LegalGNN: Legal Information Enhanced Graph Neural Network for Recommendation

JUN YANG, WEIZHI MA, and MIN ZHANG, Tsinghua University, China
XIN ZHOU, IBM Research, China
YIQUN LIU and SHAOPING MA, Tsinghua University, China

Recommendation in legal scenario (Legal-Rec) is a specialized recommendation task that aims to provide potential helpful legal documents for users. While there are mainly three differences compared with traditional recommendation: (1) Both the structural connections and textual contents of legal information are important in the Legal-Rec scenario, which means feature fusion is very important here. (2) Legal-Rec users prefer the newest legal cases (the latest legal interpretation and legal practice), which leads to a severe new-item problem. (3) Different from users in other scenarios, most Legal-Rec users are expert and domain-related users. They often concentrate on several topics and have more stable information needs. So it is important to accurately model user interests here. To the best of our knowledge, existing recommendation work cannot handle these challenges simultaneously.

To address these challenges, we propose a legal information enhanced graph neural network–based recommendation framework (LegalGNN). First, a unified legal content and structure representation model is designed for feature fusion, where the Heterogeneous Legal Information Network (HLIN) is constructed to connect the structural features (e.g., knowledge graph) and contextual features (e.g., the content of legal documents) for training. Second, to model user interests, we incorporate the queries users issued in legal systems into the HLIN and link them with both retrieved documents and inquired users. This extra information is not only helpful for estimating user preferences, but also valuable for cold users/items (with less interaction history) in this scenario. Third, a graph neural network with relational attention mechanism is applied to make use of high-order connections in HLIN for Legal-Rec. Experimental results on a real-world legal dataset verify that LegalGNN outperforms several state-of-the-art methods significantly. As far as we know, LegalGNN is the first graph neural model for legal recommendation.

CCS Concepts: • **Information systems → Recommender systems;**

Additional Key Words and Phrases: Legal information recommendation, heterogeneous environments, heterogeneous information network, graph neural network

33

This work is partly supported by the National Key Research and Development Program of China (2018YFC0831700, 2018YFC0831900), Natural Science Foundation of China (Grant Nos. 62002191, 61672311, 61532011) and Tsinghua University Guoqiang Research Institute. This article is based upon work supported by an IBM Global Academic Award and this project is also funded by China Postdoctoral Science Foundation (2020M670339). Dr. Weizhi Ma has been supported by Shuimu Tsinghua Scholar Program.

Authors' addresses: J. Yang, W. Ma, M. Zhang (corresponding author), Y. Liu, and S. Ma, Tsinghua University, 30 Shuangqing Rd, Haidian District, Beijing City, China; emails: yangjun18@mails.tsinghua.edu.cn, mawz@tsinghua.edu.cn, z-m@tsinghua.edu.cn, yiqunliu@tsinghua.edu.cn, msp@tsinghua.edu.cn; X. Zhou, IBM Research, Beijing City, China; email: zhouxin@cn.ibm.com.

1046-8188/2021/09-ART33 $15.00
https://doi.org/10.1145/3469887

ACM Reference format:
Jun Yang, Weizhi Ma, Min Zhang, Xin Zhou, Yiqun Liu, and Shaoping Ma. 2021. LegalGNN: Legal Information Enhanced Graph Neural Network for Recommendation. *ACM Trans. Inf. Syst.* 40, 2, Article 33 (September 2021), 29 pages.
https://doi.org/10.1145/3469887

1 INTRODUCTION

Legal-Rec is a specialized recommendation task, which aims to recommend potential helpful legal information to users. Legal-Rec is a significant task for two reasons: (1) Precise legal information needs is the intrinsic property of legal work. (2) Users suffer severe information overload caused by a massive increase in the number of new cases. Legal-Rec benefits legal practice by capturing domain specified and personal preference, which bridges the information gap. An ideal Legal-Rec model is capable of representing legal information, modeling domain-related user, and further providing the recommendation in the legal scenario.

Although previous methods achieve extraordinary performances in many recommendation scenarios (e.g., news recommendation, E-commerce), it is hard to adopt these models in the Legal-Rec scenario. The reason is that there are some important differences or new challenges existing in the Legal-Rec scenarios, such as more structural information (e.g., relationships between laws, cause of action), professional users (e.g., lawyers, judges), and so on. The three main challenges of Legal-Rec are summarized as follows:

How to integrate textual and structural legal information. Legal cases are well organized, as they have to follow some specific formulations, in which various legal concepts (e.g., *Corruption*, *Robbery* are criminal causes; *Home invasion* is a key element) will be included. On the one hand, as the criminal cause mentioned in legal cases are defined by laws, there are hierarchical and structural connections between concepts (e.g., *Corruption*, *Robbery* are defined in *Criminal law*; *Home invasion* is related to *Robbery*), which refers to **structural legal information** here. On the other hand, there are abundant **content features** in a legal case, not only the text in itself, but also the definition of legal concepts can be helpful content features. Both types of features are helpful for modeling legal cases, while we find that most of the previous recommendation studies only focus on one of them. For example, existing content-based methods concentrate on feature interaction [7, 15, 29], which is insufficient to model the connections between features. Graph-based recommendation methods either only adopt the structural information and ignore the textual features of the nodes, such as KGAT [45] and LightGCN [16], or integrate them by propagating node feature in the graph [21, 26], which is insufficient in utilizing structural connections. Thus, how to integrate textual structural legal information properly is an important challenge here.

Severe new item problem. New items usually attract more attention in the Legal-Rec scenario, as new cases represent the latest legal interpretation and legal practice, which leads to a critical new-item problem. The new law is different from older ones as judicial interpretations update. Outdated cases can be misleading, because they cite outdated laws, while the latest cases cite laws in force that are more valuable. There are more than 40,000 new cases in China[1] every day, and about 300 new legal opinions from federal and state courts in the United States[2] every day. However, it is not easy to model the new cases due to a lack of user-item interactions in the recommendation scenario, especially based on the collaborative filtering methods. Most of previous studies often adopt content-based methods to alleviate this problem, such as FM [29], NFM [15],

[1] https://wenshu.court.gov.cn/.
[2] https://www.courtlistener.com/.

Wide&Deep [7]. However, these methods model the new items independently, without considering the relationships between them and the previous ones with rich interactions. In the Legal-Rec scenario, legal concepts are commonly used in different legal cases, and hence can be the bridge to link the items. Therefore, to better address the new-item problem in Legal-Rec, we propose to better make use of the legal concepts information.

How to model the interests of expert users. Most users in Legal-Rec are experts, such as lawyers and judges. Comparing with other users, they often have clear information needs and more focused interests. So the key to providing better recommendations for them is to locate their specific information needs. A straightforward idea is to get more user' profile information, while it is hard in real scenarios due to user privacy constraints. We find that many legal information systems also provide a search engine for users, and the queries are strong indicators of the user's information needs. So in this study, we propose to introduce users' search history to model their preference for better recommendation results.

In this study, we design a new model, named LegalGNN, to cope with these challenges. First, we propose to construct a **Heterogeneous Legal Information Network (HLIN)**, which contains textual features, various types of nodes (users, cases, queries, legal concepts, and connections among them) to represent the features together. The introduction of user-query and item-query relationships in HLIN will help modeling user information needs for challenge III. Second, a unified representation module is proposed to combine content and structural information of nodes in HLIN to cope with challenge I, in which BERT[10] and TransR[22] are applied to extract and integrate representations. Third, to model the high-order relations between nodes in HLIN, **graph neural network (GNN)** is adopted for its remarkable performance in the recommendation, in which a new relational attention module is designed. Note that connections between new items and existing items are also taken into consideration in GNN, which is helpful to cope with challenge II. In summary, the proposed LegalGNN is able to tackle three main challenges in Legal-Rec. Experimental results in a real-world dataset illustrate the effectiveness of LegalGNN.

Our main contributions are summarized as follows:

- To the best of our knowledge, it is the first work that uses a heterogeneous information network to combine various features and adopts graph neural models in legal information recommendation. Besides, search history is first introduced to model user interest in the Legal-Rec scenario.
- A new **Legal information enhanced Graph Neural Network (LegalGNN)** is designed to cope with the main challenges in Legal-Rec, in which textual and structural information is combined by a unified representation module, and GNN with relational attention mechanism is applied to model high-order interactions.
- Experimental results in a real-world dataset verify the effectiveness of our model, which significantly outperforms some state-of-the-art methods.

The rest of this article is organized as follows: In the following section, three parts of related work are reviewed. Then, preliminaries of this study will be introduced in Section 3. In Section 4, we designed modules of the proposed LegalGNN that will be explained in detail, and some discussions about the model are listed in Session 5. In Section 6, we conduct various experiments and give some detailed analysis. The conclusions are made in Section 7.

2 RELATED WORK

In this section, we first introduce previous studies in the legal-Rec scenario. Then, some side-information enhanced recommendation methods are reviewed. Finally, graph neural networks–based recommendation methods are introduced.

2.1 Legal Information Recommendation

With an increasing amount of online legal information and users' legal information needs, some Legal-Rec attempts are conducted to bridge the information gap by capturing domain-specific and personal needs. Previous studies in the Legal-Rec address some characteristics of legal information, such as multi-topical content [23, 35] and abundant relationships [48]. So some topic model–based methods [2, 13, 23, 35] and graph-based methods [48] are proposed to capture these characteristics for legal recommendations. On the one hand, the topic-based methods, such as **Latent Dirichlet Allocation (LDA)**-based methods, are mostly used to model legal documents for recommendations [2, 13, 23, 35]. However, these methods ignore the connections of legal information (e.g., legal documents connect the *law* and *cause of action*). On the other hand, some work uses a graph to represent the relationships between legal information and conduct recommendations based on the graph [48]. For example, by constructing a citing graph between cases and laws, Winkels et al. [48] proposed a path weight–based method. However, the citing graph misses the relations in the legal knowledge, and the content of legal documents is ignored. In this study, we propose to construct a content-enhanced legal graph to represent both detailed content features and structural connections of legal knowledge. Note that to the best of our knowledge, although some work proposes to construct the legal knowledge graph [1, 8, 11] for information retrieval [11] and question answer [8], it is the first time that the legal knowledge graph is introduced into the Legal-Rec scenario.

2.2 Side Information Enhanced Recommendation

As there are abundant items and user features in real scenarios (e.g., item/user attributes, content information), content-based methods conduct recommendations by taking advantage of these features. Representative methods include **Factorization Machines (FM)** [29], **Neural Factorization Machines (NFM)** [15], Wide&Deep [7], **Deep&Cross Network (DCN)** [44], DeepFM [12], and DeepIM [55]. These methods concentrate on making full use of feature interaction and designing various structures. For example, NFM uses multi-layer perceptions under features bi-interactions. Besides, as the introduction of content features, most content-based methods are able to alleviate the cold-start problem. For example, ACCM [34] randomly drops historical interactions in training considering that cold items and users have rare interactions. Although these methods achieve notable performances, the structural connections between features are ignored, which may constrain their performance.

