KETCH: Knowledge Graph Enhanced Thread Recommendation in Healthcare Forums

Limeng Cui
The Pennsylvania State University
University Park, PA, USA
lzc334@psu.edu

Dongwon Lee
The Pennsylvania State University
University Park, PA, USA
dongwon@psu.edu

ABSTRACT
Health thread recommendation methods aim to suggest the most relevant existing threads for a user. Most of the existing methods tend to rely on modeling the post contents to retrieve relevant answers. However, some posts written by users with different clinical conditions can be lexically similar, as unrelated diseases (e.g., Angina and Osteoporosis) may have the same symptoms (e.g., back pain), yet irrelevant threads to a user. Therefore, it is critical to not only consider the connections between users and threads, but also the descriptions of users’ symptoms and clinical conditions.

In this paper, towards this problem of thread recommendation in online healthcare forums, we propose a knowledge graph enhanced Threads Recommendation (KETCH) model, which leverages graph neural networks to model the interactions among users and threads, and learn their representations. In our model, the users, threads and posts are three types of nodes in a graph, linked through their associations. KETCH uses the message passing strategy by aggregating information along with the network. In addition, we introduce a knowledge-enhanced attention mechanism to capture the latent conditions and symptoms. We also apply the method to the task of predicting the side effects of drugs, to show that KETCH has the potential to complement the medical knowledge graph. Comparing with the best results of seven competing methods, in terms of MRR, KETCH outperforms all methods by at least 0.125 on the MedHelp dataset, 0.048 on the Patient dataset and 0.092 on HealthBoards dataset, respectively. We release the source code of KETCH at: https://github.com/cuilimeng/KETCH.

CCS CONCEPTS
• Information systems → Recommender systems; Personalization.

KEYWORDS
Recommendation system, online health forum, graph neural networks, medical knowledge graph

ACM Reference Format:

Table 1: Example threads on Diabetes (taken from MedHelp)

<table>
<thead>
<tr>
<th>Query</th>
<th>Reply post</th>
</tr>
</thead>
<tbody>
<tr>
<td>How low do I need to be in terms of blood sugars before I treat? I have been experiencing low blood sugars (sometime as low as 1.5 mmol/L)</td>
<td>You need to work out how much 1 g of glucose will raise you.</td>
</tr>
<tr>
<td>My DR. will allow me either pre-mixed insulin70/30 or Lantus + humalog, which best? Decide insulin preference.</td>
<td>Premixed insulin requires you to eat on a fixed schedule, and you cannot easily change your insulin dose.</td>
</tr>
</tbody>
</table>

1 INTRODUCTION
The wide usage of social media has promoted the creation and sharing of user-generated contents. Health information seeking behavior has become more common as social media empowers the public by providing access to vast knowledge and improving their decision making [3]. According to the health tracking survey1 from the Pew Research Center, information seeking on health subjects is one of the top online activities. Online health forum is a type of social media, where users share medical information by posting contents or comments in a collaborative and social manner. An online health forum such as Patient2 usually contains several channels, each of which is related to a single medical condition or drug. However, with the increasing volume of posts on online health forums, users inevitably face the information overload problem, which can prevent users from obtaining needed information in a timely manner. Researchers find that online health forum users who have to initially spend substantial effort in seeking information may not stay engaged in the long run [35]. Therefore, thread recommendation is critical in online health forums. Given a user post (i.e., query on the left hand side) in Table 1, our goal is to return relevant solved threads (on the right hand side) to the user.

Unlike the thread recommendation problem in other domains such as online courses [21, 39] and community question and answer [40], however, health thread recommendation has its unique challenges. First, as the symptoms experienced by patients with different clinical conditions are often overlapping, it is difficult to capture each user’s underlying cause (i.e., the disease or procedure). Many traditional approaches such as tensor factorization and topic model struggle to capture user intents behind, as they essentially use word co-occurrences to determine relevant answers [4, 32, 38, 39, 41].

2https://patient.info
In this paper, we propose a novel knowledge graph enhanced method for threads recommendation in online health forums, which can better capture connections between users and threads preference from both contextual and graph view. We propose a knowledge-enhanced text encoder to incorporate auxiliary information into the representations of threads and posts. We build a unified graph for user and thread, and compute node embeddings in the graph. The main contributions are threefold:

- We study a novel problem of thread recommendation in online health forums by leveraging knowledge graph to better capture user intents.
- We design a novel graph neural network based method KETCH (Knowledge Graph Enhanced Thread reCommendation in Healthcare Forums). In particular, we introduce the medical knowledge graph to capture user intents, and adopt the message passing idea to significantly enhance the learning of user and thread representations.
- We conduct extensive experiments on three real-world medical forum datasets to demonstrate the superiority of our method over several state-of-the-art methods. The reported results show that KETCH achieves a relative improvement of 0.125 on MedHelp dataset, 0.048 on Patient dataset and 0.092 on HealthBoards dataset comparing with the best results in terms of MRR. The case study shows the potential of KETCH in providing trustworthy and high-quality information.

