we formulate the message-passing framework, which will be further utilized for the drug package generation task.

4.1.1 Drug Interaction Graph Construction. For the set $\mathcal{S}$ of all the drugs in our dataset, we define a corresponding drug interaction graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the node set and $\mathcal{E}$ is the edge set. Each specific node $v \in \mathcal{V}$ is associated with corresponding drug d and corresponding node embedding $\mathbf{v}$, which is randomly initialized. Furthermore, to fuse the information of the drug relation matrix $\mathcal{R}$, we propose the following criterion to define the topology structure of $\mathcal{G}$. For nodes v, u, if $\mathcal{R}_{vu} \neq -1$, which means this drug paired has been labeled, then edge e_{vu} exists. Otherwise, the edge e_{vu} does not exist.

4.1.2 Message Passing on Drug Interaction Graph. We propose to exploit the MPNN [17] framework for making use of the drug interaction graph constructed in the last section. MPNN is a general approach to describe GNNs, which inductively learns a node representation by recursively aggregating and transforming the feature vectors of its neighboring nodes. A per-layer update of the MPNN model in our setting involves message-passing, message aggregation, and node representation updating, which can be expressed as

$$\mathbf{m}_{vu}^{(l)} = \text{MESSAGE}\left(\mathbf{h}_u^{(l-1)}, \mathbf{h}_v^{(l-1)}, \mathbf{e}_{vu}\right), \tag{2}$$

$$\mathbf{M}_u^{(l)} = \text{AGGREGATION}\left(\left\{\mathbf{m}_{vu}^{(l)}, \mathbf{e}_{vu}\right\} \mid v \in \mathcal{N}(u)\}\right), \tag{3}$$

$$\mathbf{h}_u^{(l)} = \text{UPDATE}\left(M_u^{(l)}, \mathbf{h}_u^{(l-1)}\right), \tag{4}$$

where $\mathbf{m}_{vu}^{(l)}$ is the message vector passing from v to u, $\mathbf{h}_u^{(l)}$ is the representation of node u on the layer l; $\mathbf{e}_{vu}$ is the attribute corresponding to edge e_{vu}. Note that $\mathbf{e}_{vu}$ is a vector. $\mathcal{N}(u)$ is the neighborhood of node u from where it collects information to update its aggregated message $\mathbf{M}_u$. Based on the MPNN framework, we propose our message-passing process on the drug interaction graph as follows:

$$\mathbf{e}_{vu}^{(l)} = MLP^{(l)}\left(\left[\mathbf{h}_u^{(l)} || \mathbf{h}_v^{(l)}\right]\right), \tag{5}$$

$$\mathbf{m}_{vu}^{(l)} = W_1^{(l-1)} \mathbf{e}_{vu}^{(l-1)}, \tag{6}$$

$$\mathbf{M}_u^{(l)} = \sum_{v \in \mathcal{N}(u)} \mathbf{m}_{vu}^{(l)}, \tag{7}$$

$$\mathbf{h}_u^{(l)} = MLP\left(W_0^{(l-1)} \mathbf{h}_u^{(l-1)} + \mathbf{M}_u^{(l)}\right), \tag{8}$$

where $\mathbf{h}_u^{(0)}$ is initialized by corresponding node embedding $\mathbf{v}_u$ and W denotes the model's parameters to be learned. $MLP^{(l)}$ is the multilayer perceptron utilized in lth layer to calculate the edge attributes based on node representation. After L layer of the message-passing process, we can get the node representation $\mathbf{h}_u^{(L)}$, edge attribute $\mathbf{e}_{vu}^{(L)}$ and multilayer perceptron $MLP^{(L)}$. $\mathbf{h}_u^{(L)}$ is the drug embedding for the corresponding drug d_u, i.e., $\mathbf{d}_u = \mathbf{h}_u^{(L)}$. $\mathbf{e}_{vu}^{(L)}$ is the representation for the interaction between the corresponding drugs d_u and d_v based on the interaction feature extractor $MLP^{(L)}$. We also express $\mathbf{e}_{vu}^{(L)}$ as $\hat{\mathbf{e}}_{vu}$ and $MLP^{(L)}$ as MLP_{inter} for facilitating illustration. Note that the drug embeddings and the MLP_{inter} model will be further utilized in the following sections, and all the models and parameters mentioned in this section can be trained simultaneously with the drug package generator.

4.1.3 Learning with Edge Classification. In the above message-passing process, we get the edge representation $\hat{\mathbf{e}}_{vu}$ based on the drug embeddings and the interaction feature extractor MLP_{inter}. We further propose that the edge representation $\hat{\mathbf{e}}_{vu}$ should contain the information about the interaction type. Along this line, we define the transfer matrix $\mathbf{Q} \in \mathbb{R}^{D\times 3}$ to transform the edge representation $\hat{\mathbf{e}}_{vu}$ into classification probabilities, where D is the dimension of $\hat{\mathbf{e}}_{vu}$. Specifically, we can calculate the three-dimensional classification probability vector $\hat{\mathbf{e}}_{vu}^{\top}\mathbf{Q}$, where each dimension in this vector reflects the probability for the corresponding interaction type following equation 1. Note that the Unknown type does not exist in the drug interaction graph. Finally, we can form the cross-entropy loss function for the MPNN on the drug interaction graph, which aims to force the edge attribute $\hat{\mathbf{e}}_{vu}$ to contain the interaction type information as follows:

$$L_{graph} = -\sum_{u,v\in\mathcal{G}} \ln\left(softmax\left(\hat{\mathbf{e}}_{vu}^{\top}\mathbf{Q}\right)_{\mathcal{R}_{uv}}\right). \tag{9}$$

4.2 Patient Encoder

The next step of drug package generation is getting the embedding of the patients. As shown in Section 3.2, a patient's description consists of two heterogeneous parts. Therefore, we propose a hybrid method to get the patient embedding $\mathbf{u}$ based on patient description $\mathcal{U} = \{\mathcal{W}, \mathcal{T}\}$, which can be split into two steps. To be specific, in the first step, we extract the feature of the patient's disease document by MLP as

$$\mathbf{m}_w = MLP\left(\mathcal{W}\right). \tag{10}$$

In the second step, we associate each word t_k in patients' admission notes with a word embedding vector $\mathbf{x}_k$. By this way, we can convert $\mathcal{T}$ to a sequence of vectors $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_q)$. Then, we input the sequence into char-GRU[3] and get the final time step output $\mathbf{h}_q$ as the embedding of $\mathcal{T}$. The patient embedding $\mathbf{u}$ is the concatenation of the two parts:

$$\mathbf{u} = [\mathbf{m}_w || \mathbf{h}_q]. \tag{11}$$

Furthermore, we calculate the personalized mask vector based on the patient embedding as follows:

$$\mathbf{m} = \sigma\left(MLP\left(\mathbf{u}\right)\right), \tag{12}$$

where $\sigma(\cdot)$ is the sigmoid function. The mask layer projects the real numbers in $\mathbf{u}$ to the range 0 to 1, which can be regarded as a personalized feature selection process.

4.3 Drug Package Generation

After getting the drug and edge embedding based on the drug interaction graph and the embedding of the patient description, we can further design our model for drug package generation. We formulate the drug package generation as a sequence generation process, and utilize an RNN-based model to solve this problem. In each step of the RNN generation, we exploit a mask layer to capture the impact of the patient's condition on the drug interaction. To train the sequence generation model, we first propose to sort the drugs by frequency in descending order, and take the result of **Maximum Likelihood Estimation (MLE)** as the loss function. However, as shown in Reference [52], the order of the items in a sequence has a significant impact on the performance of the sequence generation model. Obviously, it is more appropriate to treat drug packages as

[3]Different from our preliminary work, we utilize GRU instead of LSTM, since the performance of GRU is a little bit better.

unordered sets rather than ordered sequences. Therefore, following Reference [65], we further utilize deep reinforcement learning to reduce the dependence of the model on the sequence order.

4.3.1 Drug Package Generation with Maximum Likelihood Estimation. For a patient description $\mathcal{U}$, we can get the patient embedding $\mathbf{u}$ as shown in Section 4.2. Then, in this section, we formulate the drug package generation problem as finding an optimal drug sequence that maximizes the conditional probability, which is calculated as follows:

$$p\left(d_1, \ldots, d_T \mid \mathbf{u}\right) = \prod_{t=1}^{T} p\left(d_t \mid d_1, \ldots, d_{t-1}, \mathbf{u}\right). \tag{13}$$

To convert the drug packages to drug sequences, we first sort the drugs in each drug package according to the frequency of the drugs in the training set. High-frequency drugs are placed in the front. In addition, the *bos* and *eos* symbols are added to the head and tail of the drug sequences, respectively. Then, if we choose GRU as the sequence generation model, then we could input $\mathbf{u}$ as the initial hidden state. We can calculate the conditional probability $p\left(d_{t+1} \mid d_1, \ldots, d_t, \mathbf{u}\right)$ by simply inputting the corresponding drug embedding to the GRU unit at each time step, and get the output probability distribution over the drug space at time step t as

$$d_{t+1} \sim softmax\left(\mathbf{W}_o \mathbf{h}_t\right), \tag{14}$$

where $\mathbf{h}_t$ is the output of the GRU cell at time step t. However, in this simple method, the effect of the drug interaction and the influence of the patient condition are all captured *implicitly* by the hidden state $\mathbf{h}$ of the GRU model, which severely reduces the expressive ability and the interpretability of the model. Therefore, in our DPG framework, we propose to utilize the interaction feature extractor in Section 4.1.2 and the mask layer to capture the effect of the drug interaction and the influence of the patient condition *explicitly*.

