Knowledge-Based Systems 327 (2025) 114083

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

LF-HGRILF: A law-fact heterogeneous graph representation and iterative learning framework for legal judgment prediction

Yinying Kong [a,1], You-Gan Wang [a,1,*], Haodong Deng [a], Zhanhao Xiao [b,c], Yuke Zhang [a]

[a] School of Statistics and Data Science, Guangdong University of Finance & Economics, Guangdong, 510320, China
[b] School of Computer Science, Guangdong Polytechnic Normal University, Guangdong, 510665, China
[c] Guangdong Provincial Key Laboratory of Intellectual Property and Big Data, Guangdong Polytechnic Normal University, Guangdong, 510665, China

ARTICLE INFO

Keywords:
Legal judgment prediction
Heterogeneous networks
Confusing law articles distinction

ABSTRACT

Legal Judgment Prediction (LJP) is a key aspect of legal intelligence, leveraging machine learning and natural language processing to analyze legal facts and predict relevant law articles, charges, and penalties. Most existing methods, however, focus solely on semantic relationships between individual cases and law articles, neglecting the topological structure connecting them. This oversight results in misjudgments, particularly in cases with overlapping legal grounds due to law article similarities. To address these limitations, we propose the Law-Fact Heterogeneous Graph (LF-HetG) and its corresponding iterative learning framework (LF-HGRILF). Unlike prior approaches, LF-HetG integrates four types of nodes-case facts, factual sentences, keywords, and law articles-capturing both the connections between case facts and law articles and enabling the establishment of semantically similar relationships through common keywords. LF-HGRILF addresses the shortcomings of existing models by incorporating legal domain knowledge, case associations, and word-level semantic information. The framework learns deep case embeddings through neighbor aggregation in an iterative process and introduces a law article distinction module (LADM) to enhance the distinctiveness of similar law articles' embeddings. Finally, we use different classifiers to yield prediction outcomes for the three LJP tasks.Experimental results demonstrate that LF-HGRILF significantly outperforms existing methods across all predictive tasks, highlighting its effectiveness in improving legal judgment prediction.

1. Introduction

In the realm of artificial intelligence (AI), the objective of Legal Judgment Prediction (LJP) is to learn the reasoning patterns behind legal decisions by analyzing extensive legal texts and to predict case outcomes, such as applicable law articles, penalty terms, and criminal charges. Before the emergence of intelligent judicial systems, legal decisions are often influenced by subjective judgments, particularly in cases with insufficient evidence or ambiguous interpretations. AI-driven legal judgment systems offer significant advantages: they enhance judicial efficiency, reduce human biases, promote fair and consistent rulings, aid legal education and research, optimize judicial resource allocation, and improve transparency. Additionally, AI-assisted legal consultations provide valuable support for individuals without professional legal knowledge. Given these benefits, applying AI technology in the legal field has attracted substantial academic attention.

Legal judgment prediction contains a series of fundamental tasks in AI-driven judicial models, aiming to predict case outcomes based on factual descriptions. It comprises three key sub-tasks: law article prediction, charge prediction, and penalty term prediction. These sub-tasks are highly interdependent, as law articles, charges, and penalty terms are closely correlated. For instance, the same charge may lead to different penalty terms depending on the severity of the offense. These complex relationships make LJP particularly challenging, requiring models to effectively capture both semantic and structural dependencies among legal elements. Traditional LJP methods primarily rely on basic feature extraction and pattern recognition techniques, which struggle to capture the deep-seated relationships between cases and fail to model their intricate connections effectively. To address these limitations more effectively, researchers have introduced graph-based approaches to establish topological associations between cases. However, these methods still face significant challenges, including the cold start problem for new case nodes and insufficient granularity in node representations. In

* Corresponding author.
E-mail addresses: kongyy@gdufe.edu.cn (Y. Kong), ygwanguq2012@gmail.com (Y. Wang), 827358779@qq.com (H. Deng), xiaozhanhao@gpnu.edu.cn (Z. Xiao), 1191074012@qq.com (Y. Zhang).
[1] These authors contributed to the work equally and should be regarded as co-first authors.

https://doi.org/10.1016/j.knosys.2025.114083
Received 14 March 2025; Received in revised form 3 July 2025; Accepted 10 July 2025
Available online 13 July 2025