2 RELATED WORK

This section briefly reviews two related topics: thread recommendation in online health forums and graph neural networks.

2.1 Thread Recommendation in Online Forums

A typical forum thread contains a query in its first post, and a discussion around it in subsequent posts. Thread recommendation is a special retrieval task with the first post as a query and the rest posts as a document [4]. It has been widely applied in online courses such as MOOCs [16, 21, 39] and community question and answer such as Quora and Reddit [13, 25, 30, 40, 42]. Existing methods can be broadly grouped into two categories: (1) the ones based on the topic similarity between the candidate threads and threads that user interacted in the past; and (2) the ones based on the network structure of users and threads.

Methods in the first category learn the embeddings of users based on their posts, and the recommend threads according to the post content. Halder et al. [10] propose an interest-aware topic model to align user’s self-reported medical conditions and treatments for recommendation. CLIR [22] uses LDA and CNN to match the candidate thread’s characteristics with users’ interests to make recommendations. Hansen et al. [13] quantify both discussion post similarities and social centrality measures on a Stack Overflow dataset. Halder et al. [11] use BiGRU to capture the concept dimension of user interests for recommending threads to users. Above methods model user preferences on the global level of a post. However, it is intuitive that a user is interested in a particular part of a post such as symptoms according to their own medical conditions. In this work, we will model the user preference based on their interactions with threads and learn user-specific thread representations for recommendation.
The second group of methods learns the interactions between the users and threads. Jiang et al. [14] present the forum data as a heterogeneous information network to extract the path-based features of users and threads. Kardan et al. [16] map users’ interactions to a social network and analyze the network to discover similar users for recommendation. However, these methods only consider the user-thread pair (two nodes with an edge) rather, while ignoring the high-order relations, which connect two nodes with multiple hops. Our model learns the high-order user-thread connectivity, which is essential for recommendation.

### 2.2 GNN based Recommendation

Graph Neural Networks (GNNs) refer to the neural network models which can operate on arbitrarily structured graphs [6, 7, 20, 31]. Due to their powerful capability of representation learning, GNNs have been widely applied in various domains such as computer vision [15, 28, 34], drug discovery [43], and chemistry [7, 9, 17]. Several extensions have been proposed to model the interactions between users and items for recommendation. These methods generally can be divided into two categories according to the use of side information. General recommendation methods without side information take the user-item bipartite graph as input, and propagate the information of nodes in the network. For example, a general inductive framework GraphSAGE [12] learns embeddings by sampling and aggregating features from a node’s neighbors for citation recommendation.

For side information based approaches [8, 36, 37], researchers incorporate social information of users, such as the influence of friends. DiffNet [36] extends GraphSAGE by adding users’ social relationships and behaviors. GraphRec [8] deploys a graph attention network on the users’ social network graph and the user-item bipartite graph separately, and concatenates the learned node embeddings for social recommendation. Besides user’s social relationship, DANSER [37] further considers the item-to-item relationship. Knowledge Graph (KG) can also be used as side information. Researchers take advantage of the rich information from KG to capture the potential connectivities between nodes. Wang et al. proposed KAGT [33] for product recommendation, which integrates the user-item bipartite graph and the KG into one, and adopts the attention mechanism to fully exploit the relationships between entities.

Hence in this paper, we introduce the KG into thread recommendation, aiming to improve recommendation performance in online health forums, and provide trustworthy and high-quality information simultaneously.

### 3 PROBLEM FORMULATION

In this section, we describe the notations and formulate the thread recommendation problem in online healthcare forums.

Given a forum thread set \( C \), a post set \( P \) and a user set \( U \), a user \( u_i \in U \) published a post \( p_j \in P \) in thread \( t_j \in C \). Based on this dataset, we treat the problem of thread recommendation as a specialized retrieval task with the first post of a thread as a query and each thread in our existing thread archive as a document. We estimate the probability of whether the user will reply to the target thread.

We denote \( U \) as a set of users \( u_i \in U \) with \( i \in \{1, \ldots, N_u\} \), \( C \) as a set of forum threads \( t_j \in C \) with \( j \in \{1, \ldots, N_t\} \) and \( P \) as a set of posts \( p_k \in P \) with \( k \in \{1, \ldots, N_p\} \). In order to model the complex interactions among the three types of nodes, we define the user-thread-post heterogeneous graph as follows.

**Definition 1. User Graph:** The user graph is an undirected graph denoted as \( G_U = (U, E_U) \), where \( E_U \) is the set of edges between users. Each edge connects two users that are friends in the forum. The edges are weighted with the number of interactions between users.

**Definition 2. Medical Knowledge Graph:** Let \( G_{mk} = (E, R, T) \) be a medical knowledge graph, where \( E \), \( R \) and \( T \) are the entity set, relation set and subject-relation-object triple set respectively.