Supposing that our DPG model has generated a drug sequence $\mathbf{y} = \{d_1, d_2, \ldots, d_t\}$ at time step t. If we want to capture the drug interaction effect during the generation process, then a straightforward method is to convert the drug sequence $\mathbf{y}$ to a complete graph, where all drugs are connected with each other, and calculate the drug interaction vectors as the edge attributes of the graph. However, this will make the calculation time complexity of time step t increases from $O(1)$ to $O(t^2)$, which is unacceptable for sequence generation task. Therefore, we propose to capture the drug interaction that is only related to d_t at time step t. In this way, the time complexity of time step t decreases from $O(t^2)$ to $O(t)$. Furthermore, with the update of the hidden state and the gate vectors in the GRU model, the drug interaction information before time step t can be saved effectively. Specifically, at time step t, the drug interaction vector $\mathbf{i}_t$ can be calculated as follows:

$$\mathbf{i}_t = MLP\left(\sum_{k=1}^{t-1} \mathbf{m} \odot MLP_{inter}\left([\mathbf{d}_k || \mathbf{d}_t]\right)\right), \tag{15}$$

where $\mathbf{m}$ is the mask vector calculated by equation 12. $\odot$ represents the element-wise product of two vectors. MLP_{inter} is the interaction feature extractor in Section 4.1.2. $MLP_{inter}\left([\mathbf{d}_k || \mathbf{d}_t]\right)$ captures the interaction information between d_k and d_t, which is further updated by the mask vector $\mathbf{m}$. In this way, the model can capture the personalized influence of the patient condition on the drug interaction effect. Based on the drug interaction vector, the update process of the GRU unit in our DPG framework can be formulated as follows:

$$\begin{aligned}
\mathbf{r}_t &= \sigma\left(\mathbf{W}_{dr}\left[\mathbf{d}_t || \mathbf{i}_t\right] + \mathbf{W}_{hr}\mathbf{h}_{t-1} + \mathbf{b}_r\right), \\
\mathbf{z}_t &= \sigma\left(\mathbf{W}_{dz}\left[\mathbf{d}_t || \mathbf{i}_t\right] + \mathbf{W}_{hz}\mathbf{h}_{t-1} + \mathbf{b}_z\right), \\
\tilde{\mathbf{h}}_t &= \tanh\left(\mathbf{W}_{dh}\left[\mathbf{d}_t || \mathbf{i}_t\right] + \mathbf{W}_{hh}\left(\mathbf{r}_t \odot \mathbf{h}_{t-1}\right) + \mathbf{b}_h\right), \\
\mathbf{h}_t &= \left(1 - \mathbf{z}_t\right) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t.
\end{aligned} \tag{16}$$

Furthermore, another problem of the RNN-based model is that the model will generate repeated items during the generation process. However, there is no repeated drugs in a drug package. Therefore, in our DPG framework, we propose to get the output probability distribution over the drug space at time step t as

$$d_{t+1} \sim softmax\left(\mathbf{W}_o \mathbf{h}_t + \mathbf{M}_t\right), \tag{17}$$

where $M_t \in \mathbb{R}^M$ is the mask vector, which is used to prevent the GRU model from generating repeated drugs as

$$(M_t)_i = \begin{cases} -\infty & \text{if the } i\text{th drug has been predicted,} \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

Finally, the loss function for drug package generation with MLE loss can be formulated as follows, where d_t^* is the ground-truth drug at time step t:

$$L_{MLE} = -\sum_{t=1}^{T} \log(softmax\left(\mathbf{W}_o \mathbf{h}_{t-1} + \mathbf{M}_{t-1}\right)_{d_t^*}). \tag{19}$$

4.3.2 Drug Package Generation with Policy Gradient. Although maximum likelihood estimation is extensively used in sequence generation tasks, Reference [52] proves that the order has a great impact on the performance of the sequence generation model. Training based on MLE is reasonable only when there exists a strict order in the output items and this order is known in practice, e.g., text generation [47], music generation [8]. However, drug packages are naturally unordered, which are more appropriate to be treated as unordered sets rather than ordered sequences. To solve this problem, we propose to utilize RL technologies to reduce the dependence of sequence generation on the item order.

RL technologies have been widely used to improve the performance of RNN-based models for sequence generation tasks, e.g., text generation [68], image caption [37]. In these works, an RNN-based sequence generation model is formulated as an agent in the RL framework. At time step t, the state s is the currently produced tokens, and the action a is the next token to select based on the stochastic policy defined by the parameters θ of the generation model. The methods to get the reward r can be divided into two categories. The first method is getting the reward from specific evaluation metrics, e.g., BLEU for text generation, while the second one is getting the reward from another discriminator model following the GANs [18] framework. In this work, we propose to utilize the first method to get the reward, since we find that GANs are not suitable for the drug package generation task, and we will give a more detailed discussion about this later. Based on the above definitions, given a patient embedding $\mathbf{u}$, the goal of the RL model is to minimize the negative expected reward, which can be estimated with a single sample as

$$L(\theta) = -\mathbb{E}_{\mathbf{y} \sim p_\theta(\mathbf{u})}[r(\mathbf{y})] \approx -r(\tilde{\mathbf{y}}), \tag{20}$$

where $\tilde{\mathbf{y}} = \{d_1, d_2, \ldots, d_T\}$ and d_t is the drug sampled from the model at the time step t.

A core problem is how to calculate the reward $r(\mathbf{y})$. The design of the reward function depends on the characteristic of the task. For example, for the unsupervised text generation task, the generator needs to output sequences *similar* to some real-world sequences without ground-truth. Therefore, Yu et al. [68] utilize a discriminator model to calculate the reward following the GAN framework. However, for the drug package generation task, each patient has a corresponding ground-truth drug package given by human experts. To capture the prior human knowledge, the reward function should encourage the generation model to output drug packages that are *exactly* *the same* with the ground-truth packages. Therefore, we design the reward as the F1 score as

$$Precision\,(\mathbf{y},\mathcal{P}) = \frac{|\mathbf{y}\cap\mathcal{P}|}{|\mathbf{y}|}, \tag{21}$$

$$Recall\,(\mathbf{y},\mathcal{P}) = \frac{|\mathbf{y}\cap\mathcal{P}|}{|\mathcal{P}|}, \tag{22}$$

$$r(\mathbf{y}) = F_1\,(\mathbf{y},\mathcal{P}) = \frac{2 * Precision\,(\mathbf{y},\mathcal{P}) * Recall\,(\mathbf{y},\mathcal{P})}{Precision\,(\mathbf{y},\mathcal{P}) + Recall\,(\mathbf{y},\mathcal{P})}, \tag{23}$$

where $\mathbf{y}$ is the drug sequence generated by the model and $\mathcal{P}$ is the corresponding ground-truth drug package. Note that the *bos* and *eos* symbols in $\mathbf{y}$ are deleted before calculation. Since the calculation of the F1 value is independent of the order, the generation model is free from the strict restriction of item order in sequences.

To compute the gradient $\nabla_\theta L(\theta)$, we utilize the **Policy Gradient (PG)** [48, 59] method. Policy gradient performs gradient descent by calculating the expected gradient of a non-differentiable reward function as follows:

$$\begin{aligned} \nabla_\theta L(\theta) &= -\mathbb{E}_{\mathbf{y}\sim p_\theta(\mathbf{u})}\left[r\,(\mathbf{y})\,\nabla_\theta \log p_\theta\,(\mathbf{y}\mid\mathbf{u})\right] \\ &\approx -r\,(\tilde{\mathbf{y}})\,\nabla_\theta \log p_\theta\,(\tilde{\mathbf{y}}\mid\mathbf{u})\,. \end{aligned} \tag{24}$$

Again, the gradient can be approximated using a single sample in practice. Furthermore, to reduce the variance of the gradient estimate, we can add a *baseline* b to Equation (24) as

$$\nabla_\theta L(\theta) \approx -\left(r\,(\tilde{\mathbf{y}}) - b\right)\nabla_\theta \log p_\theta\,(\tilde{\mathbf{y}}\mid\mathbf{u})\,, \tag{25}$$

where the baseline b can be any function b that only depends on state θ. Reference [43] further proposes an optimization approach for sequence generation called **self-critical sequence training (SCST)**. The rationale behind this is that rather than estimating the baseline b by another model, we can just utilize the output of the sequence generator with its own test-time inference algorithm. Specifically, for patient embedding $\mathbf{u}$, we get the baseline output drug sequence $\hat{\mathbf{y}}$ by maximizing the output probability distribution of the same model at each time step, essentially performing a greedy search on the same model. The gradient $\nabla_\theta L(\theta)$ then becomes

$$\nabla_\theta L(\theta) \approx -\left(r\,(\tilde{\mathbf{y}}) - r\,(\hat{\mathbf{y}})\right)\nabla_\theta \log p_\theta\,(\tilde{\mathbf{y}}\mid\mathbf{u})\,. \tag{26}$$

Finally, based on Equations (13) and (14), we can formulate the loss function for the reinforcement learning as follows:

$$\begin{aligned} L_{RL} &= \left(r\,(\hat{\mathbf{y}}) - r\,(\tilde{\mathbf{y}})\right)\log p_\theta\,(\tilde{\mathbf{y}}\mid\mathbf{u}) \\ &= \left(r\,(\hat{\mathbf{y}}) - r\,(\tilde{\mathbf{y}})\right)\sum_{t=1}^{T}\log p\,(d_t\mid d_1,\ldots,d_{t-1},\mathbf{u}) \\ &= \left(r\,(\hat{\mathbf{y}}) - r\,(\tilde{\mathbf{y}})\right)\sum_{t=1}^{T}\log(softmax\,(\mathbf{W}_o\mathbf{h}_{t-1}+\mathbf{M}_{t-1})_{d_t}). \end{aligned} \tag{27}$$

4.4 Training and Testing Strategies

Here, we introduce the training and testing strategies of our DPG model, including how to train the model and how to generate the candidate drug package set and select the best drug package based on our model.