Some recent works introduce **knowledge graphs (KG)** into the recommender system as knowledge-related side information. Most of the KG-enhanced recommendation methods construct a **heterogeneous information network (HIN)** to connect the knowledge information with user-item interaction. The items in the primary user-item graph are first linked to entities in the knowledge graph. Random walk–based graph embedding methods first build node sequences via random walk, and then learn the node embedding by skip-gram method on these node sequences. To handle the multiple relationships, Metapath2vec [28] and HERec [33] use meta-path designed by experts in HIN. HRLHG [18] proposes a **Relation Type Usefulness Distributions (RTUD)** matrix and uses the value in the matrix as the probabilities to initiate random work. However, real-world network-structured applications, such as e-commerce, are much more complex and include not only multiple types of nodes and/or edges but also a rich set of attributes. To utilize the node embedding of the heterogeneous network with attributes. DKN[42] combined the neighbor of entities in the knowledge graph and text information to enhance news representation. GATNE-T [4] and GATNE-I [4] aggregate information from neighbor nodes and learn node embeddings by skip-gram and random walk methods. The base embedding of GATNE-T [4] is randomly initialed and trained by network structure while the base embedding of GATNE-I [4] is transformed by the node content feature. However, these methods use a multi-step pipeline, including random walk generation, and semi-supervised training is required where each step has to be optimized separately [20]. Translation-based graph embedding methods [3, 22, 47] are adopted to calculate the embeddings of users, items, and entities [41, 58]. For example, CFKG [58] generates user and item representations by TransE [3] and then conducts the recommendation-based on these representations. Besides, RippleNet [41] obtains users' interests along with links in KG, and the head entity is tuned by a weighted sum of connected tail entities where the weights are calculated in the relationship space. However, most existing KG-enhanced methods ignore the content feature of the nodes in the knowledge graph, which only adopts the structured information.

User behavior information is also an important feature for the recommendation, especially the search logs. The reason is that we can get more clear information needs from users' queries during the search session, which will be helpful to understand the users' interests and information needs in the recommendation scenarios. So some previous studies try to incorporate query logs to model user interests. For example, NRHUB [49] extracts the textual information of queries to enhance the representation of the user, and then adopts the new user embedding for the recommendation, which achieves significant improvements. However, NRHUB ignores the connections between queries and retrieved items. We believe that all the side information is helpful to model user interests and item embeddings, so comparing to previous studies that only focus on one type of side information, we propose to adopt them together. By constructing a HIN, we propose to not only combine content features and structural information of legal knowledge, but also user logs in legal searching into the HIN for Legal-Rec.

2.3 Graph Neural Network–based Recommendation

GNNs are deep learning–based methods motivated by CNN and graph embedding that operate on non-Euclidean graph domain [60]. Many types of GNN models are proposed in recent years, including **Graph Convolution Network (GCN)** [20], **Graph Attention Network (GAT)** [20], GraphSAGE [14], and so on. GCN [20] proposes a convolutional architecture via a first-order approximation. Through multi-layer information passing by feature transformation and neighborhood aggregation, GCN [20] encodes local node and structural feature by integrating information from high-order neighbors. To further improve the capability of graph neural network, attention mechanisms are employed in GAT [38] to provide the target node guided aggregation. GraphSAGE [14] increases scalability by designing a new sampling strategy. Besides, relations are considered in R-GCN [31], where the relation-specified transmission operation is deployed. These GNN methods are widely applied to recommendation, social network [14, 20], traffic prediction [9, 27], and graph representation tasks [54].

GNN-based recommendations have achieved great success recently [6, 25, 36, 45, 45, 52]. As early applications, GNN is directly deployed in the user-item bipartite graph to learn the representation of the user and item [25, 36, 52]. Recently, the KG is considered in GNN-based methods by combining the KG with the user-item bipartite graph [43, 45]. KGAT [45] obtains initial embeddings of nodes by TransR [22], and then aggregates high-order neighbor nodes with attention weights calculated by TransR. Besides, KGNN-LS [43] applies the label smoothness regularization in training. However, directly employing GNN in recommendation task suffers the huge parameters and over-fitting problem. To solve this problem, SGC [51] theoretically and experimentally proves that removing non-linearities and collapsing weight matrices do not negatively impact accuracy on downstream applications, which reduces the calculation complexity of GNN. Later, LR-GCCF [6] and LightGCN [16] provide further analysis and validate that simplified GNN can achieve comparable or even superior performance on the recommendation task. While these simplified methods

Table 1. Notations

Notation	Description
$\mathcal{U}$	the set of users
$\mathcal{V}$	the set of items
$\mathcal{Q}$	the set of queries
$\mathcal{E}$	the set of entity in the legal knowledge graph except item
$\mathcal{R}$	the set of relations in the heterogeneous legal information network
$\mathcal{R}_{KG}$	the set of relations in legal knowledge graph
$\mathcal{R}_{UBG}$	the set of relations in user behavior graph
$\mathcal{R}_{q,e}$	the set of relations between queries and entities
$\mathcal{G}_{KG}$	the legal knowledge graph
$\mathcal{G}_{UBG}$	the user behavior graph
$\mathcal{G}$	the heterogeneous legal information network (HLIN)
$\mathcal{S}_u$	Search history of user u
$\mathcal{D}_i$	Search result of query q_i
$\mathcal{I}_u$	User-item interaction history of user u
$\mathbf{e}_u^k \in \mathbb{R}^d$	the embedding of u in the kth layer
$\mathbf{e}_{\mathcal{N}_n^r}^k \in \mathbb{R}^d$	the relational neighbor embedding of u under relation r in the k-th layer
$\mathcal{R}_u$	the relation set that u connected
$\mathcal{N}_u$	the entity set that u connected
$\mathcal{N}_u^r$	the entity set that u connected by relation r
D_u	the embedding dimension of node u
$T(n)$	the content feature of node n

ignore multiple relations in the graph. And although these GNN models achieve strong performance by capturing graph structure, they integrate content feature by propagating node features on the graph [20]. Meanwhile, propagating node features on the graph is proved insufficient in integrating the content features of nodes [21, 26, 46]. So, in this study, we propose to design a new GNN-based recommendation method to work with these content features.

Some GNN studies also address the bias issue in GNN-based recommendation algorithms. STAR-GCN [56] addresses the label leakage bias in the user-item bipartite graph when applying GNN methods, where the prediction can be made according to the ground-truth edge between the user and positive items. To tackle this, STAR-GCN proposes a *sample-and-remove* strategy by removing the ground truth edge of sampled training pairs. However, the label leakage bias is a special case of the leakage path bias in our work, and the method is insufficient in solving the leakage path problem. Thus, we also design a new strategy to tackle the problem in this study.

3 PRELIMINARIES

First, we introduce some background about Legal-Rec, then we give the definition of the research problem in this work. The main notations used in this article are summarized in Table 1.

3.1 Backgrounds

3.1.1 User-item Interaction History. In a typical recommendation task, we have historical user-item interactions (e.g., clicking, purchasing, browsing). Here, we represent the interaction history of a user as a set of items. Formally, we have the user set $\mathcal{U}$ and the item set $\mathcal{V}$. The user-item

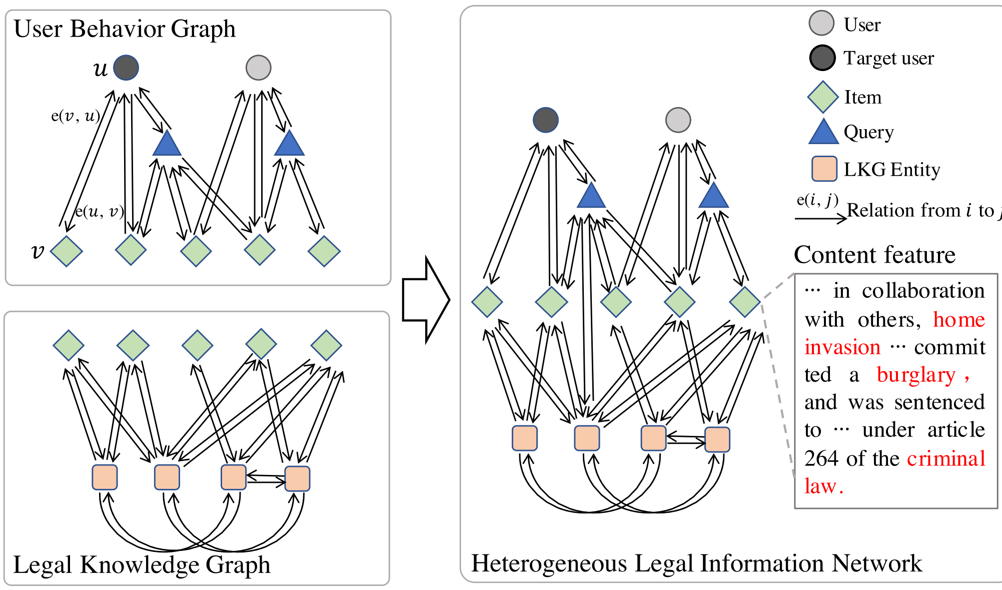


Fig. 1. Construction of Heterogeneous Legal Information Network.

interaction history of the user u is defined as $\mathcal{I}_u = \{v_1, v_2, \ldots, v_M\}$, where $u \in \mathcal{U}$ and $v_i \in \mathcal{V}$, and M is the size of the user-item interaction history.

3.1.2 *Search History.* The search history of a user is a set of queries submitted by the user and corresponding search results. For each query q_i, top N related items $\mathcal{D}_i = \{d_{i,1}, d_{i,2}, \ldots, d_{i,N}\}$ will be returned by the search engine, where $d_{i,j} \in \mathcal{V}$. A user may submit several queries, thus, we define the search history as a set of query and search result pairs $\mathcal{S}_u = \{(q_1, \mathcal{D}_1), (q_2, \mathcal{D}_2), \ldots, (q_K, \mathcal{D}_K)\}$, where K is the total number of queries submitted by user u.

3.1.3 *Legal Knowledge Graph.* In addition to the interaction history and search history of the user, we have the legal domain knowledge as the side information. The legal domain knowledge consists of textual legal concepts and structural connections between them. On the one hand, not only the textual legal concept itself, but also the definition of legal concepts can be helpful content features. For example, legal concept *Corruption* has the textual definition: *Corruption is a form of dishonesty or criminal offense undertaken by a person or organization entrusted with a position of authority, to acquire illicit benefit or abuse power for one's private gain.* So, we describe the content of legal concepts n as the content feature $T(n)$. On the other hand, structural connections in legal domain knowledge include item-entity alignments (e.g., a *case* v is related to *Criminal Law* and *Corruption*) and connections between legal concepts (e.g., *Corruption* is defined in *Criminal law*). Here, we organize the structural connections as a directed graph. Formally, the graph is represented as a set of triples $\{(h, r, t) | h, t \in \mathcal{V} \cup \mathcal{E}, r \in \mathcal{R}_{KG}\}$, where $\mathcal{R}_{KG}$ is the relation set including item-entity relations and relations among legal concepts, $\mathcal{E}$ is the legal concept set, and each triple (h, r, t) stands for a relation r from the head node h to the tail node t. For example, (*Corruption*, `defined_in`, *Criminal law*) shows the fact that the criminal cause *Corruption* is defined in *Criminal law*. It is worth noting that the legal knowledge graph is a directed graph, as Figure 1 shows, for each relation, the reverse relation is also considered, so the legal knowledge graph also contains (*Criminal law*, `definition_of`, *Corruption*). We use two arrows with opposite directions to represent these two relations.

The legal knowledge graph is defined as $\mathcal{G}_{KG} = \{(h, r, t)|h, t \in \mathcal{V} \cup \mathcal{E}, r \in \mathcal{R}_{KG}\}$, where each node n in the graph has content attribute $T(n)$.

3.2 Problem Definition

We formulate our recommendation task as follows:

Input: User history $\{(\mathcal{I}_u, \mathcal{S}_u)|u \in \mathcal{U}\}$, which includes user interaction history $\mathcal{I}_u$ and user search history $\mathcal{S}_u$; Legal knowledge graph $\mathcal{G}_{KG}$.

Output: The predicted score $y_{u,v}$, which stands the probability of user u may have an interaction with item v.

4 LEGAL INFORMATION ENHANCED GRAPH NEURAL NETWORK

In this section, we introduce LegalGNN, which consists of two important modules: (1) *Unified Content and Structure Representation Module*, (2) *Multi-relational GNN Module*. Figure 2 shows the overall structure of LegalGNN.