Each post contains \( |P| \) words, \( p = \{w_1, w_2, \ldots, w_{|P|}\} \). We perform entity linking to build the word-entity alignment set \( \{(w, e)|w \in \mathcal{V}, e \in \mathcal{E}\} \). Each word can be linked to an entity in the medical knowledge graph.

**Definition 3. Post-Entity Bipartite Graph:** The post-entity bipartite graph is denoted as \( G_{pe} = (E \cup P, \mathcal{L}) \), where \( \mathcal{L} \) is the set of links. The link is denoted as \( \{(p, \text{Contains}, e)|p \in E, e \in \mathcal{E}\} \). If a post \( p \) contains a word that can be linked to entity \( e \), there will be a link between them, otherwise none.

The medical knowledge graph and the post-entity bipartite graph together can better capture latent user intent and unspecified medical conditions. Consider a user saying they have Elbow Pain during Urate lowering therapy. Two triples from the knowledge graph (Urate, Causes, Gout) and (Gout, Causes, Elbow Pain) can be linked to the Elbow Pain and Urate and point out that the user may suffer from Gout. As such, it makes more sense to recommend this user with a post suggesting Colchicine as a treatment due to the triple (Colchicine, Heals, Gout). Conversely, if the words between two different posts are not reachable in a knowledge graph, the two posts may be irrelevant. For example, both “Colchicine” and “Bisoprolol” co-occur with “elbow pain” a lot, there is no strong connection between the two entities themselves from a medical perspective. However, existing recommendation methods may regard “Colchicine” and “Bisoprolol” as related. Hence, we argue that incorporating a medical knowledge graph can provide useful complementary information and yield higher accuracy in medical thread recommendation.

With the above notations and definitions, now we formulate the thread recommendation task in online healthcare forums as follows:

**Problem 1 (Thread Recommendation in Online Healthcare Forums):** Given a thread set \( C \) and a set of users \( U \), the goal is to assign a relevance score to each thread \( t \) in the collection \( C \) to a user \( u \) based on the history of user-thread relationships. The recommended
where

which is defined as:

which learns the entity embeddings in KG and incorporates the
information into the thread textual representation; 2) a user preference
encoder, which propagates the information between the user and thread nodes to learn user-specific preferences; 3) a prediction
layer, which takes a user’s post query as input and returns a set of
threads in the archive with relevance scores. Then, we introduce
the details of each module.

4.1 Knowledge-enhanced Text Encoding

In this section, we propose Knowledge-guided Text Encoding Lay-
ers to guide the embedding of user posts. To fully utilize the medical
knowledge graph for healthcare forum post embedding, motivated
by previous work [5], we leverage the inherent directional structure
of a medical database to learn the entity embedding. To propagate
the information from knowledge graph to user posts, we incorpo-
rate the Post-Entity Bipartite Graph and Medical Knowledge Graph
into a unified relational graph, and add a set of self-loops (edge
type 0) denoted as \( S = \{(e, 0, e) | e \in E\} \), which allows the state of
a node to be kept. Hence, the new knowledge graph is defined as
\( G = \{E', R', T'\} \), where \( E' = E \cup P \), \( R' = R \cup \{\text{Contains}, 0\} \) and
\( T' = T \cup L \cup S \).

Knowledge graph embedding is a way to map the relations and
entities into a vector space, while preserving the structure informa-
tion of a graph. Here, we use TransR [24] to learn the embed-
ings. In TransR, for each triple \((e_h, r, e_t)\), the entity embeddings
\( e_h, e_t \in \mathbb{R}^d \) are mapped into relation \( r \)’s space via a projection
matrix \( W_r \in \mathbb{R}^{dm} \). The relation embedding \( e_r \in \mathbb{R}^m \) builds a trans-
bation between projected entities by optimizing the score function,
which is defined as:

\[
f_r(h, t) = W_r e_h + e_r - W_r e_t
\]

The training of TransR considers maximizing the discrimination
between correct triples and incorrect ones:

\[
L_{KG} = \sum_{(e_h, r, e_t) \in T} \sum_{(e'_h, r, e'_t) \in T^-} \lambda + f_r(e'_h, e'_t) - f_r(e_h, e_t)
\]

where \((e'_h, r, e'_t) \in T^-\) are incorrect triples constructed from cor-
correct triples \((e_h, r, e_t) \in T\) by replacing entities and \( \lambda \) is a configur-
able margin.

Knowledge Propagation Net: Considering an entity \( e_h \), we adopt
the message passing idea [2] to model the message transferred
from its neighboring nodes to the entity through the set of triples
\( T_h = \{(e_h, r, e_t) | (e_h, r, e_t) \in T'\} \):

\[
e^{(l)}_h = \sum_{(e_r, r, e_t) \in T_h} \delta^{(l)}_{ht} e^{(l)}_t
\]

where \( l \) is the depth and \( \delta^{(l)}_{ht} \) indicates how much information
being propagated from \( e_t \) to \( e_h \) in terms of relation \( r \). Higher \( l \) will allow
the information to propagate to higher-hop neighbors. We will
explore how \( l \) will influence the performance in Section 5.4.2.

is calculated as follows:

\[
\delta^{(l)}_{ht} = g(W_r e_h + e_r, W_r e_t)
\]

where \( g(\cdot) \) is a similarity function to measure the similarity of vec-
tors. Different similarity functions can be applied here, e.g. cosine
function.