ALGORITHM 1: Training method for DPG

Input: Set of patient descriptions $\mathcal{U}_k$ and drug packages $\mathcal{P}_k$, drug relation matrix $\mathcal{R}$, hyper-parameters α, β, λ_1 and λ_2, batch size m, pretraining epoch number n', training epoch number n

Output: Model parameters Θ

Construct the drug interaction graph $\mathcal{G}$ based on $\mathcal{R}$;
Construct the DPG framework with an MPNN model for $\mathcal{G}$, a patient description embedding model, and a GRU model for package generation;
Initialize the parameters in DPG with random weights Θ;
for $i = 1$ *to* n' **do**
    Randomly select m patient descriptions and corresponding drug packages;
    Get the drug embeddings and interaction feature extractor from the MPNN model;
    Get the patient embeddings by Equation (11);
    Initialize the hidden state of the GRU model by the patient embeddings;
    Input *eos* to the GRU model and generate drug packages;
    Calculate L_{graph} and L_{MLE} based on Equations (9) and (19);
    Train the DPG model with $L_{pretrain} = L_{MLE} + \alpha * L_{graph} + \lambda_1 * \|\Theta\|_2^2$;
end
for $i = 1$ *to* n **do**
    Use the same method as the pretraining process to get the drug embeddings, patient embeddings and generate drug packages;
    Generate baseline drug package by greedy search;
    Calculate L_{graph} and L_{RL} based on Equations (9) and (27);
    Train the DPG model with $L_{DPG} = L_{RL} + \beta * L_{graph} + \lambda_2 * \|\Theta\|_2^2$;
end

4.4.1 *Training Strategies.* For learning the parameters of DPG, we propose to train the MPNN model, the patient description embedding model, and the drug package generation model simultaneously with a hybrid loss function. Specifically, we first pretrain the DPG model with MLE loss as

$$L_{pretrain} = L_{MLE} + \alpha * L_{graph} + \lambda_1 * \|\Theta\|_2^2, \quad (28)$$

where α is a hyper-parameter, which is used to control the trade-off between L_{MLE} and L_{graph}, and Θ is the parameter set. L_2 regularization is applied to prevent overfitting. Then, we train the DPG model with policy gradient as follows:

$$L_{DPG} = L_{RL} + \beta * L_{graph} + \lambda_2 * \|\Theta\|_2^2, \quad (29)$$

where β is also a hyper-parameter similar to α. Algorithm 1 describes the complete training method for our DPG model.

4.4.2 *Testing Strategies.* For the testing stage, multiple strategies are utilized as follows:

- **Greedy Search**, which could be the simplest recommendation method for our RNN-based model, where the model generates a drug package by maximizing the output probability distribution at each time step. However, the greedy search can only generate a candidate drug package set C with one drug package.

- **Beam Search**, which could generate multiple drug packages. Specifically, given the beam size n, the model keeps track of n states rather than just one. At each iteration, all the successors of all n states are generated, and the model selects the n best successors from the complete list. In this way, the model can generate a candidate drug package set C with n drug packages.
- **Neighbor Search**, which could be intuitive as existing drug packages given by human experts in the EMR dataset may also be beneficial for clinic treatment to other similar patients. To that end, we generate drug packages from the most similar patients based on the cosine similarity between patient embeddings.

Note that it is meaningless to generate a candidate set that only contains existing drug packages. Therefore, in neighbor search, the candidate set consists of both existing drug packages and packages generated by beam search simultaneously, and the proportion is a hyper-parameter.

Given the candidate set C, another important task is to select the best package from C. For beam search, we propose to select the best package based on the sequence probabilities and length normalization as follows:

$$\mathbf{y}_{best} = \arg\max_{y} \frac{1}{T^{\tau}} \sum_{t=1}^{T} \log p\left(d_t \mid d_1, \ldots, d_{t-1}, \mathbf{u}\right), \tag{30}$$

where T is the length of the sequence. Since the vanilla beam search has an undesirable effect where it unnaturally tends to prefer a very short result, we normalize the result by dividing T^{τ}, where τ is a hyper-parameter to control the punishment strength. However, this method cannot be utilized for the candidate set given by neighbor search, since the existing packages do not have sequence probabilities. Therefore, the final output package is selected by our preliminary discriminative model DPR [77] in neighbor search. The discriminative model is pretrained before utilization, and the parameters are not tied with our DPG model.

5 EXPERIMENTS

In this section, we evaluate the proposed model with a number of competitive baselines. Meanwhile, we will further present the discussions and case studies on drug package generation.

5.1 Data Statistics

Detailed statistics[4] of our APH dataset and the MIMIC-III dataset are shown in Table 4. Note that most of the drug packages contain synergistic drug pairs. Meanwhile, about 50% of the drug packages contain antagonism drug pairs, which indicates the necessity of personalized drug interaction modeling. We further found that the distribution of drugs in the APH dataset shows the long-tail distribution. As shown in Figure 4, most drugs are used infrequently. However, these low-frequency drugs may be critical for the treatment of certain rare diseases. Therefore, it is essential for our model to capture the characteristics of both high-frequency and low-frequency drugs.

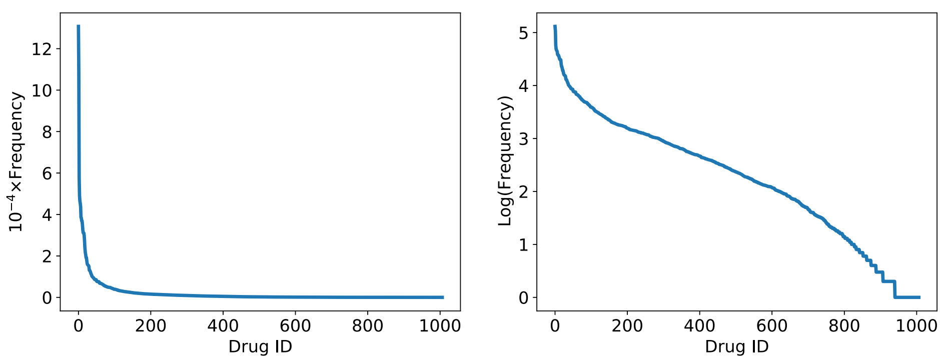
5.2 Experimental Settings

Here, we introduce the detailed settings of our experiments, including the baseline models and evaluation metrics and the details of the training stage.

[4]The private dataset is a little different from the dataset we used in Reference [77] due to further data collection and maintenance.

Table 4. Statistics of the Datasets

Description	APH	MIMIC-III
# of records	156,483	24,537
# of drugs	1,007	301
# of words in disease document	1,242	2,892
average size of drug packages	13	18.3
# of aligned drugs	565	64
# of drug pairs with No Interaction	2,560	0
# of drug pairs with Synergism	22,986	1,580
# of drug pairs with Antagonism	6,389	398
# of drug packages containing synergism drug pairs	118,758	19,188
# of drug packages containing antagonism drug pairs	86,212	11,709



(a) Drug frequency in the APH dataset.

(b) Drug frequency with log in the APH dataset.

Fig. 4. The statistics of drug frequency in the APH dataset.

5.2.1 *Baselines and Evaluation Metrics.* To evaluate the performance of our models for drug package generation, we selected several state-of-art methods as baselines. Specifically, we chose two popular traditional recommendation approaches:

- **NCF [21]:** NCF is a state-of-the-art deep-neural-network-based recommendation system, which replaces the inner product in matrix factorization with a neural architecture. This model recommends top-K drugs as packages for the patients in test sets based on the patient embeddings, where K is the average size of drug packages.
- **NN:** This method utilizes the pretrained patient embeddings based on NCF, and returns the drug package corresponding to the **Nearest Neighbor (NN)** by calculating the cosine similarity of patient embeddings.

Then, we chose several state-of-art discriminative package recommendation models as follows:

- **Package2vec:** Wan et al. [53] propose to utilize Item2vec [4] for enhancing the item embeddings in a package, and we extend Item2vec following Reference [29] to get the embedding of a package. NCF framework and BPR loss are utilized for training the package recommendation model.
- **LDA [7]:** This method utilizes the LDA model to get the embedding of a package and uses the same framework as Package2vec to recommend packages.

- **BR [42]:** BR is a package recommendation method that aggregates item latent vectors to get the package embeddings based on package size and item compatibility.
- **DAM [10]:** DAM is a neural network architecture for package recommendation that utilizes factorized attention network to get the embedding of packages.
- **DPR [77]:** DPR is our preliminary work for drug package recommendation, which considers the interaction effect between drugs, and the interaction effects could be affected by patient conditions. The DPR framework has two variants, i.e., **DPR on Weighted Graph (DPR-WG)** and **DPR on Attributed Graph (DPR-AG)**.