4.1 Model Overview

4.1.1 User Behavior Graph. Here, we construct the user behavior graph by connecting the user, query, and item nodes according to the relations in **user-item interaction history** and **search history**. As Figure 1 shows, the `interaction` relation exists between u and item v_j in interaction history $\mathcal{I}_u$. Besides, for each query and search result pair $(q_i, \mathcal{D}_i)$ in the search history $\mathcal{S}_u$, the `submit` relations exists between u and q_i, and relation `retrieval` exists between q_i and item v_j in $\mathcal{D}_i$. Formally, user behavior graph is defined as a directed graph $\mathcal{G}_{UBG} = \{(h, r, t)|h, t \in \mathcal{U} \cup \mathcal{V} \cup \mathcal{Q}, r \in \mathcal{R}_{UBG}\}$, where $\mathcal{R}_{UBG}$ contains canonical direction relation including `interaction`, `submit` and `retrieval` relations that mentioned before, and their reverse direction relations.

4.1.2 Heterogeneous Legal Information Network. Here, we define the concept of **Heterogeneous Legal Information Network (HLIN)**, which is constructed by merging the **User Behavior Graph** and the **Legal Knowledge Graph** into a unified graph.

As Figure 1 shows, $e(u, v)$ is the relation from node u to node v, which is in the opposite direction of $e(v, u)$. The two direct relations learned separately, so may have different effects on the target node. For each relation, the reverse relation is also considered. First, we integrate two graphs by aligning the items contained in both graphs. In this way, these items can connect entities in the user history graph and the legal knowledge graph. Then, we link the query node and the legal concept node if the legal concept is contained in the query text. It is worth mentioning that, compared with NRHUB, which uses the query as context text of the user, we introduce query into HLIN by linking queries with users, items, and entities in the legal knowledge graph, hence both structure connection and content information are considered. Finally, the heterogeneous legal information network is defined as a directed graph $\mathcal{G} = \{(h, r, t)|h, t \in \mathcal{U} \cup \mathcal{V} \cup \mathcal{Q} \cup \mathcal{E}, r \in \mathcal{R}_{KG} \cup \mathcal{R}_{UBG} \cup \mathcal{R}_{q,e}\}$, where $\mathcal{R}_{q,e}$ is the relation set between queries and entities. And each node n (except user) in the graph has additional content feature $T(n)$.

We now elaborate on the proposed LegalGNN model, which conducts recommendations based on HLIN. As Figure 2 shows, the main framework of LegalGNN consists of two components. The first component is the unified content and structure representation module. In this module, we first obtain the content feature of nodes by BERT, then we compress the content feature into an appropriate dimension by **Stacked Denoising Autoencoder (SDAE)**. Finally, by integrating

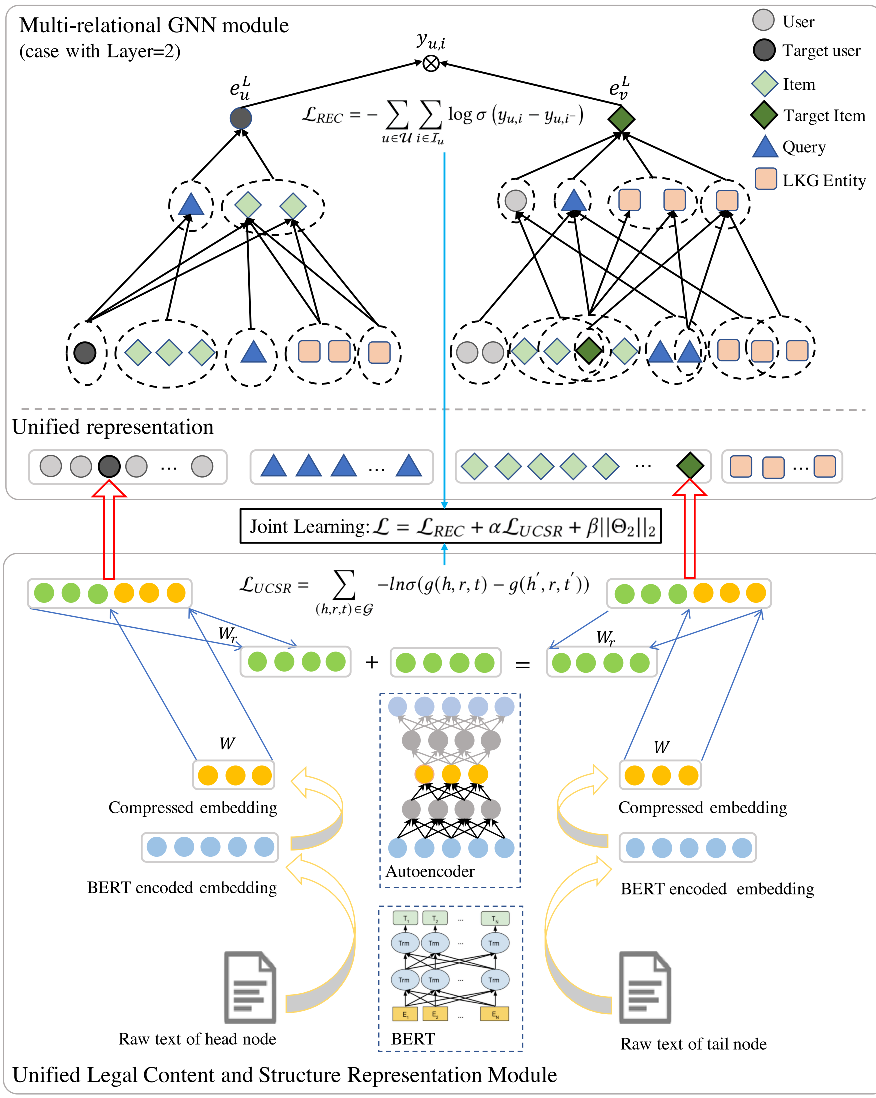


Fig. 2. Legal Information Enhanced Graph Neural Network for recommendation.

content feature and structural feature, we have the unified representation of nodes, which can be further processed in the downstream recommendation module. The second component is the multi-relational GNN module, which recursively propagates the neighbor embeddings to the target node by the **pooling** step and **aggregating** step alternately. In the pooling step, relational neighbor embeddings are obtained by pooling neighbor nodes under specified relations. In the aggregating step, relation-aware attention mechanism is employed to learn the weights of relational neighbor embeddings in aggregating.

4.2 Unified Legal Content and Structure Representation Module

Both structural connections and textual contents of legal information matter in Legal-Rec, so it is of importance to integrate these two types of information. Here, we propose a two-step method to get a unified representation of HLIN nodes. First, BERT is employed to calculate each node's content embedding and compress the content embedding into a low-dimension vector. Then, we design a unified representation module to integrate the content features and structural connections of nodes, which outputs a unified embedding for each node.

4.2.1 Content Feature Extraction. Considering the excellent performance of BERT [10] on many NLP tasks [32, 53, 57, 61], we apply BERT to encode legal information for its powerful capability in representing in-depth semantic information. To get a better embedding of legal information, we adopt the OpenCLaP [59] model, a pre-trained BERT model fine-tuned on a large legal corpus. In this way, the legal knowledge of the corpus is brought into content feature encoding. Precisely, the final representation of text is calculated by average pooling over the last layer embeddings of all token. Therefore, every text attached node n has content embedding e_n^{BERT}.

However, directly employing content embedding in the downstream neural recommendation model suffers an efficiency problem caused by its high dimension. To tackle the efficiency problem, a feature compression method is adopted to compress the content embedding into low-dimension embedding while minimizing the information loss.

Autoencoder is widely used in the recommendation task to compress features. For example, Ma et al. [24] employ SDAE [39] to compress the content information in the recommendation tasks. Following the previous work, SDAE is adopted in our model for its outstanding ability to compress features. SDAE is an unsupervised method that optimizes the model by minimizing MSE loss between the input vector and output vector after the decoder. SDAE consists of several layers of **Denoising Autoencoder (DAE)**. A two-layer SDAE is formalized as

$$x_1 = \sigma\left(\mathbf{W}_1^{encoder} x + b_1^{encoder}\right), \tag{1}$$

$$y = \sigma\left(\mathbf{W}_2^{encoder} x_1 + b_2^{encoder}\right), \tag{2}$$

$$y_1 = \sigma\left(\mathbf{W}_1^{decoder} y + b_1^{decoder}\right), \tag{3}$$

$$\overline{x} = \sigma\left(\mathbf{W}_2^{decoder} y_1 + b_2^{decoder}\right), \tag{4}$$

where $\mathbf{W}_i^{encoder}$ and $b_i^{encoder}$ are the encoder parameters, $\mathbf{W}_i^{decoder}$ and $b_i^{decoder}$ are the decoder parameters, σ is the nonlinear activation function. By minimizing the distance of input vector x and output vector $\overline{x}$, we obtain the compressed content feature y. The loss function is defined as

$$\mathcal{L}_{SDAE} = \sum_{n \in \mathcal{G}, n \notin \mathcal{U}} ||x_n - \overline{x_n}||^2, \tag{5}$$

where x_n is the input vector of node n, which refers to e_n^{BERT} in our work. By employing SDAE to compress content feature e_n^{BERT} in a flexible dimension that suits the downstream recommendation module, we obtain the fixed content feature e_n^{SDAE} of node n (except the user node) in HLIN. The content feature of the user is calculated as the mean of interacted item embeddings. Besides, SDAE is trained before the downstream model for efficiency consideration.

4.2.2 Unified Node Representation. The translation-based method is an effective and efficient way to represent node embeddings of the KG, which is widely used by KG-based recommendation methods [45, 58]. Following previous work, we adopt TransR [22] to capture the multi-relational connections of KG. By projecting head node embedding e_h and tail node embedding e_t into the relation space, we have the projected embedding e_h^r and e_t^r of the head node and tail node. Then, we optimize transformation parameters to approach $e_h^r + e_r \approx e_t^r$. Formally, for each triple (h, r, t) in the graph $\mathcal{G}$, the score function is defined as

$$g(h, r, t) = ||\mathbf{W}_r e_h + e_r - \mathbf{W}_r e_t||_2^2, \tag{6}$$

where $\mathbf{W}_r \in \mathbb{R}^{k \times d}$ is the transformation matrix of relation r that transfers the original d-dimension embedding of the node to a k-dimension relation space, and e_r is the embedding of the relation r.

To integrate content feature and structural connections, we concatenate them into a unified vector as follows:

$$e_h = e_h^c || e_h^s, \tag{7}$$

where $||$ means the concatenation operation, e_h^c is the content embedding of node h, and e_h^s is the structural embedding of node h. Because of the feature extraction module is not trained in an end-to-end way, e_h^{SDAE} is a fixed embedding. To integrate fixed content feature e_h^{SDAE}, a transfer matrix $\mathbf{Z}$ is employed to select useful information from the content feature. Formally, the content feature defined as

$$e_h^c = \mathbf{Z} e_h^{SDAE}. \tag{8}$$

Both content embedding e_h^c and structural embedding e_h^s are randomly initialized in training, and updated through backward gradient propagation. The integration of content and structural features lies in two aspects, on the one hand, content feature e_h^c extracts useful content feature from e_h^{SDAE} as well as absorbs structural information by TransR. On the other hand, benefiting from TransR, content embedding e_h^c and structural embedding e_h^s are projected and mixed in relation space to approach $e_h^r + e_r \approx e_t^r$, then content feature e_h^c has effect on structural feature e_h^s in unified representation learning. Notably, our method is different from KGAT [45], which ignores the content feature. Besides, structural feature e_h^s is considered in our model, rather than directly employing the node feature as the initial node embedding of GNN, which lacks a trainable structural embedding. Thus, our method considers both the content feature and structural feature. Besides, the weights of the content feature and structural feature can be adjusted with the length of two embeddings.