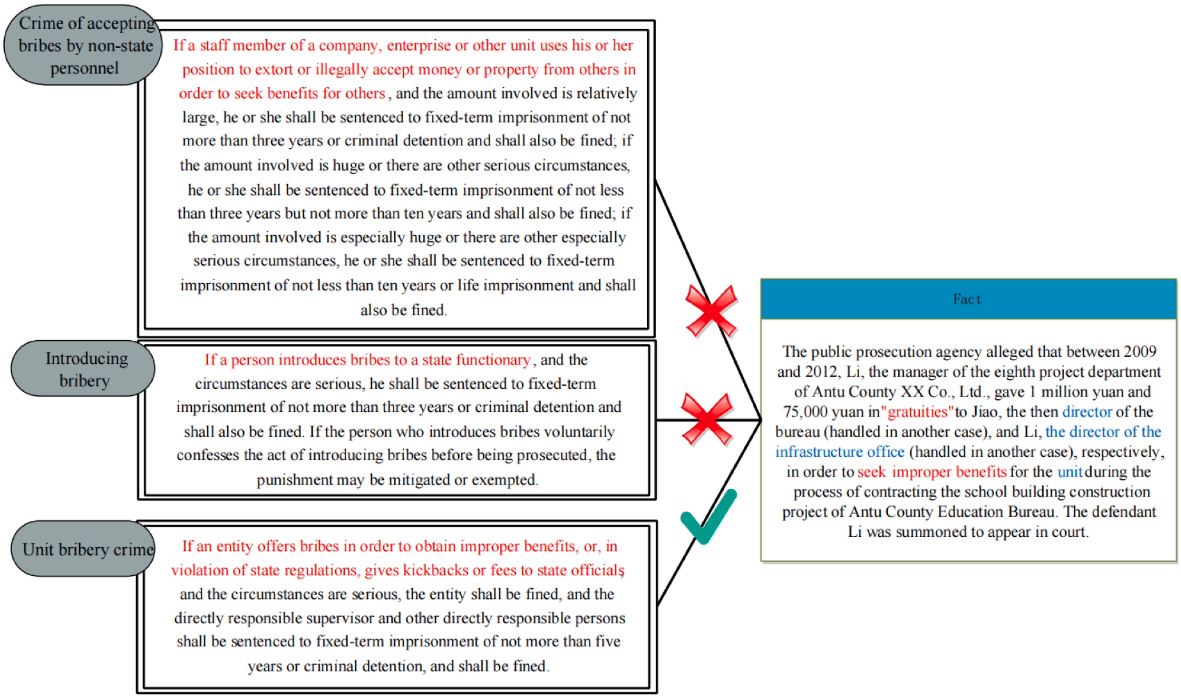


Fig. 1. An example of law articles confusion where the red font is common legal information and the blue font is differentiated information.

particular, determining whether a case violates specific law articles often depends on a few key terms, making it crucial to capture fine-grained lexical information. As illustrated in Fig. 1, identifying the law articles violated by a case often depends on specific terms, such as “director" or “manager". Therefore, accurately capturing a case’s key features requires incorporating fine-grained lexical information.

Another important challenge in LJP lies in the high similarity among law articles, which can easily lead to model misjudgments. As shown in Fig. 1, three different law articles regulate bribery but vary in terms of the subject and object of the criminal act. For instance, some articles specifically target professional groups such as “directors” or “chiefs”, while others focus on different criminal entities like “state functionaries” or “units”. This figure highlights the subtle differences between law articles, which can easily lead to confusion in legal judgment prediction. Specifically, the fact in Fig. 1 belongs to Unit bribery crime, yet it may be misjudged as Crime of accepting bribes by non-state personnel or Introducing bribery. To improve the effectiveness of LJP, it is essential to accurately distinguish these subtle differences between law articles. Hu et al. [1] introduced ten distinctive features to distinguish between similar charges. However, this method heavily relies on legal experts to define and use these features, while also requiring constant monitoring and updates to the model. This dependency limits scalability, making it impractical for large-scale legal applications, particularly when processing numerous cases or large datasets. The scarcity of expert resources further increases the cost and time required for legal analysis, restricting access to rapid and cost-effective legal services. Similarly, Xu et al. [2] constructed law article communities and generated law article community embeddings using a legal refinement module, which were then combined with reconstructed case embeddings for prediction. However, by removing similar features between nodes, this approach risks losing essential semantic information from law articles.

Furthermore, the limitations of current methods are also evident in how they model case-law relationships over time. Most existing methods fail to establish deep relationships between cases and law articles, except for L-HKN [3] and L-HetG [4]. However, these frameworks adopt the one-shot strategy, where neighbors are sampled once and the graph is generated only once. In each training epoch, the node embeddings need to be reinitialized, preventing the model from incorporating feedback from previous epochs. As a result, it struggles to fully capture and refine the intricate relationships between case facts and law articles.

In this paper, to systematically address the major limitations identified in the existing LJP approaches, we propose a novel law-fact heterogeneous graph (LF-HetG) with a novel learning framework, Law-Fact Heterogeneous Graph Representation Iterative Learning Framework (LF-HGRILF). Unlike conventional approaches that treat case facts and law articles as isolated nodes, the new graph integrates multiple legal elements-case facts, sentences, fact keywords, and law articles-into a unified graph structure. It includes intermediate representations to improve lexical granularity and similarity connectivities, which are refined over training epochs, thereby overcoming the limitations of one-shot graph learning. In particular, the establishment of similarity relationships between cases using keywords allows us to introduce unseen case facts into the graph, further solving the cold start issue.

Based on LF-HetG, the proposed learning framework LF-HGRILF jointly captures fine-grained lexical signals, semantic similarity among law articles, and higher-order relationships between cases and legal provisions. Specifically, we first construct a neighbor graph based on a LF-HetG via ramdom walk sampling. The representations of facts and laws are learned based on the aggregation of neighbors. To distinguish similar law articles, we introduce a Law Article Distinction Module (LADM) which enhances the discriminative features of similar laws by a reverse attention network. The contributions of our paper are listed as follow:

- We propose a novel kind of law-fact heterogeneous graph which contains richer relationships of cases and law articles.
- We introduce a new graph representation iterative learning framework which iteratively refines node representations across training epochs to capturing richer semantic features, thereby enhancing the performance of legal judgment prediction.
- We propose a law article distinction module to learn more discriminative representations of highly similar law articles, addressing the challenge of article-level semantic ambiguity.

- We conduct experiments on the CAIL2018 dataset and the results demonstrate that our approach surpasses the baseline models in both classification metrics and model interpretability.

2. Related work

2.1. Deep learning-based methods for legal judgment prediction

Legal judgment prediction has been an area of study for several decades, primarily approached as a text classification problem [5–8]. In recent years, deep neural networks have been extensively applied to LJP, achieving superior predictive performance compared to traditional machine learning methods. Wang et al. [9] introduced a multi-label charge prediction framework utilizing TextCNN and FastText, reformulating charge prediction as a multi-label text classification problem. Jiang et al. [10] proposed a hybrid approach combining neural networks and reinforcement learning to improve charge prediction while enhancing model interpretability. Considering the possibility of defendants facing multiple charges, Chen et al. [11] developed a model based on a gating mechanism to enhance charge prediction accuracy. Yang et al. [12] designed a legal decision prediction model, EPM, which integrates event extraction with consistency constraints between subtasks, extracting key event information to improve prediction accuracy. Similarly, Feng et al. [13] proposed an LJP model leveraging event extraction with constraints to refine the predictive process. Xu et al. [14] extended this approach by introducing a subtask to assess charge severity, enhancing attention allocation to penalty sentences. Additionally, they incorporated defendant location information to provide better contextual understanding, thereby improving legal judgment predictions.