We normalize the similarity scores across all triples connected
with \( e_h \):

\[
\delta^{(l)}_{ht} = \frac{\exp(\delta^{(l)}_{ht})}{\sum_{(h, r, t) \in T} \exp(\delta^{(l)}_{ht})}
\]

Post Encoder: We use BiGRU [1] to encode the text sequence from
both directions of words. Specifically, given the word embeddings \( \{v_1, v_2, \ldots, v_{|p_k|}\} \) of a post \( p_k \), the post embedding is computed as follows:

\[
\bar{p}_t = \text{GRU}(\bar{p}_{t-1}, v_t) \quad \bar{p}_t = \text{GRU}(\bar{p}_{t-1}, v_t)
\]

We concatenate the forward hidden state \( \bar{p}_t \) and the backward
hidden state \( \vec{p}_t \) as \( p_t = [\bar{p}_t, \vec{p}_t] \), which captures the contextual
information of the post centered around word \( v_t \).

Different words in the same sentence may have different informa-
tiveness in representing users and threads. For example, in sentence
"I’m with what has now become a chronic ectopic heart prob-
lem", the word "ectopic" is more important than the word "chronic"
in representing this problem. Thus, we use the attention
mechanism over word representations to learn informative post
representations by aggregating the important word embeddings.
The attention weight of the \( t \)-th word is computed as follows:

\[
q_t = \tanh(W_w p_t + b_w) \quad a_t = \frac{\exp(u_t^T q_t)}{\sum_{t=1}^{|p_k|} \exp(u_t^T q_t)}
\]

where \( W_w, b_w \) and \( u_w \) are trainable parameters, \( u_t \) is a hidden
representation of \( p_t \) obtained by feeding the hidden state \( p_t \) to a
fully embedding layer.

The final representation of the post \( p_k \) is the aggregation of the
contextual word representations weighted by their attention
weights: \( c_k = \sum_{t=1}^{|p_k|} a_t p_t \).

Knowledge-aware Attention: To incorporate the graph structure
into the textual representations, we replace the \( u_w \) in Eq. 7 by \( u'_w \)
to get the final attention function:

\[
u'_{t_l} = y u_{t_l} + (1 - y) W_g e_p
\]

where \( W_g \) is the node embedding of a post \( p \) obtained from the
Knowledge Propagation Net, \( W_g \) is a learnable transformation
matrix and \( y \in [0, 1] \) is a trade-off parameter that controls the relative
importance between two terms.

Thread Encoder: The thread encoder is used to learn represen-
tations of threads from the post representations. Similar to post
encoder, we use BiGRU to encode each post. Given the post embed-
ings \( \{e_1, e_2, \ldots, e_{\ell_k}\} \) in a thread \( t_k \), we capture the contextual
information in the post-level to learn the post representations \( s_k \)
from the learned post vector \( e_k \).

Intuitively, not all posts can equally contribute to the representa-
tion. For example, in a thread, some ask for clarifications about the
where $\mathbf{u}_t = \tanh (\mathbf{W}_c \mathbf{s}_t + \mathbf{b}_c)$
\[
\beta_0 = \frac{\exp(\mathbf{u}_t^T \mathbf{u}_c)}{\sum_{k=1}^{\vert j \vert} \exp(\mathbf{u}_t^T \mathbf{u}_k)}
\] (9)

where $\mathbf{u}_k$ is a hidden representation of $\mathbf{s}_k$ obtained by feeding the hidden state $\mathbf{s}_k$ to a fully embedding layer, and $\mathbf{u}_c$ is a trainable parameter to guide the extraction of the context.

The thread representation $\mathbf{d}$ learned from these posts is computed as: $\mathbf{d}_j = \sum_{k=1}^{\vert j \vert} \beta_k \mathbf{s}_k$.

### 4.2 User Preference Encoding

The user preference Encoder is to learn the representation of user preference by modeling their interactions with threads and posts in the User Graph. For the same health thread, different users may be attracted its different parts of the content according to their own preferences. In this section, we model individual preference on threads and posts in order to conduct personalized recommendation.