Finally, several generative drug recommendation models are chosen as follows:

- **GRU-MLE:** This model is a simplified variant of our models, which only uses the patient embedding as the initial hidden state and utilizes GRU as the generator. The model is trained by maximum likelihood estimation.
- **GRU-F:** This model uses the same method as GRU-MLE to generate drug packages. The difference is that the model is pretrained by maximum likelihood estimation, and further trained by policy gradient, where the reward is given by F1 score.
- **GRU-DPR:** This model uses the same method as GRU-MLE to generate drug packages. The model is pretrained by maximum likelihood estimation and further trained by policy gradient. Different from GRU-F, the reward is given by a pretrained discriminative package recommendation model DPR [77].
- **CGAN [14]:** Dai et al. [14] propose to utilize a new framework based on **Conditional Generative Adversarial Networks (CGAN)** for image captioning. Based on this framework, this method utilizes GRU as the generator and DPR as the discriminator, and trains the models following the GAN [18] framework by policy gradient.
- **KG-MIML-Net [45]:** KG-MIML-Net formulates the medicines prediction problem as a multi-instance multi-label learning task and solves this problem by an encoder-decoder model. The patient encoder utilized in KG-MIML-Net is a RNN-based model.
- **GAMENet [46]:** GAMENet integrates the drug-drug interactions knowledge graph by a memory module implemented as a graph convolutional network, and models longitudinal patient records as the query. The patient encoder utilized in GAMENet is an MLP model.
- **CompNet [55]:** CompNet uses a **Relational Graph Convolutional Network (R-GCN)** to encode the drug package at each time step and utilizes reinforcement learning for training. The patient encoder utilized in CompNet is an MLP model.

It is worth noting that the drug package recommendation is much different from the general recommendation, since there are no fixed users in our task. Therefore, in all of the baseline methods, we exploited the patient embedding model proposed in Section 4.2 to get the representation of patients. Furthermore, different from generative models, which can generate candidate drug package set following the methods in Section 4.4.2, all the discriminative models can only pick out the best package from a candidate set that consists of drug packages from 10 most similar patients.

To evaluate the quality of both the candidate sets and the selected best drug packages, different evaluation metrics were utilized. To evaluate the accuracy, comprehensiveness and diversity of the candidate set, we utilize the following evaluation metrics:

- **Set Precision, Set Recall and Set F1-value (S-Precision, S-Recall, S-F1)**, which means calculate the Precision, Recall and F1-value for each drug package in the candidate set, and calculate the average value to evaluate the accuracy of a candidate set.

- **Coverage**, which evaluate the comprehensiveness of a candidate set as follows:
$$Coverage = \frac{|(\mathcal{P}_1 \cup \mathcal{P}_2 \cup \cdots \cup \mathcal{P}_n) \cap \mathcal{P}_g|}{|\mathcal{P}_g|}, \tag{31}$$
where $\mathcal{P}_g$ is the ground truth package.
- **Diversity**, which is defined as 1-Jaccard Similarity over each drug package in the candidate set as follows:
$$Diversity = \frac{1}{n \times (n-1)} \sum_{\substack{1 \le i,j \le n \\ i \ne j}} \left(1 - \frac{|\mathcal{P}_i \cap \mathcal{P}_j|}{|\mathcal{P}_i \cup \mathcal{P}_j|}\right). \tag{32}$$

Note that each evaluation metric is averaged by the size of the test set. For the evaluation of the best drug package $\mathcal{P} \in C$ selected by the model, we propose to calculate the Precision, Recall and F1-value based on the ground truth package $\mathcal{P}_g$. Furthermore, to evaluate the ability of the models on recommending low-frequency drugs, we propose to remove the high-frequency drugs in both the ground truth package and the output package of the model, and calculate the Precision, Recall, and F1-value again for the evaluation.

5.2.2 Implementation Details. We implemented our model by PyTorch[5] and Pytorch Geometric[6] using a GeForce RTX 3090 GPU with 24 GB memory on a Linux machine. The parameters were all initialized using Kaiming [20] initialization. For the MPNN model on the drug interaction graph, we set the dimension of drug embeddings as 64. For the patient embedding model, we set the output dimension of the MLP, the dimension of char embeddings, and the hidden size of the GRU as 32, while the dimension of patient embeddings was set as 64. For the drug package generation model, we set the hidden size of the GRU as 32. For all the MLP models used in this article, we set the dimension of hidden layers as 128. In the process of model training, we used the Adam optimizer [24] for parameter optimization. We set the learning rate as 0.001 for pretraining and 0.0001 for training, and we set the mini-batch size as 1,024. The parameters of baselines were set up similarly to our method and were all tuned to be optimal to ensure fair comparisons. For the dataset splitting, we divided our dataset into 80%/10%/10% training/validation/test, and we report performance on the test set for the model that performed best on the validation set.

5.3 Discussions

5.3.1 Overall Performance. To demonstrate the effectiveness of our drug package recommendation framework, we compared DPG with all the baselines. First, we compared the ability of the models to recommend the best drug package for the patient. We selected greedy search as the generation method for generative models, and the results are shown in Table 5. From the results, we can get several observations:

(1) The performance of our models surpasses most of the baseline methods on different evaluation metrics. This clearly proves the effectiveness of our DPG framework based on reinforcement learning and message-passing neural networks.
(2) DPG performs better than GRU-F, which demonstrates the effectiveness of our method to capture the effect of the drug interaction and the influence of the patient condition explicitly.
(3) Generative models trained by maximum likelihood estimation or reinforcement learning with F1 score outperform all the discriminative models in most cases markedly. This verifies the superiority of utilizing generative models for the drug package recommendation task.

[5] https://pytorch.org/.
[6] https://github.com/rusty1s/pytorch_geometric.

Table 5. The Performance of Each Model on Recommending the Best Package

	APH			MIMIC-III		
model	Precision	Recall	F1-value	Precision	Recall	F1-value
NCF	0.4449	0.4044	0.4066	0.3997	0.3513	0.3634
NN	0.4658	0.4561	0.4439	0.3425	0.3264	0.3231
Package2vec	0.4730	0.4728	0.4550	0.3506	0.3319	0.3286
LDA	0.4745	0.4770	0.4591	0.3524	0.3319	0.3293
BR	0.4765	0.4760	0.4596	0.3583	0.3313	0.3313
DAM	0.4833	0.4874	0.4691	0.3565	0.3388	0.3350
DPR-WG	0.5146	0.4818	0.4797	0.3744	0.3295	0.3406
DPR-AG	0.5122	0.4880	0.4821	0.3756	0.3309	0.3420
CGAN	0.3914	0.3055	0.3251	0.2087	0.3172	0.2454
GRU-MLE	0.5633	0.5610	0.5462	0.4141	0.4651	0.4258
GRU-F	0.5961	0.5628	0.5639	0.4564	0.4394	0.4363
GRU-DPR	0.3124	0.2397	0.2510	0.1183	0.1772	0.1384
KG-MIML-Net	0.5590	0.5468	0.5369	0.3808	0.4497	0.4002
GAMENet	0.5700	0.5632	0.5508	0.4213	0.4601	0.4282
CompNet	0.5879	**0.5798**	0.5687	0.4571	0.4625	0.4488
DPG	**0.6060**	0.5738	**0.5740**	**0.4662**	**0.4698**	**0.4560**

We selected greedy search as the generation method for RNN-based generative models. Bold face indicates the best result in terms of the corresponding metric.

(4) Generative models trained by reinforcement learning with F1 score outperform all the other models. This demonstrates that reinforcement learning with a suitable reward function can effectively reduce the drug order's dependence, leading the performance much better than maximum likelihood estimation.
(5) The performance of GRU-DPR, which is trained by reinforcement learning with the reward given by a discriminative model, is extremely poor. The rationale behind this is that the discriminative model cannot be completely accurate, and this deviation will further make the generator perform worse due to the error accumulation. Although the GAN framework can alleviate this problem, the CGAN model cannot achieve satisfactory results and stucks in a bad local optimum. This indicates the effectiveness and accuracy of calculating the reward based on the ground truth drug package.

Then, we evaluated the quality of both the candidate sets and the selected best drug packages generated by different models. Note that only generative models can generate candidate sets, so we selected GRU-MLE, GRU-F, and DPG for evaluation. For greedy search, the size of each candidate set is 1. For beam search and neighbor search, we set the size of each candidate set as 6, while 50% packages are from existing packages in neighbor search. The results are shown in Tables 6 and 7. Furthermore, we evaluated the performance of each model on recommending low-frequency drugs based on the best packages selected by the models. We deleted the top 50% of all drugs that appeared most frequently in both ground truth packages and the generated ones, and calculated the Precision, Recall, and F1-value. The results are shown in Table 8. From the results, we can get the following conclusions:

(1) The Precision, Recall, and F1-value of the candidate sets generated by beam search are better than those generated by the neighbor search. Again, this demonstrates that generative models can generate more accurate drug packages than existing packages as shown in Table 5.

Table 6. The Performance of Each Model on Generating Candidate Sets

model	generation method	S-Precision	S-Recall	S-F1	Coverage	Diversity
GRU-MLE	Greedy Search	0.5633	0.5610	0.5462	0.5610	0.0000
	Beam Search	0.5547	0.5646	0.5437	0.6527	0.2290
	Neighbor Search	0.5196	0.5283	0.5078	0.7644	0.5373
GRU-F	Greedy Search	0.5961	0.5628	0.5639	0.5628	0.0000
	Beam Search	0.5902	0.5609	0.5601	0.6000	0.1260
	Neighbor Search	0.5382	0.5244	0.5152	0.7477	0.5118
DPG	Greedy Search	0.6060	0.5738	0.5740	0.5738	0.0000
	Beam Search	0.5979	0.5740	0.5704	0.6012	0.1011
	Neighbor Search	0.5405	0.5296	0.5189	0.7480	0.5049

Different generation methods are utilized for the generation.

Table 7. The Performance of Each Model on Selecting Best Packages

	Greedy Search			Beam Search			Neighbor Search		
model	Precision	Recall	F1-value	Precision	Recall	F1-value	Precision	Recall	F1-value
GRU-MLE	0.5633	0.5610	0.5462	0.5517	0.5841	0.5522	0.5265	0.5273	0.5108
GRU-F	0.5961	0.5628	0.5639	0.5933	0.5649	0.5641	0.5311	0.5220	0.5106
DPG	0.6060	0.5738	0.5740	0.6033	0.5764	0.5741	0.5310	0.5248	0.5120

Different generation methods are utilized for the generation.