To train the unified node representation, we randomly collect positive triples and discriminate the positive and broken triples through a pairwise loss

$$\mathcal{L}_{UCSR} = \sum_{(h,r,t) \in \mathcal{G}} -ln\sigma(g(h,r,t) - g(h^{'},r,t^{'})). \tag{9}$$

For each triple (h,r,t) in $\mathcal{G}$, we randomly corrupt the head or tail node with equal probability. When the head node is corrupted, we construct a negative triple by replacing the head node h with a sampled node $h^{'}$, so $(h^{'},r,t) \notin \mathcal{G}$, and vice versa.

4.3 Multi-relational GNN Module

After constructing HLIN and learning the node embeddings by the unified representation module, we adopt GNN to capture the high-order connections of HLIN, which is widely employed by GNN-based recommendation methods.

However, the direct usage of traditional GNN suffers from the over-fitting problem due to the heavy parameter, which is even heavier in Legal-Rec due to the sparse interaction and new item problem. To tackle these problems, LightGCN proposes a simplified GNN that is suitable for recommendation tasks. Following the guidelines of LightGCN, we design a simplified GNN module.

Meanwhile, users prefer different behaviors in accessing information (e.g., submit queries or click recommended legal documents), and attributes have different impacts on items. Surrounding neighbors under different relations contribute differently to the target node in HLIN. So it is essential to consider relations in our model.

Based on the above two points, we propose a two-step multi-relational GNN module. As Figure 3 shows, first step, we obtain the relational neighbor representation according to different relations.

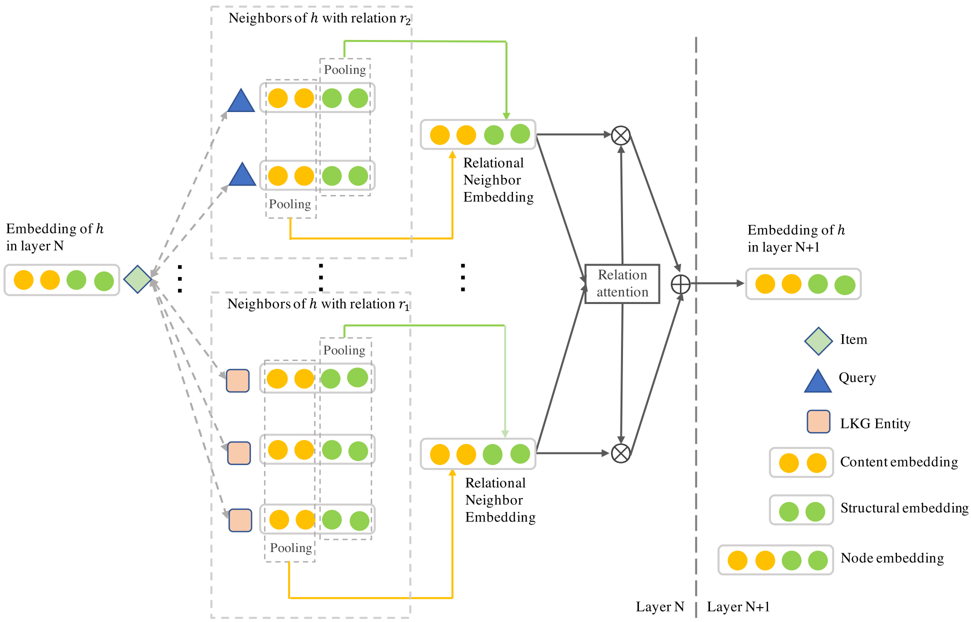


Fig. 3. Multi-relational aggregation.

In the second step, the target node embedding is calculated by aggregating the relational neighbor representations. We describe the process as follows:

$$e^k_{\mathcal{N}^r_u} = \text{POOLING}\left(\left\{e^k_j | j \in \mathcal{N}^r_u\right\}\right), \tag{10}$$

$$e^{k+1}_u = \text{AGG}\left(\left\{e^k_{\mathcal{N}^r_u} | r \in \mathcal{R}_u\right\}\right), \tag{11}$$

where $e^k_j \in \mathbb{R}^{d^{(k)}}$ is the embedding of node j in kth layer of graph neural network, $d^{(k)}$ is the dimension of node embedding in kth layer, and $e^k_{\mathcal{N}^r_u}$ is the relational neighbor representation of user u under relation r. POOLING stands for the pooling operation over the neighbor nodes under specified relation. AGG is an aggregation function that accumulates relational neighbor representations.

The difference between our two-step GNN and most traditional GNNs lies in that our model aggregates neighbor information at relation level without feature transformation matrices, while traditional GNN models either aggregate neighbor information at node level (e.g., GCN, GAT) or contain feature transformation matrices (e.g., R-GCN). Our method is able to extract main information under relations while avoiding the overfitting problem.

Here, we start by giving a detailed description of our two-step multi-relational GNN module, then we show how to generalize it to multiple layers and calculate the prediction score.

4.3.1 Relational Neighbor Embedding. Motivated by LightGCN, we adopt the pooling operation to get neighbor representation under specified relation, which is defined as

$$e^k_{\mathcal{N}^r_u} = \sum_{j \in \mathcal{N}^r_u} \frac{1}{c_{u,r,j}} e^k_j, \tag{12}$$

where $\mathcal{N}^r_u$ denotes neighbor set under relation r for target node u, and $c_{u,r,j}$ is the normalization constant. Similar to LightGCN, we assign $c_{u,r,j} = |\mathcal{N}^r_u|$. In this way, the relational neighbor representation is calculated as the sum of normalized feature vectors of neighbor nodes in terms of a specified relation.

4.3.2 Relational Attention. The second step of the propagation is accumulating relational neighbor representations to the target node. Users prefer different behaviors in accessing information (e.g., submit queries or click recommended legal documents), and attributes have different impacts on items. So, it is necessary to assign different weights to neighbor representations under a variety of relations, where the attention score is calculated by the coefficients of the target node embedding and the relational neighbor embeddings. In this way, the importance of different relations is learned, so the target node is able to obtain more information via high-related relations. The aggregating step is formulated as follows:

$$e_u^{k+1} = \text{LayerNorm}\left(\text{LeakyReLU}\left(\sum_{r \in \mathcal{E}_u} \alpha_{ur} e^k_{\mathcal{N}_u^r}\right)\right), \tag{13}$$

where α_{ur} is the attention weight indicating the quantity of information that target node u get from neighbors under relation r. We further employ the LeakyReLU nonlinearity and layer normalization to the node embedding after aggregating relational representation with attentional weights. Normalized coefficients across all connected relations are defined as

$$\alpha_{ur} = \text{softmax}_r(\pi_{ur}) = \frac{exp(\pi_{ur})}{\sum_{i \in \mathcal{E}_u} exp(\pi_{ui})}, \tag{14}$$

where π_{ur} is the coefficient between node u and $e^k_{\mathcal{N}_u^r}$ is the neighbor embedding under relation r in layer k. Here, we define a relation-aware coefficient function as

$$\pi_{ur} = \text{LeakyReLU}\left(h^r \left[\mathbf{W}_1 e_u^k || \mathbf{W}_2 e^k_{\mathcal{N}_u^r}\right]\right), \tag{15}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are two transformation matrices to extract information from target embedding e_u^k and relational neighbor embedding $e^k_{\mathcal{N}_u^r}$ separately. Besides, the attention parameter h^r for each relation is assigned individually to extract different information from relational neighbor representations, and then we apply the LeakyReLU nonlinearity.

The two steps of HRLHG are optimized separately: In the first step HRLHG obtains a global **Relation Type Usefulness Distributions (RTUD)** representing the usefulness of relations to the vertexes, while in the second step, HRLHG initiates random walk by RTUD and learns embedding of nodes by skip-gram. Compared to HRLHG, our two-step GNN is an end-to-end method, pooling and aggregating finished in one propagation of GNN.

GATNE-T and GATNE-I optimize by skip-gram, which brings the embedding of the target node and its context nodes in the random walk path closer. However, context nodes may have different influences on the target node but have the same weight here. In our GNN module, the weight of neighbor nodes is calculated by the target node supervised attention in every GNN layer. Besides, the base embedding of the node is either randomly initialized and trained by graph structure (GATNE-T) or transformed by node feature (GATNE-I), while our model fuses both structural feature and content feature by unified represent learning.

It is worth mentioning that our GNN model follows the principle of simplification and applies relation-aware attention. By removing the feature transformation matrix, our model is able to alleviate the over-fitting problem brought by the sparse interaction and new item problem in Legal-Rec. Meanwhile, users and items can select useful information from different relations by the relation aware attention, which can further provide more explainability.

4.3.3 Prediction. To explore higher-order propagation in HLIN, we stack more GNN layers that accumulate information from further neighbors. In each layer, we first calculate relational neighbor embeddings by POOLING, then obtain the target node embedding by relational attention function AGG. After L layer propagation, we obtain the final embedding of user u and target item v , noted as e_u^L and e_v^L.

Finally, the predicted score of user u and item v is calculated by inner product:

$$y_{u,v} = e_u^{L^T} \cdot e_v^L. \tag{16}$$

For the recommendation task, we optimize a pair-wise ranking loss [30] as follows:

$$\mathcal{L}_{REC} = -\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}_u} \log \sigma \left(y_{u,i} - y_{u,i^-}\right), \tag{17}$$

where σ denotes the sigmoid function and we randomly sample a negative item $i^- \notin I_u$ for each training instance.

4.4 Model Training

After constructing the HLIN $\mathcal{G}$ and calculating the text embedding by BERT, we conduct the training in two stages.

In the first stage, we compress the text embedding into a low-dimension vector by SDAE. The objective function is defined as

$$\min_{\Theta_1} \mathcal{L}_{SDAE} = \sum_{n \in \mathcal{G}, n \notin \mathcal{U}} ||x_n - \overline{x_n}||^2, \tag{18}$$

where Θ_1 is the parameter space, and the input is the content feature e_n^{BERT} encoded by BERT. After the training of SDAE, we obtain the content feature e_n^{SDAE} of n and save it for downstream modules.

In the second stage, to effectively learn recommendation task parameters, as well as integrate the content features and structural connections of nodes, we jointly learn the main recommendation task and unified representation task through a multi-task learning framework. Then the joint objective function is defined as:

$$\min_{\Theta_2} \quad \mathcal{L} = \mathcal{L}_{REC} + \alpha \mathcal{L}_{UCSR} + \beta ||\Theta_2||_2, \tag{19}$$

where Θ_2 is the parameter space, α is the weight of the unified representation loss in joint learning, and β is the weight of regularization.

To jointly train the recommendation task and unified representation task, we draw a mini-batch from $\mathcal{G}$ with the same proportions of a dataset as the recommendation task at each training step. In this way, all data are trained in one epoch for both recommendation and unified representation tasks. Besides, we adopt a sampling strategy similar to GraphSAGE [14] in the training of recommendation task. Specifically, the number of sampled neighbors is a constant and the neighbor number under different relations is averagely subdivided. Meanwhile, a debias strategy is designed to tackle the leakage path bias on the graph-enhanced method. A detailed description will be given in Session 5. The overall training procedure of LegalGNN is illustrated in Algorithm 1.

5 DISCUSSION

In this section, we first describe the bias problem in LegalGNN and our debias strategies. The relationship between our model LegalGNN and other graph-based neural networks is then discussed.