**User preference on threads:** The threads that a user engaged with reflect their interests. In our model, a user $\mathbf{u}_i$’s preference on threads $\mathbf{u}_i^t$ is modeled by aggregating the incoming message from all threads $\mathcal{N}_t$ that they reply. In the User Graph, each node is assigned to an initial representation $\mathbf{h}^{(0)}$. The message transferred from a thread $t_j \in \mathcal{N}_t$ to the user $\mathbf{u}_i$ in the $(l+1)$-th layer is:
\[
\mathbf{m}_{t_j \rightarrow u_i}^{(l+1)} = \mathbf{W}_m^{u} \mathbf{h}_j^{(l)}
\] (10)

where $\mathbf{m}_{t_j \rightarrow u_i}^{(l+1)}$ denotes the message from thread $t_j$ to user $\mathbf{u}_i$. $\mathbf{W}_m^{u}$ is a learnable weight parameter which maps the thread vector into the user embedding space. After the message passing step, we accumulate the incoming message at every user node by summing over all neighbors $\mathcal{N}_t$:
\[
\mathbf{u}_i^l = \sigma \left( \frac{1}{\vert \mathcal{N}_t \vert} \sum_{t_j \in \mathcal{N}_t} \mathbf{m}_{t_j \rightarrow u_i}^{(l+1)} \right)
\] (11)

where $\sigma(\cdot)$ denotes an activation function (we use LeakyReLU in this paper).

**User preference on posts:** The user preference on posts $\mathbf{u}_i^p$ can be learned similar to the user preference on threads, by aggregating the message passing from all the post written $\mathcal{P}_i$. The message that a post $p_t \in \mathcal{P}_i$ passes to a user $\mathbf{u}_i$ in the $(l+1)$-th layer is defined as:
\[
\mathbf{m}_{p_t \rightarrow u_i}^{(l+1)} = \mathbf{W}_m^{p} \mathbf{h}_j^{(l)}
\] (12)

where $\mathbf{W}_m^{p}$ is a learnable weight parameter which maps the post vector into the user embedding space.

Similar to Eq. 13, the user preference on posts is the aggregation of all the neighbor post information, which is defined as:
\[
\mathbf{u}_i^p = \sigma \left( \frac{1}{\vert \mathcal{P}_i \vert} \sum_{t_j \in \mathcal{P}_i} \mathbf{m}_{p_t \rightarrow u_i}^{(l+1)} \right)
\] (13)

### 4.3 Model Prediction

The user-specific representations of posts and threads are the concatenation of the representations learned from the knowledge-enhanced text encoder and user preference encoder: $\mathbf{p} = [\mathbf{u}_i^p, \mathbf{c}]$ and $\mathbf{t} = [\mathbf{u}_i^t, \mathbf{d}]$. Hence, given a post query $p_k$ of user $\mathbf{u}_i$, the existing threads can be recommended to the users through the similarity score, which is calculated by the dot product of the user-specific representation of post and thread: $(\mathbf{p}_k)^T \mathbf{t}_j$.

Following other recommendation methods [33], We use pairwise learning to train the model. For each post-thread pair, we add an unrelated thread $t'_j$ to create a training triple $(p_k, t_j, t'_j)$. The loss function is formulated as:
\[ L = \sum_{(p_k, s, t') \in O} -\ln \sigma \left( (p_k)'^T s - (p_k)'^T t' \right) \]  
where \( O \) denotes the set of triples for training.

### 4.4 Model Training

Finally, we combine the pairwise learning loss \( L \) with KG loss \( L_{KG} \) to form the final objective function as follows:

\[ L_{final} = L_{KG} + L + \eta \|\Theta\|_2^2 \]  
where \( \Theta \) is the model parameters, and \( \eta \) is a regularization factor. During the training, we optimize \( L_{KG} \) and \( L \) alternatively. We use Adam [19] to optimize the loss function. Adam can compute individual adaptive learning rates for different parameters, which has been widely used.

### 5 EXPERIMENTS

In this section, we present the experiments to evaluate the effectiveness of the proposed method. Specifically, we aim to answer the following evaluation questions:

- **EQ1**: Is KETCH able to improve thread recommendation performance by incorporating the medical knowledge graph?
- **EQ2**: How effective are knowledge graph and knowledge-aware attention, respectively, in improving the thread recommendation performance?
- **EQ3**: Can KETCH provide trustworthy and high-quality information to users?

### 5.1 Datasets

As the medical knowledge graph, we use a public medical knowledge graph KnowLife\(^3\). It is constructed from Web sources found in specialized portals and literature. There are 25,334 entity names and 591,171 triples. We use seven relations in KnowLife, including Causes, Heals, CreatesRiskFor, ReducesRiskFor, Alleviates, Aggravates and HasSideEffect.

To evaluate the performance, we gathered a collection of medical threads from three well-known online medical forums as follows.

- **MedHelp\(^4\)** is created in 1994, now has over 12 million users discussing their medical issues and looking for advice. We collected data from their Diabetes and Heart Disease Communities for our experiment. There are 5,234 threads including 98,731 posts.
- **Patient\(^5\)** is a subsidiary of EMIS Health, first launched in 1996. We collected 25,212 threads with 253,206 posts from Diabetes and Heart Disease forums on their website.
- **Healthboards\(^6\)** is a long-running social networking support group website, which consists of over 280 Internet message boards for patient to patient health support. We use the dataset collected by Mukherjee et al.\(^6\) [26].