Table 8. The Performance of Each Model on Recommending Low-frequency Drugs

	Greedy Search			Beam Search			Neighbor Search		
model	Precision	Recall	F1-value	Precision	Recall	F1-value	Precision	Recall	F1-value
GRU-MLE	0.0467	0.0426	0.0433	0.0074	0.0403	0.0116	0.0784	0.0747	0.0739
GRU-F	0.0080	0.0076	0.0078	0.0023	0.0077	0.0035	0.0734	0.0695	0.0686
DPG	0.0095	0.0091	0.0092	0.0085	0.0082	0.0083	0.0730	0.0697	0.0686

Different generation methods are utilized for the generation.

However, the Coverage and Diversity of the candidate sets generated by neighbor search are much better than beam search. This indicates that the drug packages generated by beam search tend to be consistent.

(2) Utilizing beam search can improve the performance of GRU-MLE on selecting the best packages effectively. However, this method does not work for models trained by reinforcement learning. The rationale behind this is that different from greedy search, beam search can keep track of more than one state. Therefore, it can reduce the impact on the result caused by the order of the drugs. However, reinforcement learning can solve this problem more effectively, hence it is meaningless to utilize beam search on GRU-F and DPG. Again, this demonstrates the effectiveness of reinforcement learning to reduce the dependence on the drug order, as shown in the previous discussion.

(3) From Table 8, we can find that there exists a trade-off between the accuracy of high-frequency and low-frequency drugs. Moreover, beam search will reduce the accuracy of recommending low-frequency drugs, since the best packages are selected by the joint

probability as shown in Equation (30), which discourages the model from selecting low-frequency drugs. Furthermore, we can find that the utilization of neighbor search can effectively improve the accuracy of recommending low-frequency drugs.

Based on the above discussion, we can finally propose the best strategy in practice to generate the candidate drug package set and the most suitable drug package for a new patient. For the candidate drug package set, we utilize DPG and neighbor search to generate several packages that contain both existing drug packages and packages generated by beam search at the same time. For the most suitable drug package, we select the best package generated by beam search. In this way, the model can generate a candidate set with high coverage and diversity as well as select packages with high precision and recall.

5.3.2 Ablation Study. To further validate the effectiveness of each component of our models, we also designed some simplified variants of our models as follows:

- **DPG-MLE:** This method is a simplified variant of DPG that only utilizes maximum likelihood estimation to train the model.
- **DPG-MLE w/o Mask:** This method is a simplified variant of DPG-MLE that deletes the mask layer in the calculation process.
- **DPG-MLE w/o Graph:** This method is a simplified variant of DPG-MLE that deletes the loss function for the MPNN on the drug interaction graph, i.e., $\alpha = 0$. In this way, the edge attributes do not contain the information of drug interaction type.
- **DPG-MLE-Raw:** This method is a simplified variant of DPG-MLE that deletes the drug interaction vector $\mathbf{i}$ in the calculation process. In this way, the model cannot capture the drug interaction explicitly.
- **DPG w/o Mask:** This method is a simplified variant of DPG that deletes the mask layer in the calculation process.
- **DPG w/o Graph:** This method is a simplified variant of DPG that deletes the loss function for the MPNN on the drug interaction graph, i.e., $\alpha = 0$ and $\beta = 0$.
- **DPG-Raw:** This method is a simplified variant of DPG that deletes the drug interaction vector $\mathbf{i}$ in the calculation process.
- **DPG w/o SCST:** This method is a simplified variant of DPG that deletes the self-critical baseline b in reinforcement learning.

The results of the ablation study are shown in Figure 5 from which we can draw the following conclusions:

(1) DPG-MLE performs better than DPG-MLE w/o Mask, and DPG performs better than DPG w/o Mask. This verifies that patient condition will influence the interaction effect between drugs.
(2) DPG-MLE performs better than DPG-MLE w/o Graph and DPG performs better than DPR w/o Graph. This demonstrates that the utilization of the drug interaction graph is significant and the message-passing neural network can capture the interaction between drugs effectively.
(3) Both DPG-MLE-Raw and DPG-Raw perform worst among the corresponding variants, which verifies our assumption that it is necessary to capture the effect of the drug interaction and the influence of the patient condition explicitly during the generation process.
(4) DPG outperforms DPG w/o SCST by a large margin. Furthermore, we find that the training process of DPG w/o SCST is very unstable. This demonstrates the effectiveness of the self-critical sequence training method on improving performance and reducing the variance.

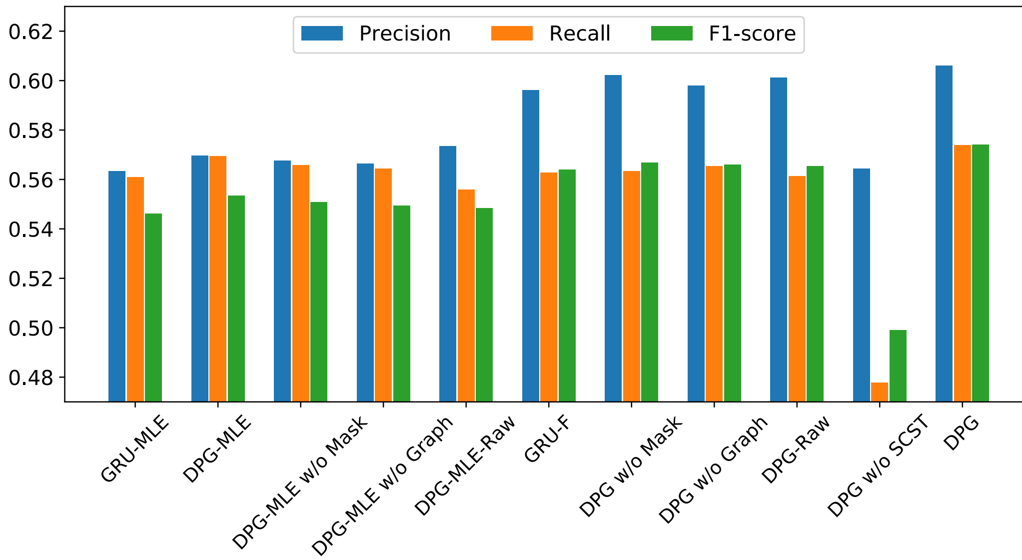


Fig. 5. The results of ablation study.

5.4 Parameter Sensitivity

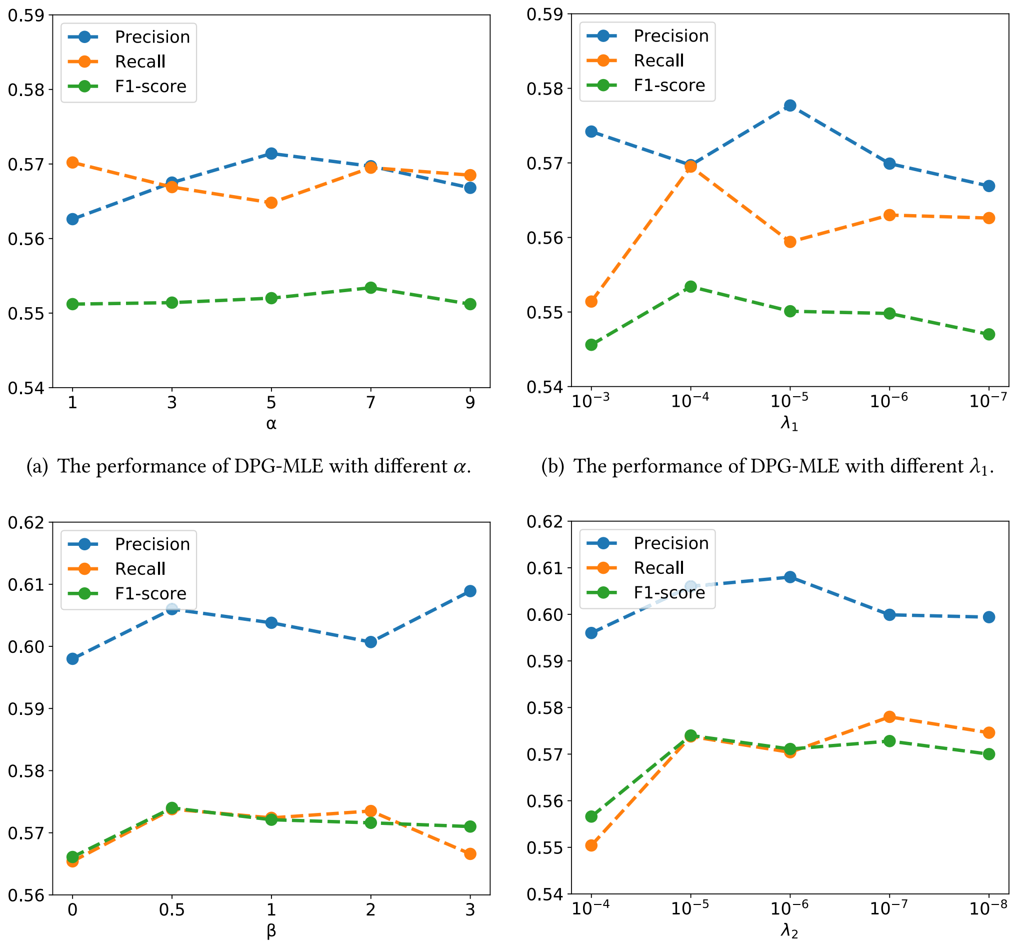
We evaluated how hyper-parameter hyper-parameters α, β, λ_1, and λ_2 affected the performance in this section, and the results are shown in Figure 6. First, we separately evaluated the effect of α and λ_1 on the pretraining process of DPG, i.e., the performance of DPG-MLE. We fixed $\lambda_1 = 10^{-4}$ during the evaluation process for α and $\alpha = 7$ during the evaluation process for λ_1. The results are shown in Figures 6(a) and 6(b). We can find that DPG-MLE performs best when we set $\alpha = 7$ and $\lambda_1 = 10^{-4}$. Furthermore, we can observe that the performance of DPG-MLE is good enough when α ranges from 1 to 9 and λ_1 ranges from 10^{-4} to 10^{-6}, which proves the robustness of DPG-MLE.