ALGORITHM 1: Learning algorithm for LegalGNN

Input: user-item interactions data $\bigcup_{u \in \mathcal{U}} I_u$; Heterogeneous Legal Information Network (HLIN) $\mathcal{G}$, where each node n has feature embedding e_n^{BERT} encoded by BERT; learning rate η; embedding size d; joint learning hyperparameter α; l2-normalization weight β
Output: model parameters Θ_1,Θ_2

Randomly initialize all parameters Θ_1,Θ_2
while *stopping criteria is not met* **do**
  Draw a mini-batch (n, e_n^{BERT}) from $\mathcal{G}$
  $\mathcal{L} \leftarrow \mathcal{L}_{SDAE}$
  Update model parameters Θ_1 according to $\mathcal{L}$ and optimizer
end while
while *stopping criteria is not met* **do**
  Draw a mini-batch (u, i, i^-) from $\bigcup_{u \in \mathcal{U}} I_u$
  Draw a mini-batch (i, j, i^-, j^-, r) from $\mathcal{G}$
  Compute $\mathcal{L}_{REC}$ according to Equation (17)
  Compute $\mathcal{L}_{UCSR}$ according to Equation (9)
  $\mathcal{L} \leftarrow \mathcal{L}_{REC} + \alpha \mathcal{L}_{UCSR} + \beta ||\Theta_2||_2$
  Update model parameters Θ_2 according to $\mathcal{L}$ and optimizer
end while
return Θ_1,Θ_2

5.1 Leakage Path Bias during Graph Modeling

When we build the user embedding via its multi-hops neighborhoods, the positive (or ground-truth) items may be involved in the neighborhoods through some specified paths. For example, when we aim to predict the interaction between user u and item v, the ground truth item is thus not only the target output but also the input feature. This may cause our model to overfit the items in the input features. For those negative items, since there is no connectivity between the user and negative items, it deteriorates the bias towards the positive items. Similar problems are also discussed in the previous work [56]. When applying to high-order relationships, this leakage bias also exists through some specified paths. For example, the positive (or ground-truth) items may connect to a user through some query nodes, which results in a bias towards the positive items. To make it clear, we generalize the leakage path bias issue as follows: Given user u and item v, if there exists a path connecting u and v, then the leakage path bias appears through this path.

To solve this problem, we provide a *break-path* strategy on the leakage path. Specifically, for a leakage path $P = (u, n_1, n_2, \ldots, n_k, v)$, we randomly drop an edge in P. The same drop probability is applied to negative items to keep the distribution of neighbor counts similar to positive items. After breaking the leakage path, our model avoids the over-fitting problem on the biased items and substantially boosts recommendation performance.

5.2 Relationships with Existing Graph Neural Networks

We conduct model analysis to demonstrate the difference between our LegalGNN and traditional GNN models. These models include GCN [20], GAT [38], R-GCN [31], and LightGCN [16]. We first describe the propagation function of these GNN models, then give a detailed discussion of them:

5.2.1 LightGCN. is a simplified GNN model for the recommendation. The target node embedding is calculated as the average of neighbor node embeddings in each GNN layer. The aggregation function of LightGCN is formulated as:

$$e_u^{k+1} = \sum_{j \in \mathcal{N}_u} \frac{1}{|\mathcal{N}_u|} e_j^k. \tag{20}$$

Then, we separate the aggregation function into two steps:

$$e_{\mathcal{N}_u^r}^k = \sum_{j \in \mathcal{N}_u^r} \frac{1}{|\mathcal{N}_u^r|} e_j^k, \tag{21}$$

$$e_u^{k+1} = \sum_{r \in \mathcal{E}_u} \frac{|\mathcal{N}_u^r|}{|\mathcal{N}_u|} e_{\mathcal{N}_u^r}^k. \tag{22}$$

As the formulas of LegalGNN show, the scale of the relational neighbor embedding is the fixed value $\frac{|\mathcal{N}_u^r|}{|\mathcal{N}_u|}$. While in our model, task-appropriate dynamic weights are learned by relational attention. Thus, our LegalGNN has a stronger learning ability than LightGCN.

5.2.2 *GCN and R-GCN.* The aggregation function of GCN is formulated as

$$e_u^{k+1} = \sigma\left(\sum_{r \in \mathcal{E}_u} \sum_{j \in \mathcal{N}_u^r} \frac{1}{\sqrt{|N_u|}\sqrt{|N_j|}} \mathbf{W}^k e_j^k + \frac{1}{|N_u|} \mathbf{W}^k e_u^k\right). \tag{23}$$

The aggregation function of R-GCN is formulated as:

$$e_u^{k+1} = \sigma\left(\sum_{r \in \mathcal{E}_u} \sum_{j \in \mathcal{N}_u^r} \frac{1}{c_{u,r,j}} \mathbf{W}_r^k e_j^k + \mathbf{W}_0^k e_u^k\right). \tag{24}$$

There are mainly two differences between our LegalGNN and these two GNN models. First, we remove the feature transformation matrix, which is verified by LightGCN that has little effect on the recommendation task. And the feature transformation matrix may result in over-fitting. Second, we employ the target node supervised attention aggregation. Different users may choose the same item for personalized interests in Legal-Rec, so we apply the target node guided attention mechanism to precisely model the user interests.

5.2.3 *GAT.* The Aggregation Function of GAT is Formulated as

$$e_u^{k+1} = \sum_{r \in \mathcal{E}_u} \sum_{j \in \mathcal{N}_u^r} \alpha_{uj} \mathbf{W} e_j^k + \alpha_{uu} \mathbf{W} e_u^k, \tag{25}$$

and our LegalGNN can be formulated as

$$e_u^{k+1} = \sum_{r \in \mathcal{E}_u} \sum_{j \in \mathcal{N}_u^r} \frac{\alpha_{ur}}{|\mathcal{N}_u^r|} e_j^k. \tag{26}$$

Besides the difference in the feature transformation matrix, GAT calculates the attention weights of each neighbor individually, while LegalGNN shares the attention weights for neighbors in the same relation, which increases the robustness of the model.

6 EXPERIMENTS AND ANALYSES

In this section, we introduce our experimental settings and results. Our experiments are designed to answer the following research questions:

- **RQ1:** How effective is our proposed LegalGNN model in the Legal-Rec scenario compared to state-of-the-art recommendation methods?

Table 2. Dataset Statistics

User-Item Interactions	#users	3,327
	#items (interacted)	60,100
	#interactions	64,642
	#items (all)	343,244
	density	0.00032
HIN Statistics	#node type	7
	#relation type	14
	#node number	436,161
	#relation number	5,354,935

- **RQ2:** Are the integration of content and structural features helpful in the model?
- **RQ3:** What are the impacts of incorporating query and relational graph neural networks in our model?
- **RQ4:** How do the hyper-parameters influence the recommendation performance, such as the weights in joint learning?

6.1 Experimental Settings

6.1.1 Dataset. We choose the dataset from a real-world legal information system, in which the majority of users are lawyers and judges. Users are first sampled and then filtered by the constraint of no less than three interactions. In total, we retain 3,327 users and collect their last month's interactions and search histories during January 2019 to April 2020. The statistics of the dataset are listed in Table 2. To the best of our knowledge, it is the largest legal recommendation dataset.

Consistent with the experimental settings in the literature [5, 37], we further split the dataset by leave-one-out strategy. Specifically, we use each user's most recent interaction for testing, the second recent one for validation, and the previous interactions and search histories for training. As we do not focus on the repeat recommendation problem, the duplicated leave-out test items are removed from the user's interactions and search histories. A negative sampling method is used for evaluating the recommendation performance in the ranking task. For the one ground-truth item of each user, we randomly sample 99 non-interacted items.

We build the legal knowledge graph from a large-scale knowledge graph manually annotated in a commercial legal platform. The types of selected legal concepts include cause, law, clause, and factor. We use all cause and factor nodes and reserve law nodes that connect at least three cases in our dataset as well as all its clause. All relations between these selected nodes are retrained. The language of our dataset is Chinese, and detailed descriptions and statistics of constructed HIN are listed in Table 3. It is worth noting that only forward relations are shown in Table 2 and Table 3, and both forward and reverse relations are considered in our model. Since this dataset is related to commercial systems and concerning the user privacy issue, it cannot be fully released.

6.1.2 Evaluation Protocols. We use the evaluation metrics ***Hit Ratio*** **(HR)** and ***Normalized Discounted Cumulative Gain*** **(NDCG)** [17] to evaluate the performance of recommendation methods. HR@k concerns whether the ground-truth item appears in the Top-K recommendation list, and NDCG@k measures the ground-truth item's position in the Top-K recommendation list. Formally, we use $g_u \in [1, 100]$ to denote the rank of the ground-truth item in the recommendation

Table 3. Vertexes and Relations of HIN

No.	Vertex	Number	Description
1	User	3,327	User in legal information system.
2	Case	343,245	Case documents in legal information system.
3	Query	28,593	Search words.
4	Cause	2,214	Cause of action, the grounds (as violation of a right) that entitle a plaintiff to bring a suit.
5	Law	1,241	A legal text enacted by the legislature.
6	Clause	53,497	A separate section of a law.
7	Factor	4,044	Critical circumstances of a case.
No.	**Relation**	**Number**	**Description**
1	User → Query	31,165	A user and the query submitted.
2	Query → Case	500,074	A query and the item in its search result.
3	Case → Cause	352,256	A case cites a cause.
4	Case → Law	816,078	A case cites a law.
5	Case → Clause	1,853,558	A case cites a clause.
6	Cause → Clause	3,735	A cause is related to a clause.
7	Law → Clause	53,497	A law contains a clause.
8	User → Case	57,988	A user clicks a case.
9	Case → Factor	1,657,627	A case contains a factor.
10	Cause → Factor	2,001	A cause is related to a factor.
11	Clause → Factor	919	A clause is related to a factor.
12	Cause → Cause	1,347	A broad cause of action contains a precise cause of action.
13	Query → Cause	13,146	A query contains a cause in its text.
14	Query → Factor	11,544	A query contains a factor in its text.

list for user u, then HR@k and NDCG@k are defined as follows in our experiments:

$$\begin{aligned} \text{HR@k} &= \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} I(g_u \leq k), \\ \text{NDCG@k} &= \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} \frac{I(g_u \leq k)}{\log_2(g_u+1)}, \end{aligned} \tag{27}$$

where $I(\cdot)$ indicates whether the condition is true, and returns 1 if the condition is true, otherwise 0. The average of five repeated experiments is reported as the final performance, where each experiment has the same parameter settings except different random seeds.

6.1.3 Baseline Methods. To evaluate the performance of the LegalGNN model, we compare it with various baselines. These methods include traditional factorization-based methods (BPR), GNN-based recommendation models (KGAT, LightGCN), HIN-based methods (CFKG, GATNE-T, GATNE-I), and content-based recommendation models (ACCM, NFM, NRMS, NRHUB).

- **BPR** [30]. This method applies the Bayesian Personalized Ranking objective function to optimize the Matrix Factorization model.
- **LightGCN** [16]. LightGCN is a simplified GNN-based model, where the user and item embedding is aggregated by the average of neighbors' embedding in each GNN layer.