The detailed statistics of the datasets are shown in Table 2. Following the existing work [11, 22], we removed threads that have less than 3 or more than 100 posts. Given a user’s post and a candidate thread, the user-thread pair is positive if the user replied to the thread before; otherwise, it is negative.

### 5.2 Baselines

We compare KETCH with representative and state-of-the-art forum thread recommendation algorithms, which are listed as follows:

- **ConvMF** [18]: ConvMF integrates convolutional neural network into probabilistic matrix factorization for review recommendation on Amazon and MovieLens. It captures contextual information of documents to enhance the rating prediction accuracy.
- **AMF** [39]: AMF uses a matrix factorization framework to make thread recommendations to users.
- **PVLM** [32]: The authors design several linguistic features and calculate the similarity between questions and threads to retrieve corresponding threads.
- **XMLC** [11]: XMLC uses stacked BiGRU for text encoding along with cluster sensitive attention to find the correlations between users and threads.
- **CVAE** [23]: CVAE is a collaborative variational autoencoder system that jointly models the generation of item content while extracting the implicit relationships between items and users collaboratively. It performs well on datasets of academic articles and their citations.
- **IATM-JNCTR** [10]: The authors use the topic model to capture the implicit interests embodied by users’ textual descriptions in their profiles and how users interact with various symptoms.

Note that for a fair comparison, we choose above contrasting methods that use features from following aspects: (1) only user-thread interactions, such as ConvMF and AMF; (2) only thread contents, such as PVLM, XMLC; and (3) both thread contents and user-thread interactions, such as CVAE, CLIR, IATM-JNCTR.

### 5.3 Experimental Setup

To evaluate the performance of thread recommendation algorithms, we use the following metrics, which are commonly used in related work [10, 22]: Recall, NDCG (Normalized Discounted Cumulative Gain) and MRR (Mean Reciprocal Rank). To evaluate the top-\( k \) results returned by the recommendation system, we use Recall@\( k \) and NDCG@\( k \) with \( k \in \{5, 10\} \).

We implement our model with Keras. We randomly split users into a training set (80%) and a test set (20%). Following the previous setting [10], users with fewer than five interactions always appear in the training set. We set the hidden dimension of our model to

<table>
<thead>
<tr>
<th>Dataset</th>
<th># Threads</th>
<th># Posts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medhelp</td>
<td>5,234</td>
<td>98,731</td>
</tr>
<tr>
<td>Patient</td>
<td>25,212</td>
<td>253,206</td>
</tr>
<tr>
<td>HealthBoards</td>
<td>5,584</td>
<td>1,048,576</td>
</tr>
</tbody>
</table>

Table 2: Statistics of datasets
The dimension of word embeddings is 100. We tested different learning rates of KETCH \( l_r = \{10^{-2}, 10^{-3}, 10^{-4}\} \) and depth \( l = \{1, 2, 3, 4\} \). For baseline methods, we follow the network architectures as shown in the papers.

5.4 Thread Recommendation (EQ1)
To answer EQ1, we first compare KETCH with the representative recommendation algorithms introduced in Section 5.2 and then investigate the performance of KETCH when using different depths \( l \).

5.4.1 Overall Comparison. Table 3 summarized the recommendation performance of all competing methods (reporting the average of 5 runs). From the table, we make the following observations:

- For user-thread interactions based methods, ConvMF and AMF, the performance is less than satisfactory. Both methods design a feature space to characterize user interests to match threads and users. AMF performs slightly better than ConvMF as AMF represents the content of the thread as a bag of words. However, both methods can not handle long threads very well as they do not fully utilize the linguistic information of threads.
- For thread contents based methods, PVLM and XMLC perform better than those methods purely based on user-thread interactions, which indicates these methods can utilize the semantic and linguistic clues in the threads. XMLC can better capture the word and sentence embeddings by using the BiGRU structure.
- Moreover, methods using both user-thread interactions and thread contents, CVAE, CLIR, IATM+JNCTR and KETCH, perform comparable or better than those methods using either one of them. This indicates that user–thread interactions and thread contents can provide complementary information, which can be both beneficial to thread recommendation.
- Generally, among the methods using both user-thread interactions and thread contents, we can see that KETCH consistently outperforms other methods in terms of Recall@k, NDCG and MRR on three datasets.
- KETCH achieves a bigger improvement on Healthboards and MedHelp than that on Patient in terms of recall and MRR. We assume that it is because the average number of posts in a thread on Healthboards and MedHelp is larger than that of on Patient. In terms of NDCG, KETCH has significant improvement on Patient instead of Healthboards and MedHelp, because as for Patient, the average number of threads that a user replied to is larger. This allows KETCH to better capture the rich information of the threads and returns more related threads first.

5.4.2 Performance Comparison w.r.t. Depth in KG. We vary the depth \( l \) of KETCH to investigate how KG improves the performance on three datasets. The larger \( l \) allows the information to propagate to further nodes. We search \( l \) in the set of \( \{1, 2, 3, 4\} \). As we did not get satisfying results on fourth- or higher-order interactions, we exclude those results. The results of Recall@10, NDCG@10 and MRR are listed in Figure 3. We can see that higher order of knowledge paths between entities can better improve the performance of KETCH. KETCH achieves the best results when \( l = 3 \).