Recent advancements in LJP have seen the integration of attention mechanisms to enhance predictive accuracy and interpretability. For instance, Luo et al. [15] introduced an attention-based neural network that jointly models charge prediction and relevant law article extraction within a unified framework. Similarly, Pan et al. [16] proposed an attention-based multi-layer architecture designed to handle cases involving multiple defendants. Building upon this, Li et al. [17] developed a multi-channel attention neural network framework, which effectively learns latent feature representations of case facts, defendants, and related law articles. Additionally, Gan et al. [18] represented declarative legal knowledge as a set of first-order logic rules and integrated these into a co-attention network model in an end-to-end manner.

Recently, there has been a concerted effort among researchers to enhance the ability to connect actual charges and relevant law articles. For instance, Le et al. [19] introduced a new approach to legal judgment prediction by structuring it into two stages: recall and ranking. In the recall stage, candidate results are generated using a classifier and a high-likelihood sampling strategy. The ranking stage then refines these results through verification techniques and comprehensive judgment strategies to filter out ambiguous candidates and improve the final judgment. Similarly, Zhang et al. [20] proposed a contrastive learning framework (CL4LJP), which leverages contrastive learning loss alongside legal judgment prediction loss. This framework enhances LJP performance by capturing subtle distinctions between similar law articles and charges.

However, these methods have not fully exploited the topological structure inherent in legal facts, overlooking critical relationships between case descriptions, law articles, and other legal elements. Despite previous efforts to extract key information from case descriptions, existing approaches have struggled to provide legal professionals with transparent decision-making logic. In this paper, we address this limitation by constructing a law-fact heterogeneous graph that captures intrinsic relationships between different cases. By modeling the connections between case facts and law articles, we integrate these insights to refine their embeddings, ultimately improving LJP performance.

2.2. Graph-based methods for legal judgment prediction

Graph Neural Networks (GNNs) are a class of deep learning models designed to preserve topological structures, making them highly effective in various real-world applications. With the rapid advancement of deep learning, GNNs have been extensively applied in Natural Language Processing (NLP) due to their ability to process structured data efficiently. Liu et al. [21] improved the quality of knowledge graph construction through adaptive multi-hop retrieval and redundant information filtering, and proposed a novel natural question and answer generation method by combining large model optimization to generate answers. Li et al. [22] introduced the concept of session-oriented fairness and designed a dual TCN architecture to effectively alleviate exposure differences while improving recommendation effectiveness. Li et al. [23] proposed an efficient knowledge hypergraph embedding model, which significantly reduced the computational overhead while improving the embedding effect, achieving a good balance between effect and efficiency. Zeng et al. [24] introduced the S-GCN model, which enhances multi-label text classification by constructing a global graph and leveraging semantic-sensitive graph convolutional networks. Addressing the feature sparsity in short texts, Ai et al. [25] proposed an edge-enhanced graph attention network, EMGAN, which significantly improves short text classification accuracy. Liu et al. [26] focused on hierarchical text classification and developed the LSE-HiAGM model, which optimizes label embeddings through a density coefficient and a rebalanced loss function.

Given the limitations of traditional neural networks in handling graph-structured data, many scholars have proposed GNN-based models for LJP. To better utilize case-related textual information, some researchers have constructed five types of text graphs to model case fact descriptions and communities of related law articles, though this approach overlooks inter-case connections, limiting its effectiveness [52]. Among other developments, Xu et al. [2] presented LADAN, a comprehensive model that uses graph neural networks to pinpoint subtle distinctions between similar legal articles. It leverages attention mechanisms to isolate distinguishing features from case fact descriptions. Recognizing the importance of label relationships in improving prediction accuracy, researchers have incorporated these dependencies into LJP models. Zhong et al. [27] proposed a novel framework, TOPJUDGE, which models multiple subtasks within legal judgment as topological dependencies in a directed acyclic graph. Similarly, Dong and Niu [28] introduced a relationship learning-based LJP approach that classifies nodes within a globally consistent graph and employs a variational EM-optimized masked transformer network to obtain both cross-task consistency and task-specific node representations, enhancing prediction accuracy and logical consistency. Wang et al. [29] developed the GraSCL method, transforming legal decision prediction into a node classification problem by designing a graph inference network with multiple relationship graphs to capture label dependencies.

Beyond LJP, Graph Neural Networks (GNNs) have also been widely applied in legal recommendation systems, with extensive research exploring their effectiveness [30–34]. One notable example is the work of Bhattacharya et al. [35], which introduced Hier-SPCNet, a heterogeneous graph model that integrates legal document text with citation networks. This model employs various fusion techniques to combine network and text embeddings, improving the accuracy of legal case similarity estimation. Similarly, Bi et al. [4] proposed constructing a heterogeneous knowledge graph that includes legal entities and documents. By learning graph node embeddings through a heterogeneous graph neural network, their approach enhances the representation of legal cases, providing deeper node features for downstream case recommendation tasks.

However, these methods primarily focus on integrating graph information at the individual case level or capturing the topological relationships between cases and decision subtasks. Most existing models

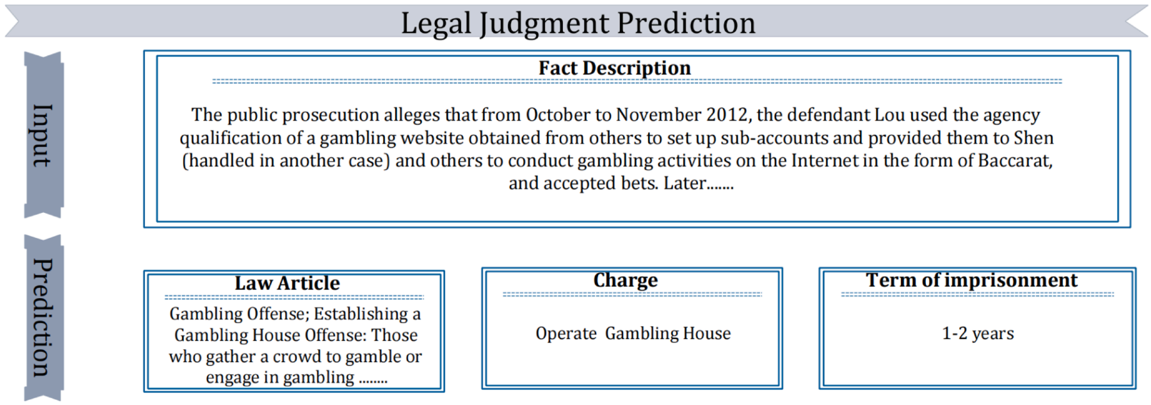


Fig. 2. Problem description of legal judgment prediction.

represent graph nodes at the document or sentence level, limiting their ability to capture a broader spectrum of legal information.