Table 4. Comparison of Baseline Methods and LegalGNN

Characteristics	Interactions	KG	Content	Relation	Query
BPR	✓				
LightGCN	✓				
KGAT	✓	✓		✓	
CFKG	✓	✓		✓	
GATNE-T	✓	✓		✓	✓
GATNE-I	✓	✓	✓	✓	✓
ACCM	✓		✓		
NFM	✓		✓		
NRMS	✓		✓		
NRHUB	✓		✓		✓
LegalGNN	✓	✓	✓	✓	✓

- **KGAT** [45]. KGAT is a GNN-based method that applies TranR to obtain the initial node embeddings and learns the user and item embeddings by multi-layer attention-based aggregation.
- **CFKG** [58]. The model learns the user and item embedding by applying TransE.
- **GATNE-T** [4]. This model is a graph embedding method for modeling attributed multiplex heterogeneous networks. For each node, GATNE-T leverages separate GCNs to learn multiple node embeddings according to different types of relations. The base embedding and initial edge embedding are directly trained based on the network structure.
- **GATNE-I** [4]. Different from GATNE-T, for each node, GATNE-I trains transformation functions that transform the raw node feature to get node embedding and edge embedding. So, GATNE-T uses the structural information while GATNE-I uses both the structural and content information.
- **ACCM** [34]. Attentional Content&Collaborate Model applies an attention mechanism to weigh the content part and collaborative filtering part. A cold-sampling strategy is proposed to handle the cold-data problem.
- **NFM** [15]. Neural Factorization Machine is a content-based method, which uses multi-layer perceptions under feature bi-interaction.
- **NRMS** [50]. NRMS is a neural news recommendation method that uses multi-head self-attention for news and user modeling.
- **NRHUB** [49]. NRHUB is a neural news recommendation model exploiting users' click, search, and browser behavior. The model applies the word-level and record-level attention to select informative words and behavior after encoding text with CNN.

We compare our LegalGNN model with these baseline methods in terms of the types of information included, as shown in Table 4. Traditional factorization-based method BPR only uses the historical interactions, and KGAT and CFKG further incorporate the Knowledge Graph. ACCM, NFM, NRMS, and NRHUB models use more content information but no relations with neighbors. LegalGNN takes all this information to learn the preference of legal users and the items.

6.1.4 Implementation Details. We implement our model with *PyTorch* and the implementation codes are publicly available,[3] which is based on the recommendation framework ReChrous[4] [40].

[3] https://github.com/yangjun12/LegalGNN.
[4] https://github.com/THUIR/ReChorus.

We use the authors' codes for KGAT,[5] LightGCN,[6] GATNE-T, and GATNE-I.[7] Besides, we use recommendation framework ReChrous[8] [40] to run BPR, NFM, and CFKG. The other methods are implemented with *PyTorch*.

Adam optimizer [19] is adopted as the optimizer, which is effective in neural model training. We adopt the early stop strategy and the stop triggered if the validation performance does not improve for 10 epochs. The length of the input text is truncated to 512 due to the restrain of BERT. For a fair comparison, we use the same text as the input of NRHUB, NRMS, ACCM. The input token length of NRHUB, NRMS, and ACCM is assigned as 256, which covers 99.99% of texts. We regard the directly connected entities as items' feature in NFM. The walk length is 5 and the number of walk heads is 3 in GATNE-T and GATNE-I. The embedding size is 64. GATNE-I and our model have the same content features to achieve a fair comparison. We follow the meta-path {(User, Case, User), (Case, User, Case), (User, Query, User), (Query, User, Query), (Case, Query, Case), (Query, Case, Query), (Case, Cause, Case), (Cause, Case, Cause), (Case, Law, Case), (Law, Case, Law), (Case, Clause, Case), (Clause, Case, Clause), (Case, Factor, Case), (Factor, Case, Factor)} to start random-walk in GATNE-T and GATNE-I. In our method, the batch size is set to 128, and the embedding size is set to 64. Content length is 16 based on pilot experiments. The hyper-parameters are tuned according to the performance in the validation set. Considering the efficiency, we use a two-layer GNN. Considering the high scalability of our method due to the usage of the neighbor sampling, we believe that stacking GNN layers will lead to improved results. In our methods, the weight parameter α in joint learning is tuned within $[0.01, 0.1, 1, 10, 100]$, the neighbor sample number is tuned within [4, 8, 16, 32, 64], the content feature length is tuned within [0, 16, 32, 48, 64], while the total length of the content feature and structural feature remains at 64. Our best performance achieved when the weight parameter α is 0.1, the content feature length is 16, the learning rate is 10^{-3}, and $l2$-normalization coefficients λ is 10^{-5}. All the parameters are normally initialized with 0 mean and 0.01 standard deviation.

The embedding size and batch size setting of baseline methods follow the description of their papers. The hyper-parameters of all baseline methods are carefully tuned to achieve optimal performances. For all methods, The learning rates β are tuned within $[0.01, 0.001, 0.0001]$. The $l2$-normalization coefficients λ are tuned within [10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 0]. The experiments are conducted with a single GTX-1080 Ti GPU.

6.2 Overall Performance

We first evaluate the overall performance of all methods, as shown in Table 5. Generally, our proposed LegalGNN significantly outperforms all the baseline methods, demonstrating its effectiveness.

Among all of the baselines, the text-based method NRMS achieves the highest performance, indicating the content feature is useful in our task. Besides, compared with ACCM, NRHUB achieves better performance. The main reason is that NRHUB additionally considers the query as the user's context information, which proves the effectiveness of incorporating queries. Surprisingly, the performance of KGAT is worse than CFKG. One possible reason is that KGAT may be misled by the leakage path bias in such a sparse dataset. HIN-based method CFKG outperforms GATNE-T and GATNE-I. The possible reason could be that GATNE-T and GATNE-I are optimized by skip-gram, in which context nodes contribute equally to the target nodes, while in CFKG, the contributions

[5]https://github.com/xiangwang1223/knowledge_graph_attention_network.
[6]https://github.com/kuandeng/LightGCN.
[7]https://github.com/THUDM/GATNE.
[8]https://github.com/THUIR/ReChorus.

Table 5. Performance of Different Models

Model	HR@5	HR@10	HR@20	NDCG@5	NDCG@10	NDCG@20	Improv.
BPR	0.0890	0.1097	0.1707	0.0724	0.0791	0.0941	+542.7%
LightGCN	0.0712	0.1127	0.2079	0.0540	0.0671	0.0909	+620.5%
KGAT	0.1638	0.2822	0.4562	0.1047	0.1424	0.1862	+234.5%
CFKG	0.3381	0.4476	0.5642	0.2450	0.2801	0.3094	+80.5%
GATNE-T	0.2912	0.3763	0.5145	0.2109	0.2384	0.2732	+108.4%
GATNE-I	0.3095	0.4250	0.5746	0.2176	0.2550	0.2926	+93.8%
ACCM	0.2649	0.4057	0.5790	0.1782	0.2235	0.2671	+117.7%
NFM	0.4001	0.5122	0.6354	0.2966	0.3327	0.3638	+53.7%
NRMS	<u>0.4725</u>	<u>0.5917</u>	<u>0.7147</u>	<u>0.3572</u>	<u>0.3957</u>	<u>0.4268</u>	+31.0%
NRHUB	0.3883	0.5424	0.6969	0.2600	0.3099	0.3490	+58.6%
LegalGNN	**0.6280****	**0.7161****	**0.8023****	**0.5228****	**0.5513****	**0.5731****	-

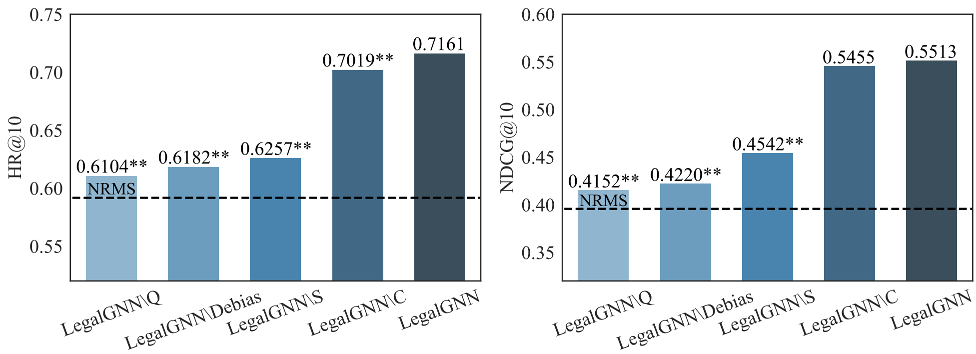
The Best performing method is boldfaced, and the second best method is underlined. ** means significantly better than the strongest baseline ($p < 0.01$). "Improv." means the relative improvement of the LegalGNN model over corresponding baseline (averaged across all metrics).

of neighbor nodes are dynamically learned. In the model that uses content information, the performance of GATNE-I is weaker than NFM, NRMS, and NRHUB. The possible reason could be GATNE-I learns the embeddings of users on sequences produced by meta-paths (User, Item, User) and (User, Query, User), which are sparse in our dataset, and then restrict the performance of the model. Finally, performances of BPR and LightGCN are lowest among all baselines, since they only use the user-item interaction.

Our method, LegalGNN, is significantly superior over the best baseline for a large margin on all adopted evaluation metrics. We summarize three reasons: (1) Integrating the content and structural feature into a unified representation. Both methods use the content feature (e.g., NRMS, NRHUB) and using structural connections (e.g., CFKG) achieves notable performances. However, our model achieves better performance by integrating the two aspect information in the unified representation module, contrasting to these methods that lack an effective way to integrate content features and structural connections. (2) Incorporating query and legal knowledge graph as interconnected nodes in HLIN. On the one hand, the introduced HLIN enriches the connectivity of new items, which alleviates the new item problem. Compared with LightGCN and KGAT, part of the performance rise comes from enriched connectivity between users and items. On the other hand, we introduce both textual and structural information of queries to model users' precise information needs. Our model is superior to NRHUB by considering the structural connections of queries, which absorb information from connected users, items, and entities. (3) Adopting the graph neural network with the debias strategy. GNN is employed to capture high-order connections in HLIN in our work, which is superior to CFKG, which only considers the first-order connection. Besides, we reveal the leakage path bias in the graph-enhanced recommendation. As elaborated in Section 5, the bias between ground truth items and sampled negative items misleads the model into capturing the wrong signal, which leads to the declined performance of KGAT.

6.3 Ablation Study

To verify the effectiveness of our model's main components, we investigated the performance under four variants of our model. Some modifications of our LegalGNN model are listed as follows:



(a) HR@10 (b) NDCG@10

Fig. 4. Results of ablation study evaluating the usefulness of main components. The paired t-test is performed between each variant and the original LegalGNN, labeled by * ($p < 0.05$) and ** ($p < 0.01$). The performance of the best-performing baseline, NRMS, is shown as a line for comparison.

- **LegalGNN\C.** This model removes the content feature. The length of the content feature is 0, and the node embedding is randomly initialed and constrained by transR only.
- **LegalGNN\S.** This model removes the structural feature. The length of the structural feature is 0.
- **LegalGNN\Debias.** This model removes the debias strategy in training.
- **LegalGNN\Q.** This model removes the query node and all its related relations on the HLIN.

Figure 4 shows HR@10 and NDCG@10 performance of these LegalGNN variants. From the figure, we have the following observations:

First, the ablations of the main components result in performance declining, verifying the effectiveness of these components. The largest decline is achieved by LegalGNN/Q, which shows the incorporation of query promotes the embedding learning of users and items. Besides, the performance also declines by a large margin in LegalGNN/Debias, which shows the serious impact of leakage path bias in the sparse dataset and indicates the effectiveness of our debias strategy. Second, LegalGNN\S is the traditional way of integrating content features and structural connections, which learns the node embeddings by propagating node features on the graph. The decline of LegalGNN\S verified propagating node features on the graph is insufficient. LegalGNN achieves better performance for employing the unified representation module, a proposed new method that effectively integrates content and structure features. Third, the performance of LegalGNN\C declines slightly but significantly, which shows content features cannot be ignored.

6.4 Performance in the Cold-start Scenario

We further compare our model with representative baselines on both the new item set and interacted item set. As shown in Table 6, our proposed LegalGNN significantly outperforms baseline methods on the new item set, demonstrating the effectiveness of our model in the cold-start scenario.