5.5 Ablation Study (EQ2)
In order to answer EQ2, we explore each component of KETCH. We examine the components of the knowledge-enhanced text encoder and the user preference modeling by deriving several variants.

- w/o Preference: In this model, we simply use the knowledge-enhanced text encoder without user preference. This variant is designed to validate the effectiveness of user preference modeling in our model.
- w/o KG: This variant excludes the medical knowledge graph and knowledge-aware attention from the original model, and keeps the text encoder. We develop this variant to evaluate the necessity of the knowledge graph.
- w/o Post Node: This variant only considers two types of nodes, the thread and user nodes, in the user graph. We also remove the user preference on posts from the model. This variant is to investigate the effectiveness of modeling user preference on posts, as most existing methods only use user-thread pairs.

We summarize the results in Table 4 and have the following findings:

- When we solely use the text encoder without considering the user graph, the performance of KETCH largely degrades, which suggests the necessity of modeling user preference.
- Removing the knowledge graph and related knowledge-aware attention degrades the model’s performance, as the attention mechanism only considers the semantic clues instead of the relationships between medical conditions. The KG can provide useful side information, especially for forums like Patient where most users are not medical professionals.
- When we do not include post nodes in the user graph, we have no way of knowing how much interaction the user has with threads. Users submit more replies to the threads that they are more interested in. Thus, only considering the user–thread pair can not fully capture the user preference information.

Through the ablation study of KETCH, we can conclude that (1) knowledge-enhanced text encoder can contribute to thread recommendation in online health forums; (2) it is important to capture user preference on both thread- and post-level.

5.6 Case Study (EQ3)
In order to illustrate the potential of KETCH in providing trustworthy and high-quality information, we focus on two user studies: drug side-effect detection and user helpfulness prediction.

5.6.1 Drug Side-Effect Detection. The detection of drug side-effects is tightly linked to patient safety and pharmacovigilance. After the drug is on the market, it is very important to continuously improve and expand the drug list. Posts in online health communities often providing rich information about drug side-effects, such as "Been on bisoprolol for 6 weeks now, 3.75mg does it affect breathing? I only seem to get the tiredness but sometimes it feels you have slight asthma". In this section, we will explore whether the embeddings of drugs from KETCH can be used to identify the side-effects of drugs. We randomly delete some links between some drugs and their side-effects from Know-Life, and use this cut-down version of KG to train KETCH. Following
Table 3: Performance comparison on MedHelp, Patient and HealthBoards datasets.

<table>
<thead>
<tr>
<th>Method</th>
<th>MedHelp</th>
<th>Patient</th>
<th>HealthBoards</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Recall</td>
<td>NDCG</td>
<td>MRR</td>
</tr>
<tr>
<td></td>
<td>@5</td>
<td>@10</td>
<td>@5</td>
</tr>
<tr>
<td>ConvMF</td>
<td>0.018</td>
<td>0.035</td>
<td>0.097</td>
</tr>
<tr>
<td>AMF</td>
<td>0.053</td>
<td>0.092</td>
<td>0.121</td>
</tr>
<tr>
<td>PVLM</td>
<td>0.119</td>
<td>0.130</td>
<td>0.125</td>
</tr>
<tr>
<td>XMLM</td>
<td>0.180</td>
<td>0.220</td>
<td>0.086</td>
</tr>
<tr>
<td>CVAE</td>
<td>0.153</td>
<td>0.176</td>
<td>0.281</td>
</tr>
<tr>
<td>CLIR</td>
<td>0.203</td>
<td>0.242</td>
<td>0.172</td>
</tr>
<tr>
<td>KETCH</td>
<td>0.243</td>
<td>0.376</td>
<td>0.321</td>
</tr>
</tbody>
</table>

Improvement (%)

-19.70% 55.37% 14.23% -0.90% 29.69%
14.78% 19.86% 32.86% 14.89% 11.76%
64.33% 73.17% 12.23% 7.64% 27.38%

Table 4: Ablation study of KETCH demonstrated the advantage of modeling user preference and KG.

<table>
<thead>
<tr>
<th>Method</th>
<th>MedHelp</th>
<th>Patient</th>
<th>HealthBoards</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Recall</td>
<td>NDCG</td>
<td>MRR</td>
</tr>
<tr>
<td></td>
<td>@5</td>
<td>@10</td>
<td>@5</td>
</tr>
<tr>
<td>w/o Preference</td>
<td>0.115</td>
<td>0.128</td>
<td>0.258</td>
</tr>
<tr>
<td>w/o KG</td>
<td>0.276</td>
<td>0.347</td>
<td>0.313</td>
</tr>
<tr>
<td>w/o Post Node</td>
<td>0.225</td>
<td>0.359</td>
<td>0.325</td>
</tr>
<tr>
<td>KETCH</td>
<td>0.243</td>
<td>0.376</td>
<td>0.321</td>
</tr>
</tbody>
</table>