In this paper, we address this limitation by constructing case fact relationships at a finer granularity while simultaneously capturing detailed connections between cases and law articles. Furthermore, most existing graph-based methods employ a one-time neighbor sampling strategy, failing to effectively utilize feedback from past iterations. To overcome this, we propose an iterative framework that dynamically updates node neighbors, allowing for the integration of richer node information and enhancing the robustness of our model.

3. Problem description

In this section, we define the key concepts underlying our proposed approach:

Fact description: A fact description provides a detailed account of a case, including the offense, time, location, impact, means of execution, and the court's decision in the sentencing document.

Law article: A law article serves as the foundational reference for convictions, ensuring that each charge is substantiated by at least one specific law article. We use L_j to denote the j-th law article.

Charge: A charge represents a specific offense as defined in the Criminal Law of the People's Republic of China, covering crimes such as theft, fraud, and robbery.

Terms of penalty: This refers to the duration of the penalty imposed by the court, determining the length of time an offender is deprived of their liberty.

Legal judgment Prediction: The task of LJP involves predicting court rulings based on fact descriptions. An LJP model takes a fact description as input and produces three outputs: the relevant law articles, the applicable criminal charges, and the terms of imprisonment.

Fig. 2 illustrates an example of the legal judgment prediction process.

4. Legal-fact heterogeneous graph

In this section, we introduce the core components of the proposed Legal-Fact Heterogeneous Graph (LF-HetG) and describe the process of constructing it from fact descriptions and corresponding law articles (see Fig. 3).

We then compare LF-HetG with two classical legal knowledge graphs: the Legal Hybrid Knowledge Network (L-HKN) [3] and the Legal Judgment Heterogeneous Graph (L-HetG) [4]. LF-HetG offers several key advancements:

(a). Enhanced Semantic Representation: Unlike previous models, LF-HetG incorporates not only legal facts and law articles but also fact sentences and keywords as distinct nodes. This integration enriches semantic understanding and refines the granularity of legal knowledge networks.

(b). Improved Sentence Connectivity: LF-HetG strengthens the relationships between sentences by leveraging shared vocabulary, enabling efficient integration and analysis of new cases.

(c). Refined Edge Formation: To better distinguish relevant from irrelevant legal sentences, LF-HetG introduces a similarity threshold when establishing connections. This mechanism enhances the precision of legal relationships and improves interpretability in law article interactions.

Given a set of fact descriptions, we construct four types of nodes in a legal-fact heterogeneous graph: case fact descriptions, fact sentences, keywords, and law articles. The construction principles of nodes are described as follows:

(1) Fact nodes: Each represents a specific judicial case, referring to a fact description that contains a detailed statement of the case and possibly includes a masked legal basis and a masked crime allegation, denoted by F.

(2) Sentence nodes: Each case fact description is segmented into sentences and each sentence acts as a node for a fine-grained analysis. The set of sentence nodes for each case is represented as $\{S_1, S_2, \cdots, S_m\}$.

(3) Keyword nodes: In order to better extract useful keywords, we designed a case keyword extraction method. Firstly, gerunds are extracted from sentences described by facts. To minimize redundancy and maximize text clarity, keywords are selected according to the TF-IDF weight [36] to represent the key information points of the case. Since the importance of vocabulary varies under different law articles, we independently calculate the TF-IDF weight of each gerund under each law article using the following formula:

$$w_k^{L_j} = \frac{\sum_{l=1} TF(v_k, C_l^{L_j}) \cdot IDF(v_k, C^{L_j})}{\|C^{L_j}\|}, \tag{1}$$

where $w_k^{L_j}$ represents the TF-IDF weights of k-th gerunds, L_j represents the j-th law article, $C_l^{L_j}$ represents the l-th case that belongs to the law article L_j, $TF(v_k, C_l^{L_j})$ is the word frequency of the gerund v_k in this case, $IDF(v_k, C^{L_j})$ represents the inverse document frequency of the gerund v_k across all cases that fall under the law article L_j.

For each sentence, the top K gerunds with the highest TF-IDF weights are selected as keywords, which can be expressed as