Table 6 shows HR@10 and NDCG@10 performance of these baselines. From the table, we have the following observation:

Furthermore, the improvement on new items is greater than that on warm items, illustrating the ability of our model to solve the cold-start problem. Comparing with the best baseline NRMS, our model achieves an 11.59% performance increase on warm items and a 33.92% increase on new items.

Table 6. Performance of Representative Models on New Item Set and Interacted Item Set

Model	New item			Interacted item		
	HR@10	NDCG@10	Improv.	HR@10	NDCG@10	Improv.
KGAT	0.2386	0.1161	+290.9%	0.4776	0.2910	+60.7%
GATNE-T	0.3803	0.2378	+111.8%	0.3504	0.2427	+104.6%
GATNE-I	0.4206	0.2531	+95.71%	0.4531	0.2667	+72.66%
NFM	0.4886	0.3186	+61.16%	0.5786	0.3976	+24.43%
NRMS	<u>0.5832</u>	<u>0.3861</u>	+33.92%	<u>0.6378</u>	<u>0.4482</u>	+11.59%
LegalGNN	**0.7202****	**0.5573****	-	**0.6953****	**0.5117****	-

The best-performing method is boldfaced, and the second-best method is underlined. ** means significantly better than the strongest baseline ($p < 0.01$).

This is because LegalGNN uses relation-aware GNN with the debias strategy on the constructed HLIN, so it can obtain better representations of items, especially new ones.

We also notice that our model performs better on new items than on warm items. We speculate that the reason derives from the ability of learning new item embedding and data imbalance. Our model fuses content feature and structural features, which are helpful for cold item, and we conduct GNN with debias strategy to learn a better representation of nodes. The majority of items (95.5%) in the training set are new items dominate items dominant the parameter learning process. Therefore, the model fits the new items better, but the interacted items may be underfitting in the dataset. Because the main focus of our model is not to deal with the data imbalance problem, and the unbalanced data is common in practice, we keep the original dataset for a fair comparison. To verify this speculation, we further conduct an experiment on a reconstructed training set, where the number of training pairs containing warm items is doubled. After training on new items and interacted items together in the new dataset, HR@10 of LegalGNN on the warm item set is promoted to 0.7188, while performance on the new items declines to 0.7044. And the overall HR@10 is 0.7063, which is lower than our original dataset. This verifies our conjecture that our model learns the embedding of new items better. The results also inspire us that in the future, according to the distribution of the dataset, training on new items or interacted items only can be considered if necessary.

6.5 Impacts of Incorporating Query

We conduct experiments to further explore the influence of incorporating query. Specifically, we want to investigate whether the user-query and item-query connection helpful in our task. Thus, we design the following LegalGNN variants:

- **LegalGNN\Q.** This variant removes query nodes and all the linked edges of query nodes in HLIN.
- **LegalGNN\Q-I.** This variant removes only the query-item edges in HLIN. Query nodes aggregate information from linked users and entities.
- **LegalGNN\Q-U.** This variant removes the query-user edges in HLIN. Query nodes aggregate information from linked items and entities.
- **LegalGNN\Q-E.** This variant removes query-entity edges in HLIN. Query nodes aggregate information from linked users and items.

Figure 5 shows the performances of these LegalGNN variants, measured by the HR@10 and NDCG@10. First, both the query-item and query-user connections are beneficial for the recommendation task. Compared to LegalGNN\Q that removes the query node and all the types

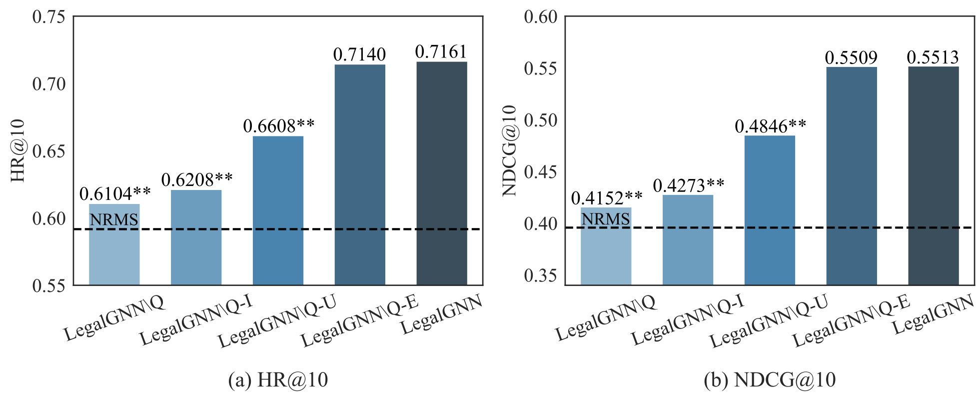


Fig. 5. Results of ablation study evaluating the impacts of different types of query-related information. The paired t-test is performed between each variant and the original LegalGNN, labeled by * ($p < 0.05$) and ** ($p < 0.01$). The performance of the best-performing baseline, NRMS, is shown as a line for comparison.

of related edges, LegalGNN\Q-I and LegalGNN\Q-U have better performance, indicating the usefulness of lateral information obtained from the queries. Second, LegalGNN\Q-U is superior to LegalGNN\Q-I by a large margin. A potential reason is that query has more connections to item nodes. Third, LegalGNN is superior to both LegalGNN\Q-I and LegalGNN\Q-U, mainly because of the extra legal information brought by the query. Specifically, representation of the query is strength aggregating information from connected neighbors. As the result, we get a better representation of users and items due to the aggregation of the information from connected queries. Note that LegalGNN/Q-E leads to a slight performance loss, which shows query mainly serves as the bridge between users and items.

6.6 Hyper-parameter Analyses

In this section, we conduct several experiments on the valid set to investigate the impact of the main hyper-parameters in our LegalGNN model, including the weight parameter α of joint learning and the neighbor sampling size.

6.6.1 The Hyperparameter of Multi-task Learning. The hyperparameter α controls the weight of the recommendation task and unified represent task by adjusting the rate of their loss. Figure 6(a) shows the trend of LegalGNN performance change with the multi-task learning hyperparameter α. From the figure, we have the following observations:

First, the restriction of TransR in the united representation module promotes our recommendation performance. Compared with the performance under α=0, we get better performance under an appropriate α=0.1. Second, the performance increases as α rises, then the performance decreases after the peak value. The overall trend reflects that the unified representation module benefits the recommendation task by modeling the relations of HLIN triples. However, a dominant weight of unified representation task results in performance loss.

6.6.2 Neighbor Sampling Size. Following the design of GraphSAGE [14], we sampled a fixed-size set of neighbors in each layer. To investigate the impact of neighbor sampling size, we observe the model performance and runtime in our experiments. Figure 6(b) shows the model performance with respect to the neighbor sampling size, where the result is under the setting of GNN Layer=2,

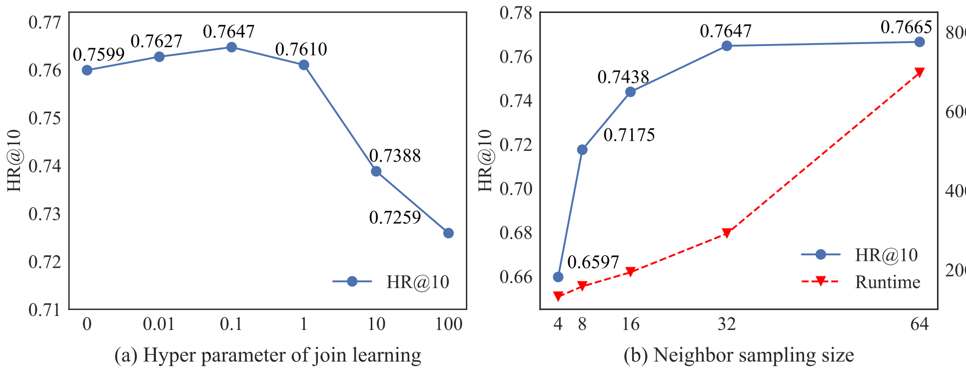


Fig. 6. Results of hyperparameter analysis on valid set. (a) the impacts of the weight α of joint learning. (b) the impacts of the size of neighbor sampling.

and the neighbor sampling size of all layers is equal. From the figure, we have the following observations:

The overall performance rises with the increase of the neighbor sampling size, which benefits from aggregating more neighborhood information. Besides, the runtimes (second) of one epoch under different neighbor sampling sizes are reported. As the figure shows, the diminishing of the neighbor sample size significantly reduces the runtime but still maintains a strong performance. For the sack of efficiency, we adopt the neighbor sampling size 32 in our experiments.

6.7 Case Study

To understand how integrating content and structural features benefit the legal recommendation, we show the inference process behind our model's recommendation through a sampled case (user u, item v_{57}).

User u is a lawyer dealing with an inheritance case related to a *pre-marital house demolition*. To collect related legal information, the user clicks two items v_{86} and v_{95}. As Figure 7 shows, there exist two paths from user u to the target item v_{57}: (1) through the structural connections among legal concepts. A key element e_{489} is linked to the user u through the previous interactions with item v_{95} and is also related to cause of action e_{27} and hence its linked item v_{57}. $[u \rightarrow v_{95} \rightarrow e_{489} \rightarrow e_{27} \rightarrow v_{57}]$. (2) through the similarity of content, v_{57} is of high similarity with user u's search history $[u \rightarrow q_1, q_2 \rightarrow v_{57}]$.

Compared with the item v_2 that only has the content similarity with query q_1 and v_{223} that uses only the structural connections, the target item v_{57} achieves a higher score due to the integration of both information. The case study shows the importance of content and structural information in the Legal-Rec scenario and also demonstrates the rationality and effectiveness of the representation module in the LegalGNN model.

7 CONCLUSION

In this work, we propose a legal information enhanced graph neural network for the recommendation task in the legal scenario (Legal-Rec). Through revealing the differences between Legal-Rec and other recommendation scenarios (e.g., news recommendation, E-commerce), we summarize three main challenges: (1) Integrating textual and structural legal information; (2) New item problem; (3) Modeling the interests of expert users. To tackle these challenges, we propose Legal information enhanced Graph Neural Network (LegalGNN). First, we design a unified

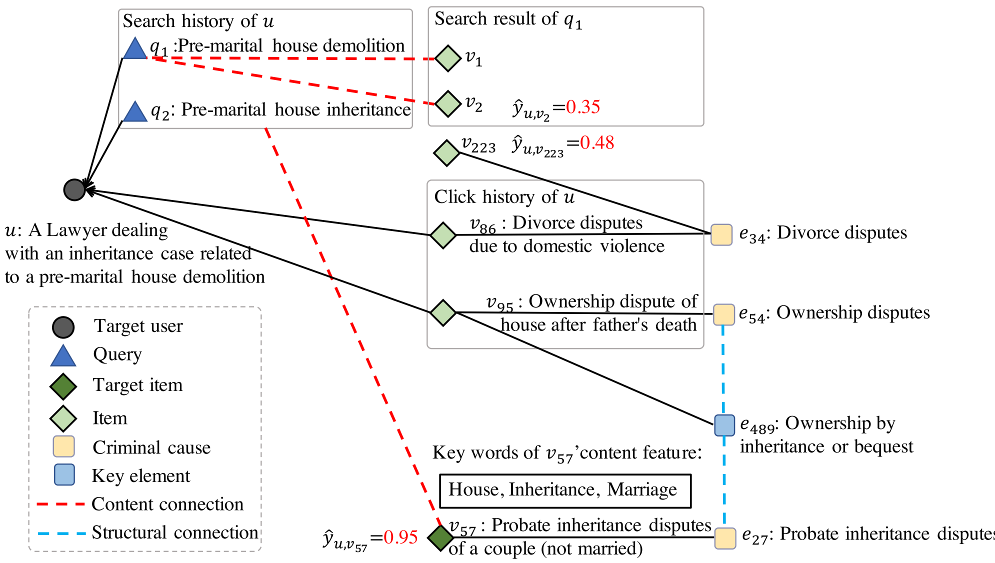


Fig. 7. A real example of LegalGNN. User u clicks item v_{95}, which connects to *key element* e_{489}, and e_{489} expands to *criminal cause* e_{27}, then the target item v_{57} is recommended by expanding from e_{27}. Besides, content feature of v_5 has high similarity with the content feature in the search history of u, then the target item v_{57} is recommended due to the content similarity.

representation module to fuse content and structural feature into the unified embedding, which provides an informative representation of legal information. Second, we introduce users' active information requirement to alleviate the new item problem at the scenario level. Specifically, we incorporate the query node into the heterogeneous information network (HIN), which mitigates the effect of the new item problem by enriching the connection between item nodes. The query node also provides extra information for user modeling. Finally, a light multi-relational GNN is proposed to capture the message for the recommendation. To the best of our knowledge, this is the first attempt that adopts neural graph methods in legal information recommendation. Besides, we summarize the bias problem of graph-based recommendation method and give a rational solution. After conducting experience on a real-world legal recommendation dataset, our model achieves remarkable performances over state-of-the-art models by a large margin.