Improvement (%)

-11.96% 4.74% -1.23% 0.61% 3.41%
1.67% -1.53% -0.70% 1.69% 11.76%
8.40% 8.81% 8.33% 4.32% 3.13%

5.6.2 User Helpfulness Prediction. In the second user study, we evaluate how well our model can predict # thanks that a post received in a community. We extract a subset of user posts in the dataset with various # of thanks. Our goal is to predict the # thanks received by post through its text embedding. To compare with other baselines such as CVAE and CLIR, we get the post embeddings from the pre-trained models and them into a Logistic Regression classifier. We run the LR classifier using scikit-learn [27] with default parameter settings.

As HealthBoards does not have such information, we exclude HealthBoards from the user study. We use the # of thanks and whether the post is from a medical professional (1 and 0 are for from and not from) on Patient and MedHelp respectively. We use

Table 5: Performance comparison (RMSE) of predicting user helpfulness on MedHelp and Patient.

<table>
<thead>
<tr>
<th>Method</th>
<th>MedHelp</th>
<th>Patient</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVAE</td>
<td>1.295</td>
<td>1.335</td>
</tr>
<tr>
<td>CLIR</td>
<td>0.918</td>
<td>1.256</td>
</tr>
<tr>
<td>KETCH</td>
<td>0.835</td>
<td>1.117</td>
</tr>
</tbody>
</table>

Tang et al. [29], we take the embeddings from the Knowledge Propagation Net in Section 4.1 and feed them into an MLP (Multi-Layer Perceptron) to get the probability that a link between a drug and a side-effect exists. We also test the results with TransR and GCN [20] as embedding methods.

We test the above mentioned three embedding methods on MedHelp and Patient datasets. To evaluate the performance of drug side-effect detection, we use the following metrics, which are commonly used to evaluate classification performance: Accuracy and F1 Score. The results are shown in Figure 4. From the results, we can see that KETCH outperforms the two baseline models. It performs well in extracting information from the forum data.
The metric RMSE to evaluate the results in Table 5. We can see that KETCH outperforms the baselines in predicting user helpfulness.

5.6.3 Visualization. In order to illustrate the importance of knowledge graph for helping health thread recommendation, we use an example to show the triples captured by KETCH in Figure 5. The user posted in a thread if anaemia is related to Bisoprolol. They also replied to another heart disease thread where another user reported the use of Bisoprolol and several symptoms. From the KG, we can infer that both users (1) suffer from related symptoms: anaemia and muscle cramp; (2) might suffer from the side-effects of Bisoprolol (anaemia and joint pain); (3) are using Bisoprolol to treat heart disease. Thus, the KG can give us more additional information than the word co-occurrence to determine the relevant threads.

To visualize the effects of the KG, we randomly select a user and one of its positive relevant thread samples in the test set. As shown in Figure 5, the triples from KG can build connections between the user and the thread. The left part shows a prediction without the help of KG. “Bisoprolol” is the only overlapping word, resulting in a relatively low matching score between the two. The right part shows a higher matching score between the two after introducing triples from KG. KG can build multiple connections between the two (see the green dashed lines as examples). Thus, by providing additional information, the thread can be recommended to the user with higher matching score.

In addition, “Bisoprolol” has higher attention weights to the texts. The related triples (Bisoprolol, HasSideEffect, Anaemia) and (Bisoprolol, HasSideEffect, JointPain) can explain why the user and the thread are related. We can see that KETCH can not only perform thread recommendation but also yields the explanations of the recommendation results.

6 CONCLUSION

In this paper, we propose KETCH, a knowledge graph enhanced thread recommendation in online healthcare forums. KETCH leverages additional information from a medical knowledge graph to guide the text embedding with a knowledge-aware attention mechanism. The network also adopts the message passing idea to capture user preference on both thread- and post-level in order to better represent user intents. We conduct extensive experiments on three real-world medical forum datasets to demonstrate the strong performance of our method over several state-of-the-art methods. We also use two case studies to show the potential of KETCH in completing knowledge graph and promoting trustworthy information to users.

This work aims at healthcare domain and uses a medical knowledge graph to enrich the connections. In subsequent work, we plan to extend KETCH to other thread recommendation problems, such as MOOCs recommendation. Besides, several interesting future directions need to be investigated. First, we can consider the hierarchical structure of posts under each thread for better embedding. On most forums, users can either reply to the first message (query) or to a specific post in the thread. The latter creates a lot of sub-threads under each thread. We sort the posts according to their timestamps, without considering sub-threads in this paper. Second, we can further incorporate user credibility, such as certified health workers, to recommend trustworthy information to users. On forums like MedHelp, a lot of verified medical professionals interact with users and answer questions. Their answers have higher credibility and quality.

ACKNOWLEDGMENTS

This work was in part supported by NSF awards #1940076, #1909702, and #1934782.