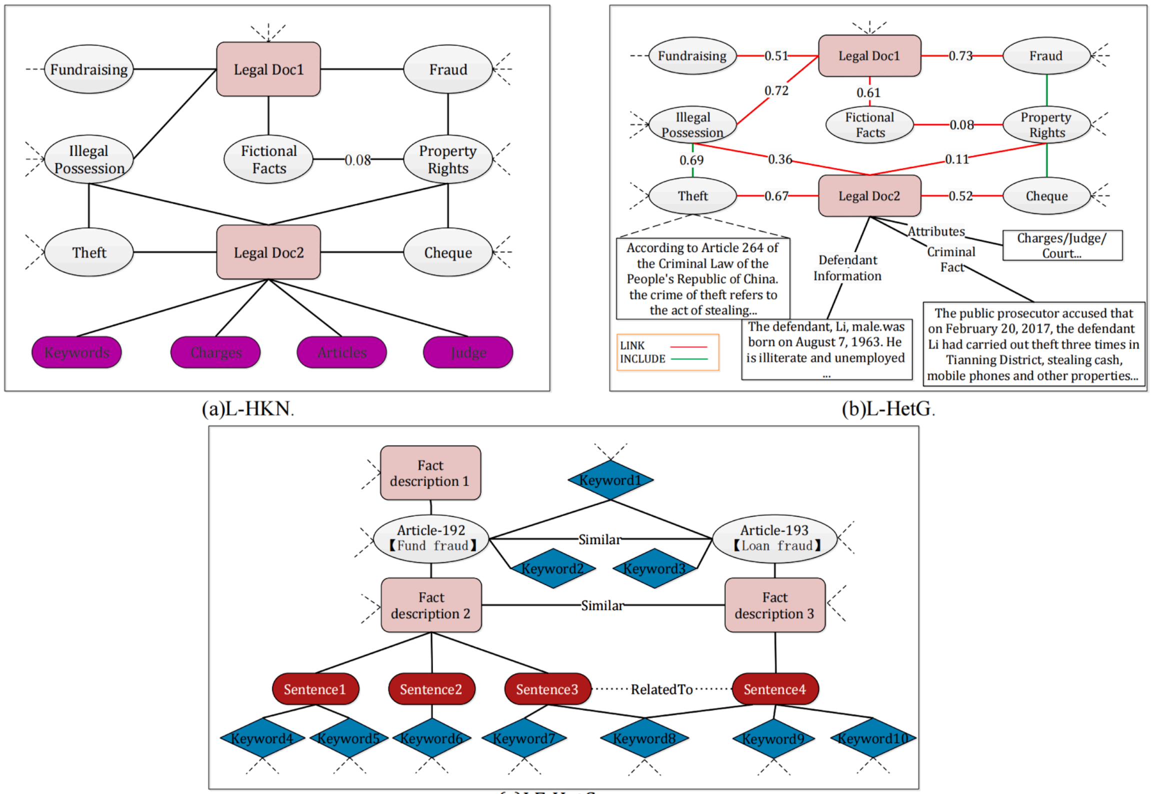
$$V = \{v_1, v_2, \cdots, v_K\}. \tag{2}$$

This refined approach enables the extraction of specific keywords customized to the nuances of various law articles, thereby enhancing the precision and adaptability of legal fact analysis.

(4) Law article nodes: Each represents a certain law article, which is the legal basis for the judgment of the case, denoted by L.

The relationships among these nodes are listed as follows:

(1) Edges between facts and sentences: Each fact node is linked to the sentence nodes that it contains, reflecting the structure of the case



(c)LF-HetG.

Fig. 3. The top left image is L-HKN [3] and the top right image is L-HetG [4], while the bottom image is LF-HetG. L-HKN contains entity nodes (in gray) and legal file nodes (in pink). Each node has its attributes (in red) and each edge has a weight. Compared to L-HKN, L-HetG introduces text information for legal documents and legal entities and considers inclusion and linking edges between nodes. Compared with both, LF-HetG contains more node types and relationships.

fact description. The rule can be expressed as

$$S_n \in F_a \rightarrow (F_a, \text{ContainedIn}, S_n). \tag{3}$$

(2) Edges between sentences and keywords: Each sentence node connects to its respective keyword nodes, indicating the representation relationships between keywords and sentences. The rule can be expressed as

$$v_k \in S_n \rightarrow (S_n, \text{Include}, v_k). \tag{4}$$

(3) Edges between fact descriptions: Through textual similarity, two fact nodes that are larger than the threshold σ_F are connected to reveal the potential connection between the cases. The rule for two fact nodes F_a and F_b can be expressed as

$$\text{Similarity}(F_a, F_b) \geqslant \sigma_F \rightarrow (F_a, \text{Sim}, F_b), \tag{5}$$

where Similarity denotes the similarity between F_a and F_b, and σ_F is a threshold.

(4) Edges between sentences: Based on the number of shared keywords between sentences, the relationship between new and existing cases is constructed, helping new cases quickly integrate into the mapping. The rule for two sentences S_n and S_m can be expressed as

$$\text{SharedKeywords}(S_n, S_m) \geqslant \sigma_S \rightarrow (S_m, \text{RelatedTo}, S_n), \tag{6}$$

where SharedKeywords computes the number of keywords that S_n and S_m contain commonly and σ_S is a threshold.

(5) Edges between law articles: Connecting similar law article nodes based on the similarity between law articles to reflect their potential connection. The rule for two articles L_i and L_j can be expressed as

$$\text{Similarity}(L_i, L_j) \geqslant \sigma_L \rightarrow (L_i, \text{Sim}, L_j), \tag{7}$$

where Similarity denotes the similarity between L_i and L_j and σ_L is a threshold.

(6) Edges between fact and law article: Ensure that each fact node is linked to its related law article node to clarify the legal basis of the case. The rule can be expressed as

$$F_a \succ L_j \rightarrow (F_a, \text{ReferTo}, L_j). \tag{8}$$

We also compare our LF-HetG against the Legal Hybrid Knowledge Network (L-HKN) and the Legal Judgment Heterogeneous Graph (L-HetG) in Fig. 3. The LF-HetG offers a more information-rich structure by incorporating multiple types of nodes, including case fact descriptions, factual sentences, keywords, and law articles. This integration not only enriches the relationships within the graph but also enables the capture of more fine-grained semantic associations. The diversity of nodes and the complexity of connections in LF-HetG enhance its adaptability and flexibility when handling various legal cases. This design allows the model to effectively respond to changes in the legal environment and the addition of new cases. Furthermore, by introducing fine-grained associations of law articles, LF-HetG can more accurately capture the nuanced meanings and details of legal facts, thereby improving the accuracy of legal judgment prediction.

In summary, LF-HetG's enriched structure and detailed semantic associations provide a robust framework for legal judgment prediction, offering significant advantages over previous models like L-HKN and L-HetG.