Next, we trained the DPG model based on the best DPG-MLE model. Similarly, we separately evaluated the effect of β and λ_2 on the training process of DPG. We fixed $\lambda_2 = 10^{-5}$ during the evaluation process for β and $\beta = 0.5$ during the evaluation process for λ_2. The results are shown in Figures 6(c) and 6(d). We can find that DPG-MLE performs best when we set $\beta = 0.5$ and $\lambda_1 = 10^{-5}$. We can observe that the performance of DPG is good enough when β ranges from 0.5 to 3 and λ_2 ranges from 10^{-5} to 10^{-8}. However, we can also find that DPG cannot perform well when λ_2 is too large. All the above experiments have proved that the models proposed in this article are robust enough, and the parameters are set in a reasonable range.

5.5 Case Study

In this part, we present some cases to illustrate the effectiveness of our model and reveal some interesting medical rules based on the derived insights on patient conditions and drug interaction.

5.5.1 Mask Vector Analysis. As mentioned before, we extracted the mask vector $\sigma\,(MLP\,(\mathbf{u}))$ of patient u to describe the impact of the patient condition. To analyze the effect of the mask vectors, we randomly selected 2,000 patients and their corresponding mask vectors, and projected them into two-dimensional space with t-SNE, which is proposed in Reference [39]. We further selected two representative patient groups with special needs for drugs based on common sense, respectively, pregnant women and infants (or young children), as well as two representative patient groups suffering from common diseases, respectively, patients with heart and stomach diseases.



(a) The performance of DPG-MLE with different α.

(b) The performance of DPG-MLE with different λ_1.

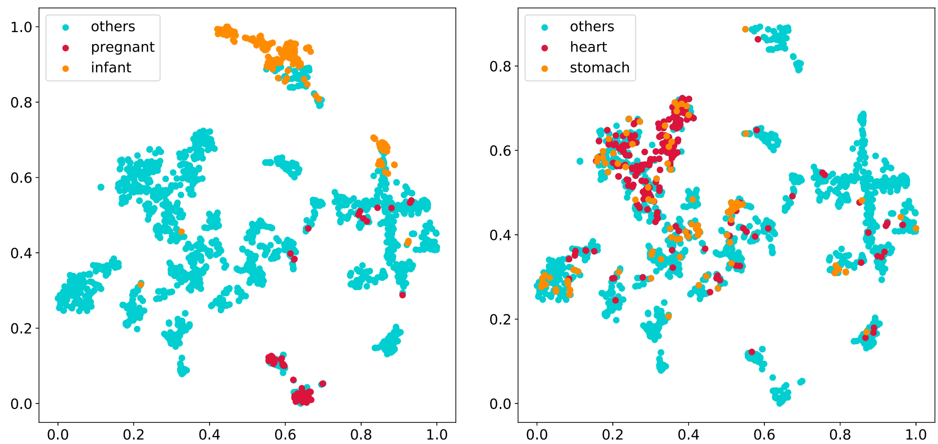
(c) The performance of DPG with different β.

(d) The performance of DPG with different λ_2.

Fig. 6. The performance of DPG-MLE and DPG with different hyper-parameters.

Figure 7 shows the visualization result. From Figure 7(a), we can find that the mask vectors of infants and pregnant women deviate the most from the vectors of other patients, which indicates that these two groups have special requirements for drug selection, and this is consistent with our common sense. Moreover, from Figure 7(b), we can find that due to the complexity and diversity of heart and stomach diseases, the mask vectors of patients with these two diseases are not visibly clustered together, and their special needs for medicines are personalized. We can further study the impact of patient conditions on drug selection by statistical methods such as clustering, which shows a great possibility of our method to help medical researchers.

5.5.2 Interaction Vector Analysis. We propose to calculate the drug interaction vectors to capture the drug interaction effect during the generation process in Section 4.3. To analyze the effect of the interaction vectors, we randomly selected 600 drug pairs, which contain 300 synergetic drug pairs and 300 antagonistic drug pairs, and calculated their corresponding drug interaction vectors. We further projected them into two-dimensional space with t-SNE and Figure 8 shows the



(a) Mask vectors of pregnant and infant patients. (b) Mask vectors of heart and stomach patients.

Fig. 7. Visualization of mask vectors. Different colored dots represent different types of patients.

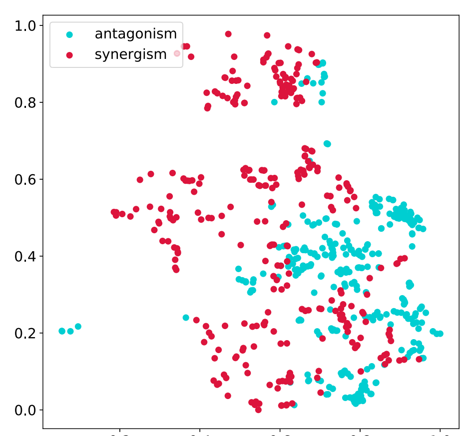


Fig. 8. Visualization of interaction vectors. Different colored dots represent different types of drug interaction.

visualization result. We can find that the interaction vectors corresponding to different interaction types form different clusters, which indicates that different interaction types have different influences on patients. This demonstrates that our model can capture the effect of drug interactions accurately.

5.5.3 Edge Attribute Analysis. In Section 4.1.2, edge attribute vectors are calculated to describe the interaction between two drugs. The attribute vectors are forced to contain drug interaction category information, and mask vectors are utilized to bring the impact of patient condition in Section 4.3. We propose that the mask vector plays a role by feature selection. If we multiply a contextual edge attribute vector $\tilde{\mathbf{e}}_{vu} = \mathbf{m} \odot \hat{\mathbf{e}}_{vu}$ with the classification transfer matrix $\mathbf{Q}$, then we can get a personalized drug interaction classification result, and we will illustrate this in this case study.

Table 9. Edge Attribute Analysis for Patient #28266

Drug 1	Drug 2	Type	$softmax\ (\hat{\mathbf{e}}_{vu}^\top \mathbf{Q})$	$softmax\ (\tilde{\mathbf{e}}_{vu}^\top \mathbf{Q})$
Promethazine	Cyclophosphamide	Synergism	[0.050, 0.950, 0.000]	[0.041, 0.959, 0.000]
Dexamethasone	Vindesine	Antagonism	[0.003, 0.007, 0.990]	[0.676, 0.042, 0.282]

$softmax\ (\hat{\mathbf{e}}_{vu}^\top \mathbf{Q})$ indicates the raw classification result while $softmax\ (\tilde{\mathbf{e}}_{vu}^\top \mathbf{Q})$ indicates the personalized classification result.

Table 10. Drug Package Generation Result for Patient #24595

Model	Result	Synergistic Drug Pairs	Antagonistic Drug Pairs
Ground Truth	Hexadecadrol, Tropisetron, Thalidomide, Pantoprazole, Rabeprazole	Thalidomide-Rabeprazole, Tropisetron-Rabeprazole	None
GRU	Zoledronate, **Hexadecadrol**, Omeprazole, Torasemide, **Tropisetron**, Endoxan	None	Hexadecadrol-Torasemide, Endoxan-Torasemide Tropisetron-Torasemide
DPG-MLE	**Hexadecadrol**, **Tropisetron**, Zoledronate, **Pantoprazole**	None	None
DPG	**Hexadecadrol**, **Tropisetron**, Zoledronate, **Pantoprazole**, **Rabeprazole**	Tropisetron-Rabeprazole	None

The generated results shown here are unordered. Bold face indicates drugs appeared in ground truth package.

We picked patient #28266 for detailed analysis. This patient was a 47-year-old man with lymphoma and had surgery in the hospital. We got the corresponding patient mask vector and drug interaction vectors by DPG. We also got the non-personalized and personalized drug interaction classification results for the drug interaction vectors. Table 9 shows two examples for this. We can find that Promethazine and Cyclophosphamide have a synergistic effect, and the initial drug interaction vector reflects this point. Furthermore, the mask vector keeps this feature, since these two drugs can enhance the sedation effect and treat cancer. In addition, Dexamethasone and Vindesine are marked as antagonistic, and this is reflected in the drug interaction vector. However, the mask vector weakened the antagonistic effect between these two drugs, since both the anti-inflammatory effect of dexamethasone and the anti-cancer effect of vindesine is very important for the patient. The above examples strongly confirm the effectiveness and interpretability of DPG from different perspectives.

5.5.4 Generation Result Analysis. First, we compared the drug packages generated by different models, respectively, GRU, DPG-MLE, and DPG, for the same patient to demonstrate the effectiveness of our model to select suitable drugs. We picked patient #24595 for detailed analysis. This patient was a 60-year-old woman with stomach and lung diseases, and the results are shown in Table 10. We can find that the drug package generated by GRU contains only two correct drugs. Furthermore, due to the insufficient ability to capture the effect of drug interaction, the model generated Torasemide, which is incorrect and has an antagonistic effect with several other drugs. Our DPG model, which utilizes interaction feature extractor and mask layer to capture drug interaction explicitly, solved this problem effectively and generated more correct drugs. Moreover, thanks to the intense power of reinforcement learning, we can find that the package generated by DPG is better than DPG-MLE.