In the future, we plan to consider users' short interests to achieve better legal information recommendation performances and to adopt this model in real systems. Besides benefiting from the unified representation of nodes in HIN, our methods can also be applied to other recommendation scenarios. So, we also would like to apply our model to other scenarios, which may yield more convincing experiment results.

REFERENCES

[1] Sheng Bi, Yanhui Huang, Xiya Cheng, Meng Wang, and Guilin Qi. 2019. Building Chinese legal hybrid knowledge network. In *Knowledge Science, Engineering and Management - 12th International Conference, KSEM 2019, Athens, Greece, August 28-30, 2019, Proceedings, Part I (Lecture Notes in Computer Science)*, Christos Douligeris, Dimitris Karagiannis, and Dimitris Apostolou (Eds.), Vol. 11775. Springer, 628–639. DOI: https://doi.org/10.1007/978-3-030-29551-6_56

[2] Alexander Boer and Radboud Winkels. 2016. Making a cold start in legal recommendation: An experiment. In *Legal Knowledge and Information Systems - JURIX 2016: The Twenty-Ninth Annual Conference (Frontiers in Artificial Intelligence and Applications)*, Floris Bex and Serena Villata (Eds.), Vol. 294. IOS Press, 131–136. DOI: https://doi.org/10.3233/978-1-61499-726-9-131

[3] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5–8, 2013, Lake Tahoe, Nevada, United States*, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 2787–2795. Retrieved from https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html.

[4] Yukuo Cen, Xu Zou, Jianwei Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Representation learning for attributed multiplex heterogeneous network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4–8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 1358–1368. DOI: https://doi.org/10.1145/3292500.3330964

[5] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7–11, 2017*, Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White (Eds.). ACM, 335–344. DOI: https://doi.org/10.1145/3077136.3080797

[6] Lei Chen, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. 2020. Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. In *the Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020.* AAAI Press, 27–34. Retrieved from https://aaai.org/ojs/index.php/AAAI/article/view/5330.

[7] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS@RecSys 2016, Boston, MA, USA, September 15, 2016*, Alexandros Karatzoglou, Balázs Hidasi, Domonkos Tikk, Oren Sar Shalom, Haggai Roitman, Bracha Shapira, and Lior Rokach (Eds.). ACM, 7–10. DOI: https://doi.org/10.1145/2988450.2988454

[8] Ademar Crotti Junior, Fabrizio Orlandi, Declan O'Sullivan, Christian Dirschl, and Quentin Reul. 2019. Using mapping languages for building legal knowledge graphs from XML files. http://arxiv.org/abs/1911.07673

[9] Zhiyong Cui, Kristian Henrickson, Ruimin Ke, and Yinhai Wang. 2020. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *IEEE Trans. Intell. Transport. Syst.* 21, 11 (2020), 4883–4894. DOI: 10.1109/TITS.2019.2950416

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. DOI: https://doi.org/10.18653/v1/n19-1423

[11] Erwin Filtz. 2017. Building and processing a knowledge-graph for legal data. In *European Semantic Web Conference*. Springer, 184–194.

[12] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

[13] Zichen Guo, Tieke He, Zemin Qin, Zicong Xie, and Jia Liu. 2019. A content-based recommendation framework for judicial cases. In *International Conference of Pioneering Computer Scientists, Engineers and Educators*. Springer, 76–88.

[14] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4–9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 1024–1034. Retrieved from https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html.

[15] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7–11, 2017*, Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White (Eds.). ACM, 355–364. DOI: https://doi.org/10.1145/3077136.3080777

[16] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and powering graph convolution network for recommendation. DOI: https://doi.org/10.1145/3397271.3401063

[17] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* 20, 4 (2002), 422–446.

[18] Zhuoren Jiang, Yue Yin, Liangcai Gao, Yao Lu, and Xiaozhong Liu. 2018. Cross-language citation recommendation via hierarchical representation learning on heterogeneous graph. In *the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval.* ACM, 635–644.

[19] 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.). http://arxiv.org/abs/1412.6980

[20] Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[21] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. *arXiv preprint arXiv:1801.07606* (2018).

[22] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *the 29th AAAI Conference on Artificial Intelligence*.

[23] Qiang Lu and Jack G. Conrad. 2012. Bringing order to legal documents - an issue-based recommendation system via cluster association. In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development, Barcelona (KEOD'12)*, Joaquim Filipe and Jan L. G. Dietz (Eds.). SciTePres, 76–88.

[24] Weizhi Ma, Min Zhang, Chenyang Wang, Cheng Luo, Yiqun Liu, and Shaoping Ma. 2018. Your tweets reveal what you like: Introducing cross-media content information into multi-domain recommendation. In *the International Joint Conference on Artificial Intelligence*. 3484–3490.

[25] Federico Monti, Michael Bronstein, and Xavier Bresson. 2017. Geometric matrix completion with recurrent multi-graph neural networks. In *the International Conference on Advances in Neural Information Processing Systems*. 3697–3707.

[26] Hoang NT and Takanori Maehara. 2019. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550* (2019).

[27] Afshin Rahimi, Trevor Cohn, and Timothy Baldwin. 2018. Semi-supervised user geolocation via graph convolutional networks. *arXiv preprint arXiv:1804.08049* (2018).

[28] Yongli Ren, Martin Tomko, Flora Dilys Salim, Jeffrey Chan, Charles L. A. Clarke, and Mark Sanderson. 2017. A location-query-browse graph for contextual recommendation. *IEEE Trans. Knowl. Data Eng.* 30, 2 (2017), 204–218.

[29] Steffen Rendle. 2010. Factorization machines. In *the IEEE International Conference on Data Mining*. IEEE, 995–1000.

[30] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 452–461.

[31] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *the European Semantic Web Conference*. Springer, 593–607.

[32] Yunqiu Shao, Jiaxin Mao, Yiqun Liu, Weizhi Ma, Ken Satoh, Min Zhang, and Shaoping Ma. BERT-PLI: Modeling paragraph-level interactions for legal case retrieval. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI'20)*, Christian Bessiere (Ed.). 3501–3507. DOI: 10.24963/ijcai.2020/484

[33] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and S. Yu Philip. 2018. Heterogeneous information network embedding for recommendation. *IEEE Trans. Knowl. Data Eng.* 31, 2 (2018), 357–370.

[34] Shaoyun Shi, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Attention-based adaptive model to unify warm and cold starts recommendation. In *the 27th ACM International Conference on Information and Knowledge Management*. 127–136.

[35] Erwin van den Berg, Bachelor Opleiding Kunstmatige Intelligentie, and AWF Boer. 2015. *Development of a Recommender System for Dutch Case Law, with the Use of a Topic Model.* Bachelor Thesis. Universiteit van Amsterdam.

[36] Rianne van den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *CoRR* abs/1706.02263 (2017).

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *the International Conference on Advances in Neural Information Processing Systems*. 5998–6008.

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

[39] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.* 11, 12 (2010).

[40] Chenyang Wang, Min Zhang, Weizhi Ma, Yiqun Liu, and Shaoping Ma. 2020. Make it a chorus: Knowledge-and time-aware item modeling for sequential recommendation. In *the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 109–118.

[41] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *the 27th ACM International Conference on Information and Knowledge Management*. 417–426.

[42] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep knowledge-aware network for news recommendation. In *the World Wide Web Conference*. 1835–1844.
[43] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. 2019. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 968–977.
[44] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *the International Workshop on Data Mining for Online Advertising*. 1–7.
[45] 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 & Data Mining, KDD 2019, Anchorage, AK, USA, August 4–8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 950–958. DOI: https://doi.org/10.1145/3292500.3330989
[46] Xiao Wang, Meiqi Zhu, Deyu Bo, Peng Cui, Chuan Shi, and Jian Pei. 2020. AM-GCN: Adaptive multi-channel graph convolutional networks. In *KDD'20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23–27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1243–1253. DOI: https://doi.org/10.1145/3394486.3403177
[47] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27–31, 2014, Québec City, Québec, Canada*, Carla E. Brodley and Peter Stone (Eds.). AAAI Press, 1112–1119. Retrieved from http://www.aaai.org/ocs/index.php/AAAI/AAAI14/paper/view/8531.
[48] Radboud Winkels, Alexander Boer, Bart Vredebregt, and Alexander van Someren. 2014. Towards a legal recommender system. In *the International Conference on Legal Knowledge and Information Systems*. 169–178.
[49] Chuhan Wu, Fangzhao Wu, Mingxiao An, Tao Qi, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with heterogeneous user behavior. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3–7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 4873–4882. DOI: https://doi.org/10.18653/v1/D19-1493
[50] Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with multi-head self-attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 6390–6395.
[51] Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. 2019. Simplifying graph convolutional networks. *arXiv preprint arXiv:1902.07153* (2019).
[52] Yuexin Wu, Hanxiao Liu, and Yiming Yang. 2018. Graph convolutional matrix completion for bipartite edge prediction. In *the International Conference on Knowledge Discovery and Information Retrieval*. 49–58.
[53] Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Simple applications of BERT for ad hoc document retrieval. *CoRR* abs/1903.10972 (2019).
[54] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. 2018. Hierarchical graph representation learning with differentiable pooling. In *the International Conference on Advances in Neural Information Processing Systems*. 4800–4810.
[55] Feng Yu, Zhaocheng Liu, Qiang Liu, Haoli Zhang, Shu Wu, and Liang Wang. 2020. Deep interaction machine: A simple but effective model for high-order feature interactions. In *the 29th ACM International Conference on Information & Knowledge Management*. 2285–2288.
[56] Jiani Zhang, Xingjian Shi, Shenglin Zhao, and Irwin King. 2019. STAR-GCN: Stacked and reconstructed graph convolutional networks for recommender systems. DOI: https://doi.org/10.24963/ijcai.2019/592
[57] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating text generation with BERT. Retrieved from https://openreview.net/forum?id=SkeHuCVFDr.
[58] Yongfeng Zhang, Qingyao Ai, Xu Chen, and Pengfei Wang. 2018. Learning over knowledge-base embeddings for recommendation. *arXiv preprint arXiv:1803.06540* (2018).
[59] Haoxi Zhong, Zhengyan Zhang, Zhiyuan Liu, and Maosong Sun. 2019. *Open Chinese Language Pre-trained Model Zoo.* Technical Report. NLP Lab, Department of Computer Science, Tsinghua University.
[60] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2018. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434* (2018).
[61] Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2020. Incorporating BERT into neural machine translation. *arXiv preprint arXiv:2002.06823* (2020).

Received November 2020; revised April 2021; accepted June 2021