5. Our method

Based on law-fact heterogeneous graphs, we propose a graph iterative learning framework LF-HGRILF, as shown in Fig. 4. Given a set of fact descriptions, we construct a law-fact heterogeneous graph G, and

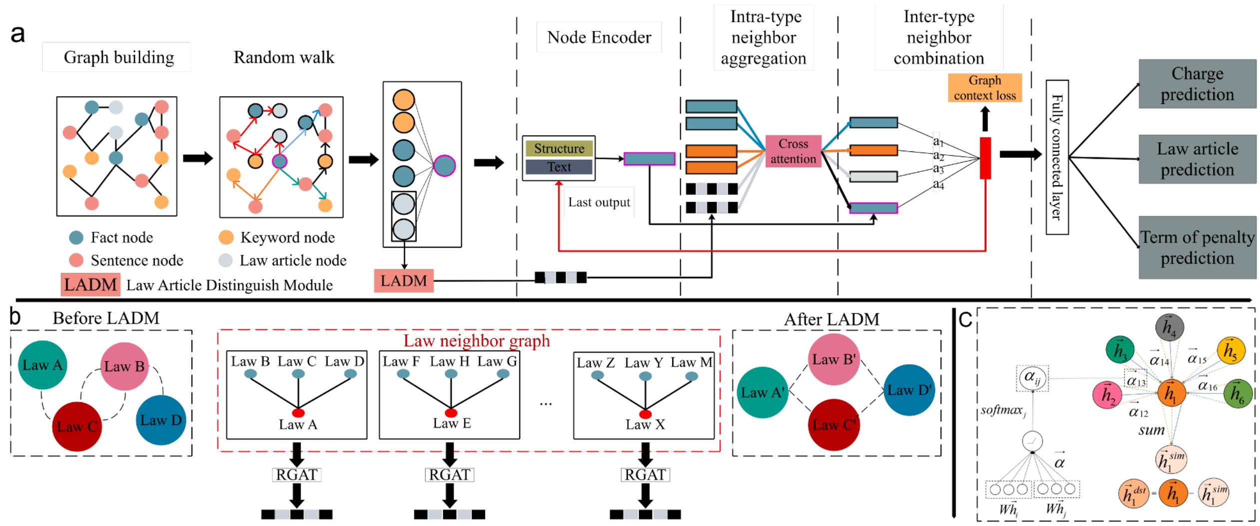


Fig. 4. a. The overview of LF-HGRILF. It takes the heterogeneous graph of legal facts and the pre-trained features of nodes as input, then generates the representations of fact nodes $\gamma_{v_{fact}}$ through graph representation iterative learning and finally uses the representations for the downstream decision prediction task; b. LADM: it takes each law article node as the central node and extracts distinguishable features of law articles through the Reverse Graph Attention Network (RGAT); c. RGAT: it first calculates the importance of neighboring nodes to the central node through a graph attention network. Then, it subtracts the neighbor vectors weighted by importance from the central node vector to obtain the distinguishable central node vector.

further build a neighbor graph G_n for each fact node and law node by path-based random sampling. That is, for each fact or law node v, we find neighbor nodes v_c in G by following pre-defined meta-paths. Then, we use different pre-trained models to obtain the initial embedding of each node and perform heterogeneous content aggregation operations for different types of neighbor nodes to obtain the deep representation of the graph.

To address the challenge of distinguishing similar law articles, we have designed a specialized module that enhances the computation of law node embeddings, called Law Article Distinction Module (LADM), as illustrated in Fig. 4.b. This module employs an iterative learning strategy to refine node representations over multiple training cycles.

During each iteration, the neighbor graph for each node is resampled, allowing the model to effectively manage uncertainty and adapt to new information. The embeddings generated in the previous iteration serve as inputs for the current one, facilitating the integration of feedback and promoting the evolution of more accurate representations.

In the final stage, fact node embeddings are utilized as inputs to train a classifier. This classifier predicts the corresponding law articles, charges, and terms of imprisonment associated with each case, thereby enhancing the model's capability to make precise legal judgments.

This iterative approach, combined with the law article distinction module, significantly improves its performance in distinguishing between similar law articles, leading to more accurate and reliable legal judgment predictions.

5.1. Law-fact heterogeneous graph representation iterative learning framework

5.1.1. Type neighborhood random walk sampling

A significant challenge in heterogeneous graph analysis is how to sample different types of neighbors effectively and sufficiently. Most graph neural networks only consider the features of first-order neighbors when aggregating neighbors, such as [37–39]. That is, only those nodes connected by one edge with a certain node are collected. This leads to those nodes with deeper and multi-hop relationships not being taken into account. As a result, the aggregation of neighbors would fail to incorporate the characteristics of the latently correlated neighbor nodes.

Inspired by [40], we employ a path-based neighborhood random walk sampling strategy. Specifically, we design five kinds of meta-paths for the sampling strategy, which are shown in Table 1. These meta-paths are constructed based on the internal logic of legal knowledge, leveraging multiple relationships such as similarity, containment relationships, and citation relationships among fact nodes, sentence nodes, keyword nodes, and law article nodes. For example, $P1$, $P4$, and $P5$ are associated based on similarity, $P2$ is associated based on text content relevance, and $P3$ establishes targeted associations through keywords. They complement each other, reflecting the complex relationships in legal facts and law articles from multiple dimensions, enabling the model to comprehensively understand the connections between cases and law articles.

For these sequences of nodes from a node v, we count the occurrence frequency of each node and take the nodes in the top k frequencies are chosen as the neighboring nodes of v to construct a neighbor graph, denoted as $G_n(v)$. By adopting a dynamic stochastic walk process, it takes into account the diversity of different neighboring nodes. Furthermore, it accounts for nodes in the heterogeneous graph that, despite not having direct connections, may exert significant and underlying influences. This comprehensive consideration allows the sampling results to reflect not only the information of directly connected neighbors but also to cover a wider range of topologies in the law-fact heterogeneous graph.