Second, we compared the order of the sequences generated by DPG and DPG-MLE to show the influence of reinforcement learning on the generation order. We picked patient #135 for detailed analysis. This patient has myocardial infarction, and the results are shown in Table 11. We can

Table 11. Drug Package Generation Result for Patient #135

Model	Result in order
DPG-MLE	1/2 NS, Acetaminophen, **Aspirin EC**, NS, **Atorvastatin**, **Atropine Sulfate**, **Captopril**, Clopidogrel Bisulfate, **Docusate Sodium**, Eptifibatide, Furosemide, **Heparin**, Heparin Sodium, **Lisinopril**, Magnesium Sulfate, Metoprolol, Pantoprazole, Potassium Chloride, Senna
DPG	1/2 NS, **Captopril**, **Aspirin EC**, **Atropine Sulfate**, **Atorvastatin**, Senna, Acetaminophen, Clopidogrel Bisulfate, Eptifibatide, **Docusate Sodium**, Heparin Sodium, Furosemide, Pantoprazole, **Lisinopril**, Potassium Chloride, Metoprolol, **Heparin**, Magnesium Sulfate, NS

The generated results shown here are ordered. Bold face indicates drugs appeared in ground truth package.

find that although the drugs in the two drug packages are the same, the order of drugs generated by the two models is different. Furthermore, DPG tends to generate some important drugs, like Captopril, Aspirin EC, Atropine Sulfate, which can act directly on the heart to save the patient's life in the first few time steps, while the auxiliary drugs like Heparin in the last few time steps. This demonstrates that reinforcement learning can change the generation order of the same model, which further leads to better results.

6 CONCLUSION

In this article, we proposed a novel generative model named DPG to solve the problem of drug package recommendation. Specifically, we first proposed to construct a drug interaction graph based on the drug interaction data we collected from two large online pharmaceutical knowledge bases. Then, we utilized a message-passing neural network to learn drug embeddings that contain the interaction information between drugs. After that, we proposed a novel generative drug package recommendation framework named DPG, in which the drug interaction and the influence of the patient condition are captured explicitly by a mask layer. Furthermore, we proposed a training method based on both maximum likelihood estimation and reinforcement learning to reduce the dependence on the drug order. Finally, extensive experiments on a real-world data set from a first-rate hospital demonstrated the effectiveness of our DPG framework compared with several competitive baseline methods.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.* 17, 6 (2005), 734–749.

[2] Naveed Afzal, Sunghwan Sohn, Sara Abram, Christopher G. Scott, Rajeev Chaudhry, Hongfang Liu, Iftikhar J. Kullo, and Adelaide M. Arruda-Olson. 2017. Mining peripheral arterial disease cases from narrative clinical notes using natural language processing. *J. Vasc. Surg.* 65, 6 (2017), 1753–1761.

[3] Jinze Bai, Chang Zhou, Junshuai Song, Xiaoru Qu, Weiting An, Zhao Li, and Jun Gao. 2019. Personalized bundle list recommendation. In *Proceedings of the World Wide Web Conference*. 60–71.

[4] Oren Barkan and Noam Koenigstein. 2016. Item2vec: Neural item embedding for collaborative filtering. In *Proceedings of the IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP'16)*. IEEE, 1–6.

[5] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. 2013. Generalized denoising auto-encoders as generative models. In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13)*. Curran Associates, Red Hook, NY, 899–907.

[6] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. 2013. Generalized denoising auto-encoders as generative models. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 899–907. Retrieved from https://proceedings.neurips.cc/paper/2013/hash/559cb990c9dffd8675f6bc2186971dc2-Abstract.html.

[7] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.* 3, Jan (2003), 993–1022.

[8] Jean-Pierre Briot, Gaëtan Hadjeres, and François Pachet. 2017. Deep learning techniques for music generation - A survey. Retrieved from http://arxiv.org/abs/1709.01620.

[9] Jianxin Chang, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. 2020. Bundle recommendation with graph convolutional networks. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1673–1676.

[10] Liang Chen, Yang Liu, Xiangnan He, Lianli Gao, and Zibin Zheng. 2019. Matching user with item set: Collaborative bundle recommendation with deep attention network. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'19)*. 2095–2101.

[11] Miao Chen, Chao Wang, Chuan Qin, Tong Xu, Jianhui Ma, Enhong Chen, and Hui Xiong. 2021. A trend-aware investment target recommendation system with heterogeneous graph. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'21)*. IEEE, 1–8. https://doi.org/10.1109/IJCNN52387.2021.9533535

[12] Wanyu Chen, Fei Cai, Honghui Chen, and Maarten De Rijke. 2019. Joint neural collaborative filtering for recommender systems. *ACM Trans. Info. Syst.* 37, 4, Article 39 (Aug. 2019), 30 pages. https://doi.org/10.1145/3343117

[13] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). ACL, 1724–1734. https://doi.org/10.3115/v1/d14-1179

[14] Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. 2017. Towards diverse and natural image descriptions via a conditional GAN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'17)*. IEEE Computer Society, 2989–2998. https://doi.org/10.1109/ICCV.2017.323

[15] Ting Deng, Wenfei Fan, and Floris Geerts. 2013. On the complexity of package recommendation problems. *SIAM J. Comput.* 42, 5 (2013), 1940–1986.

[16] Steven E. Dilsizian and Eliot L. Siegel. 2014. Artificial intelligence in medicine and cardiac imaging: Harnessing big data and advanced computing to provide personalized medical diagnosis and treatment. *Curr. Cardiol. Rep.* 16, 1 (2014), 441.

[17] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning (ICML'17)*. JMLR.org, 1263–1272.

[18] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proceedings of the Annual Conference on Neural Information Processing Systems*, Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger (Eds.). 2672–2680. Retrieved from https://proceedings.neurips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html.

[19] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. MIT Press, 1024–1034.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*. 1026–1034.

[21] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. 173–182.

[22] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9, 8 (1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[23] Alistair E. W. Johnson, Tom J. Pollard, Lu Shen, H. Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. 2016. MIMIC-III, a freely accessible critical care database. *Sci. Data* 3, 1 (2016), 1–9.

[24] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR'15)*, Yoshua Bengio and Yann LeCun (Eds.). Retrieved from http://arxiv.org/abs/1412.6980.

[25] Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR'14)*, Yoshua Bengio and Yann LeCun (Eds.). Retrieved from http://arxiv.org/abs/1312.6114.

[26] Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. Retrieved from https://arXiv:1609.02907.

[27] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[28] Theodoros Lappas, Kun Liu, and Evimaria Terzi. 2009. Finding a team of experts in social networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 467–476.

[29] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the International Conference on Machine Learning*. 1188–1196.

[30] Dan Li, Tong Xu, Peilun Zhou, Weidong He, Yanbin Hao, Yi Zheng, and Enhong Chen. 2021. Social context-aware person search in videos via multi-modal cues. *ACM Trans. Info. Syst.* 40, 3, Article 52 (Nov. 2021), 25 pages. https://doi.org/10.1145/3480967

[31] Shuangli Li, Jingbo Zhou, Tong Xu, Dejing Dou, and Hui Xiong. 2021. GeomGCL: Geometric graph contrastive learning for molecular property prediction. Retrieved from https://arxiv.org/abs/2109.11730.

[32] Shuangli Li, Jingbo Zhou, Tong Xu, Liang Huang, Fan Wang, Haoyi Xiong, Weili Huang, Dejing Dou, and Hui Xiong. 2021. Structure-aware interactive graph neural networks for the prediction of protein-ligand binding affinity. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Feida Zhu, Beng Chin Ooi, and Chunyan Miao (Eds.). ACM, 975–985. https://doi.org/10.1145/3447548.3467311

[33] Shuangli Li, Jingbo Zhou, Tong Xu, Hao Liu, Xinjiang Lu, and Hui Xiong. 2020. Competitive analysis for points of interest. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1265–1274. https://doi.org/10.1145/3394486.3403179

[34] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. 2016. Gated graph sequence neural networks. In *Proceedings of the 4th International Conference on Learning Representations (ICLR'16)*, Yoshua Bengio and Yann LeCun (Eds.). Retrieved from http://arxiv.org/abs/1511.05493.

[35] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I. Sánchez. 2017. A survey on deep learning in medical image analysis. *Med. Image Anal.* 42 (2017), 60–88.

[36] Qi Liu, Yong Ge, Zhongmou Li, Enhong Chen, and Hui Xiong. 2011. Personalized travel package recommendation. In *Proceedings of the IEEE 11th International Conference on Data Mining*. IEEE, 407–416.

[37] Siqi Liu, Zhenhai Zhu, Ning Ye, Sergio Guadarrama, and Kevin Murphy. 2017. Improved image captioning via policy gradient optimization of SPIDEr. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'17)*. IEEE Computer Society, 873–881. https://doi.org/10.1109/ICCV.2017.100

[38] Yang Liu, Zhi Li, Wei Huang, Tong Xu, and En-Hong Chen. 2020. Exploiting structural and temporal influence for dynamic social-aware recommendation. *J. Comput. Sci. Technol.* 35 (2020), 281–294.

[39] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *J. Mach. Learn. Res.* 9 (Nov. 2008), 2579–2605.

[40] Tamara P. Miller, Yimei Li, Kelly D. Getz, Jesse Dudley, Evanette Burrows, Jeffrey Pennington, Azada Ibrahimova, Brian T. Fisher, Rochelle Bagatell, Alix E. Seif, et al. 2017. Using electronic medical record data to report laboratory adverse events. *Brit. J. Haematol.* 177, 2 (2017), 283–286.