5.1.2. Node encoder

In the legal fact heterogeneous graph, each node encompasses two distinct types of content information: the first derived from the graph structure, and the second originating from the associated text content. For each node $v \in \Lambda$, we employ Metapath2Vec [41] based on the meta-paths in Table 1 to obtain its graph content embedding, noted $\mathbf{x}_s(v)$, and employ Glove [42] to obtain its text content embedding, noted $\mathbf{x}_t(v)$. To incorporate feature interactions more deeply and capture larger expressive capability, we adopt the Bi-LSTM model to fuse the content features of nodes. Then we employ an average pooling on all hidden states to obtain a feature representation of v, denoted as $g(v)$. In our iterative framework, the feature representation of each node from the previous training round would be utilized in the current round. Without ambiguity, we use $g^m(v)$ to denote the resulting feature representation of v in the m-th round. Formally, the process is described by the following formula:

$$\mathbf{X} = \phi_t \mathbf{x}_t(v) \oplus \phi_s \mathbf{x}_s(v), \quad (9)$$

Table 1
Metapaths of nodes and their descriptions.

No	Metapaths	Descriptions
$P1$	Fact $\xrightarrow{\text{Sim}}$ Fact $\xrightarrow{\text{ReferTo}}$ Law $\xrightarrow{\text{BeReferred}}$ Fact	The head fact node connects to another fact node based on similarity, and then links to the relevant law article.
$P2$	Fact $\xrightarrow{\text{ContainedIn}}$ Sentence $\xrightarrow{\text{RelatedTo}}$ Sentence $\xrightarrow{\text{BeContained}}$ Fact $\xrightarrow{\text{ReferTo}}$ Law $\xrightarrow{\text{BeReferred}}$ Fact	This metapath finds relevant cases and relevant law articles through the correlation of sentences in the cases.
$P3$	Fact $\xrightarrow{\text{ContainedIn}}$ Sentence $\xrightarrow{\text{Include}}$ Keyword $\xrightarrow{\text{BeInclude}}$ Sentence $\xrightarrow{\text{BeContained}}$ Fact $\xrightarrow{\text{ReferTo}}$ Law $\xrightarrow{\text{BeReferred}}$ Fact	This metapath uses the same keywords to find similar cases and related law articles.
$P4$	Law $\xrightarrow{\text{Sim}}$ Law	This metapath directly connects two law nodes through similarity.
$P5$	Law $\xrightarrow{\text{BeReferred}}$ Fact $\xrightarrow{\text{Sim}}$ Fact $\xrightarrow{\text{ReferTo}}$ Law	This metapath identifies law articles that are confusing on the similarity.

$$g^m(v) = \begin{cases} \frac{[\overrightarrow{LSTM}(\mathbf{X}), \overleftarrow{LSTM}(\mathbf{X})]}{z}, m = 0 \\ AvP(\frac{[\overrightarrow{LSTM}(\mathbf{X}) \oplus \overleftarrow{LSTM}(\mathbf{X})]}{z}, \mathbf{W}_u \gamma_v^{(m-1)}), m > 0, \end{cases} \tag{10}$$

where m represents the number of iterations and z represents the number of content features(here is 2). $\mathbf{W}_u, \phi_t, \phi_s \in \mathbb{R}^d$ are trainable weight matrix, $\gamma_v^{(m-1)} \in \mathbb{R}^d$ denotes the last round output, $AvP(\bullet)$ denotes average pooling,ω denotes heterogeneous content count and the operator $\oplus$ denotes concatenation. Generally, this content encoder first applies different transformation functions to the heterogeneous content, and then utilizes Bi-LSTM to learn the interactions among the information features. Finally, we use $g^m(v)$ to represent the heterogeneous content embedding of node v.

5.1.3. Intra-type neighbor aggregation

In order to aggregate the content embeddings of the same type neighbors of each node, we design different aggregation functions for different types $p \in \Omega$ of nodes. Specifically, in Section 5.1.1, we elaborate on the sampling strategy for each type of neighbor node set and construct a neighbor graph with a fixed number of neighbors in the same type. Here, we use a neural network τ^p to aggregate the heterogeneous content embeddings of the neighbor nodes v_c. Because of the uniqueness of law articles, the direct feature fusion of the neighbors of law article nodes will aggravate the confusion between law articles. Therefore, in particular, we elaborate in Section 5.1.4 how to effectively aggregate the nodes of law articles to ensure that different law articles can be more accurately distinguished and understood in the model.We denote the set of type p neighbor nodes for each node v as $N_p(v)$, the mathematical formulation is as follows:

$$f_1^p(v) = \tau^p_{v_c \in N_p(v)}\{g^m(v), g^m(v_c)\}, \tag{11}$$

where $f_1^p(v) \in \mathbb{R}^d$ (d is the dimension of the heterogeneous content embedding). τ^p is the aggregator for type p neighbors, which can be a fully connected neural network, a convolutional neural network, a recurrent neural network, etc. In this paper, different types of nodes use different aggregation methods. For non-law article nodes, we use a cross-attention mechanism [43], which not only optimizes the information integration process, but also significantly improves the model's ability to recognize and respond to key features through a dynamic weight allocation strategy. For the law article nodes, we use a reverse graph attention mechanism, which is specifically designed to capture the unique properties of law articles to ensure that the model can accurately distinguish between different legal concepts and articles.Therefore, we reformulate $f_1^{\{p|law \notin p\}}(v)$ as follows:

$$\mathbf{q} = \mathbf{W}_Q g^m(v), \tag{12}$$

$$\mathbf{k} = \mathbf{W}_K g^m(v), \tag{13}$$

$$\mathbf{v} = \mathbf{W}_V g^m(v_c), \tag{14}$$

$$\alpha = \mathrm{softmax}(\frac{\mathbf{q}\mathbf{k}^T}{\sqrt{d}}), \tag{15}$$

$$f_1^{\{p|law \notin p\}}(v) = \alpha \mathbf{v}, \tag{16}$$

where $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d\times d}$ are all trainable weight matrices, and α is the attention weight corresponding to the neighbor node v_c, normalized by the Softmax function to ensure that the sum of the weights is 1.