[41] Aditya G. Parameswaran and Hector Garcia-Molina. 2009. Recommendations with prerequisites. In *Proceedings of the third ACM Conference on Recommender Systems*. 353–356.

[42] Apurva Pathak, Kshitiz Gupta, and Julian McAuley. 2017. Generating and personalizing bundle recommendations on steam. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1073–1076.

[43] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. IEEE Computer Society, 1179–1195. https://doi.org/10.1109/CVPR.2017.131

[44] Ruslan Salakhutdinov and Geoffrey E. Hinton. 2009. Deep boltzmann machines. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS'09) (JMLR Proceedings, Vol. 5)*, David A. Van Dyk and Max Welling (Eds.). JMLR.org, 448–455. Retrieved from http://proceedings.mlr.press/v5/salakhutdinov09a.html.

[45] Junyuan Shang, Shenda Hong, Yuxi Zhou, Meng Wu, and Hongyan Li. 2018. Knowledge guided multi-instance multi-label learning via neural networks in medicines prediction. In *Proceedings of the 10th Asian Conference on Machine Learning (ACML'18) (Proceedings of Machine Learning Research, Vol. 95)*, Jun Zhu and Ichiro Takeuchi (Eds.). PMLR, 831–846. Retrieved from http://proceedings.mlr.press/v95/shang18c.html.

[46] Junyuan Shang, Cao Xiao, Tengfei Ma, Hongyan Li, and Jimeng Sun. 2019. GAMENet: Graph augmented memory networks for recommending medication combination. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI'19), the 31st Innovative Applications of Artificial Intelligence Conference (IAAI'19), and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI'19)*. AAAI Press, 1126–1133. https://doi.org/10.1609/aaai.v33i01.33011126

[47] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the Annual Conference on Neural Information Processing Systems*, Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger (Eds.). 3104–3112. Retrieved from https://proceedings.neurips.cc/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html.

[48] Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning—An Introduction*. MIT Press. Retrieved from https://www.worldcat.org/oclc/37293240.

[49] Chang Tan, Qi Liu, Enhong Chen, Hui Xiong, and Xiang Wu. 2014. Object-oriented travel package recommendation. ACM Trans. Intell. Syst. Technol. 5, 3, Article 43 (Sept. 2014), 26 pages. https://doi.org/10.1145/2542665

[50] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural discrete representation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Curran Associates, Red Hook, NY, 6309–6318.
[51] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. Retrieved from https://arXiv:1710.10903.
[52] Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. 2016. Order matters: Sequence to sequence for sets. In *Proceedings of the 4th International Conference on Learning Representations (ICLR'16)*, Yoshua Bengio and Yann LeCun (Eds.). Retrieved from http://arxiv.org/abs/1511.06391.
[53] Mengting Wan, Di Wang, Jie Liu, Paul Bennett, and Julian McAuley. 2018. Representing and recommending shopping baskets with complementarity, compatibility and loyalty. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management.* 1133–1142.
[54] Meng Wang, Mengyue Liu, Jun Liu, Sen Wang, Guodong Long, and Buyue Qian. 2017. Safe medicine recommendation via medical knowledge graph embedding. Retrieved from https://arXiv:1710.05980.
[55] Shanshan Wang, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2019. Order-free medicine combination prediction with graph convolutional reinforcement learning. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM'19)*, Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu (Eds.). ACM, 1623–1632. https://doi.org/10.1145/3357384.3357965
[56] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'19)*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 950–958. https://doi.org/10.1145/3292500.3330989
[57] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 165–174.
[58] Zhengwei Wang, Qi She, and Tomás E. Ward. 2021. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Comput. Surv.* 54, 2, Article 37 (Feb. 2021), 38 pages. https://doi.org/10.1145/3439723
[59] Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.* 8 (1992), 229–256. https://doi.org/10.1007/BF00992696
[60] Libing Wu, Cong Quan, Chenliang Li, Qian Wang, Bolong Zheng, and Xiangyang Luo. 2019. A context-aware user-item representation learning for item recommendation. *ACM Trans. Info. Syst.* 37, 2, Article 22 (Jan. 2019), 29 pages. https://doi.org/10.1145/3298988
[61] Shiwei Wu, Joya Chen, Tong Xu, Liyi Chen, Lingfei Wu, Yao Hu, and Enhong Chen. 2021. Linking the characters: Video-oriented social graph generation via hierarchical-cumulative GCN. In *Proceedings of the ACM Multimedia Conference (MM'21)*, Heng Tao Shen, Yueting Zhuang, John R. Smith, Yang Yang, Pablo Cesar, Florian Metze, and Balakrishnan Prabhakaran (Eds.). ACM, 4716–4724. https://doi.org/10.1145/3474085.3475684
[62] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. 2021. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* 32, 1 (2021), 4–24. https://doi.org/10.1109/TNNLS.2020.2978386
[63] Min Xie, Laks V. S. Lakshmanan, and Peter T. Wood. 2014. Generating top-k packages via preference elicitation. *Proc. VLDB Endow.* 7, 14 (Oct. 2014), 1941–1952. https://doi.org/10.14778/2733085.2733099
[64] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep matrix factorization models for recommender systems. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'17)*, Vol. 17. 3203–3209.
[65] Pengcheng Yang, Fuli Luo, Shuming Ma, Junyang Lin, and Xu Sun. 2019. A deep reinforced sequence-to-set model for multi-label classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.* Association for Computational Linguistics, 5252–5258. https://doi.org/10.18653/v1/P19-1518
[66] Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. SGM: Sequence generation model for multi-label classification. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING'18)*, Emily M. Bender, Leon Derczynski, and Pierre Isabelle (Eds.). Association for Computational Linguistics, 3915–3926. Retrieved from https://www.aclweb.org/anthology/C18-1330/.
[67] Tianchi Yang, Linmei Hu, Chuan Shi, Houye Ji, Xiaoli Li, and Liqiang Nie. 2021. HGAT: Heterogeneous graph attention networks for semi-supervised short text classification. *ACM Trans. Info. Syst.* 39, 3, Article 32 (May 2021), 29 pages. https://doi.org/10.1145/3450352
[68] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence generative adversarial nets with policy gradient. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, Satinder P. Singh and Shaul Markovitch (Eds.). AAAI Press, 2852–2858. Retrieved from http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14344.

[69] Le Zhang, Tong Xu, Hengshu Zhu, Chuan Qin, Qingxin Meng, Hui Xiong, and Enhong Chen. 2020. Large-scale talent flow embedding for company competitive analysis. In *Proceedings of the Web Conference*, Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen (Eds.). ACM/IW3C2, 2354–2364.

[70] Le Zhang, Ding Zhou, Hengshu Zhu, Tong Xu, Rui Zha, Enhong Chen, and Hui Xiong. 2021. Attentive heterogeneous graph embedding for job mobility prediction. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'21)*, Feida Zhu, Beng Chin Ooi, and Chunyan Miao (Eds.). ACM, 2192–2201. https://doi.org/10.1145/3447548.3467388

[71] Ping Zhang, Fei Wang, Jianying Hu, and Robert Sorrentino. 2014. Towards personalized medicine: Leveraging patient similarity and drug similarity analytics. In *Proceedings of the AMIA Summit on Translational Science*. 132.

[72] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning-based recommender system: A survey and new perspectives. *ACM Comput. Surveys* 52, 1 (2019), 1–38.

[73] Weijia Zhang, Hao Liu, Yanchi Liu, Jingbo Zhou, Tong Xu, and Hui Xiong. 2020. Semi-supervised city-wide parking availability prediction via hierarchical recurrent graph neural network. *IEEE Trans. Knowl. Data Eng.* (2020), 1. https://ieeexplore.ieee.org/document/924142710.1109/TKDE.2020.3034140.

[74] Yutao Zhang, Robert Chen, Jie Tang, Walter F. Stewart, and Jimeng Sun. 2017. LEAP: Learning to prescribe effective and safe treatment combinations for multimorbidity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1315–1324. https://doi.org/10.1145/3097983.3098109

[75] Zhou Zhao, Zhu Zhang, Xinghua Jiang, and Deng Cai. 2019. Multi-turn video question answering via hierarchical attention context reinforced networks. *IEEE Trans. Image Process.* 28, 8 (2019), 3860–3872. https://doi.org/10.1109/TIP.2019.2902106

[76] Zhou Zhao, Zhu Zhang, Shuwen Xiao, Zhenxin Xiao, Xiaohui Yan, Jun Yu, Deng Cai, and Fei Wu. 2019. Long-form video question answering via dynamic hierarchical reinforced networks. *IEEE Trans. Image Process.* 28, 12 (2019), 5939–5952. https://doi.org/10.1109/TIP.2019.2922062

[77] Zhi Zheng, Chao Wang, Tong Xu, Dazhong Shen, Penggang Qin, Baoxing Huai, Tongzhu Liu, and Enhong Chen. 2021. Drug package recommendation via interaction-aware graph induction. In *Proceedings of the Web Conference 2021 (WWW'21)*. Association for Computing Machinery, New York, NY, 1284–1295. https://doi.org/10.1145/3442381.3449962

[78] Zhi Zheng, Tong Xu, Chuan Qin, Xiangwen Liao, Yi Zheng, Tongzhu Liu, and Guixian Tong. 2020. Multi-source contextual collaborative recommendation for medicine. *J. Comput. Res. Dev.* 57, 8 (2020), 1741–1754.

[79] Tao Zhu, Patrick Harrington, Junjun Li, and Lei Tang. 2014. Bundle recommendation in ecommerce. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 657–666.

Received 16 August 2021; revised 2 November 2021; accepted 10 January 2022