5.1.4. Law article distinction module

In Section 5.1.3, we discussed in detail the neighbor aggregation policy of the same type for nodes other than legal nodes. However, for law article nodes, direct feature aggregation may make similar law sentences closer to each other in feature space, which may blur the boundaries between them. There is often textual similarity between law articles, which may affect the model's accurate prediction of the specific legal basis of cases. Previous works have addressed this issue by using graph information distillation and establishing law article communities to distinguish similar law articles. However, this method relies on the construction of the graph and the division of communities [2,44,45], if the topology of the graph does not accurately reflect the actual differences between law articles, it may affect the distinguishing capability. To solve this problem, we propose a Reverse Graph Attention Network (RGAT), as illustrated in Fig. 4.(c)). When constructing adjacency subgraphs in Section 5.1.1, each law article node can find similar law articles based on meta-paths. For those law articles that are less likely to be confused with others, we adopt a specific padding strategy to ensure the stability and accuracy of the model's predictions. Specifically, we first calculate the multihead-attention values between the center law article node and its neighbor nodes. Then, by subtracting the weighted neighboring node vectors from the central node vector, we finally obtain the distinguishing features of the central node. It can be denoted as follows:

$$e_{ij}^h = \mathbf{W}_B([g^m(v_i^{law}); g^m(v_j^{law})]), j \in N_{law}, \tag{17}$$

$$\alpha_{ij}^h = \frac{\exp(\mathrm{LeakyReLU}(e_{ij}^h))}{\sum_{k \in N_i} \exp(\mathrm{LeakyReLU}(e_{ik}^h))}, \tag{18}$$

$$Z_{v_i^{\mathrm{law}}}^h = \sum_{j \in N_i} \alpha_{ij}^h g(v_j^{\mathrm{law}}), \tag{19}$$

$$f_1^{\mathrm{law}}(v_i^{\mathrm{law}}) = g(v_i^{\mathrm{law}}) - \mathrm{Concat}(Z_{v_i^{\mathrm{law}}}^1, \ldots, Z_{v_i^{\mathrm{law}}}^h)\mathbf{W}_O, \tag{20}$$

where $Z_{v_i^{law}}^h$ represents the weighted sum of neighbor node features for a law article node v_i^{law} under the h-th attention head. N_{law} represents the number of neighbors of law article (fixed at 3 here), $\mathbf{W}_V \in \mathbb{R}^{2d\times 1}$ and $\mathbf{W}_O$ are both trainable matrix, α_{ij} is the attention coefficient.

5.1.5. Inter-type neighbor combination

In Section 5.1.3, we obtain the embedding based on the nodes of the same type of neighbors through the intra-type neighbor aggregation step. Next, we combine these different types of neighbor features as the final embedding of the node. Considering that different types of neighbors may contribute differently to the node embedding, the attention mechanism [46] is used to measure the importance of different neighbor types and weight the combination of their contributions accordingly. The final embedding γ_v of a node v is the weighted sum of its heterogeneous content embedding $g^m(v)$ and all types of neighbor aggregation embedding $f_1(v)$. We set $\Phi(v) = \{g^m(v) \cap f_1(v)\}$, which can be mathematically expressed as:

$$\gamma_v = \sum_{f_i \in \Phi(v)} \beta^{v,i} f_i, \tag{21}$$

$$\beta^{v,i} = \frac{\exp\{\text{LeakyReLU}(\boldsymbol{\delta}^T[f_i \oplus f_1(v)])\}}{\sum_{f_j \in \Phi(v)} \exp\{\text{LeakyReLU}(\boldsymbol{\delta}^T[f_j \oplus f_1(v)])\}}, \tag{22}$$

where $\gamma_v \in \mathbb{R}^d$ (d is the dimension of the output), $\beta^{v,*}$ denotes the importance weight between different embedding, and $\delta \in \mathbb{R}^{2d\times 1}$ is the attention parameter. To maintain the consistency of the model and simplify the tuning process, the dimensions of the graph structure embedding, aggregated content embedding and output node embedding d are kept consistent in the model.

5.2. Model training

5.2.1. Graph representation learning model training

In this section, we describe how to optimize heterogeneous graph representation learning. The expression for our optimization objective is as follows:

$$\arg\max_{\Theta_G} \prod_{v\in\Lambda} \prod_{p\in\tau} \prod_{v_c \in CN_v^p} p(v_c|v;\Theta_G), \tag{23}$$

Where Θ_G is the target parameter, CN_v^p denotes the set of p type neighbors of node v, and the conditional probability $p(v_c|v;\Theta_G)$ can be defined as a heterogeneous softmax function with the following expression:

$$p(v_c|v;\Theta_G) = \frac{\exp\{\gamma_{v_c} \cdot \gamma_v\}}{\sum_{v_q \in \Lambda_p} \exp\{\gamma_{v_q} \cdot \gamma_v\}}, \tag{24}$$

Λ_p denotes the set of nodes of type p in the graph. For each pair of nodes (v, v_c), we compute the loss of the positive samples and compare it with the loss of the negative samples, we define the graph context loss function as:

$$\mathcal{L}_G = -\log\sigma(\gamma_{v_c} \cdot \gamma_v) + \mathbf{E}_{v^- \sim z(v)} \log(\gamma_{v^-} \cdot \gamma_v), \tag{25}$$

where $(\gamma_{v_c} \cdot \gamma_v)$ denotes the similarity of the connection between the node v_c and v, and σ is a sigmoid function, and v^- is a negative sample drawn from a random distribution of negative samples $z(v)$.

5.2.2. Prediction model training

LJP involves multiple learning tasks. In the prediction layer, we introduce a softmax classifier that takes as input the vectors $\gamma_{v_{fact}}$, which is a vector of cases obtained from graph representation learning, using the following equations:

$$\hat{y}_i = \text{softmax}(\mathbf{W}_i \gamma_{v_{fact}} + b_i), \tag{26}$$

where i represents the total number of tasks. Our training goal is to minimize the cross-entropy loss by comparing the true labels with the predicted labels of each subtask. We sum up the losses of all sub-tasks to arrive at the overall prediction loss with the following expression:

$$\mathcal{L}_L = -\sum_{i=1}^{3} \sum_{a=1}^{A_i} y_{i,a} \log(\hat{y}_{i,a}), \tag{27}$$

where A_i represents the number of classes under the i-th task, $y_{i,a}$ denotes the one-hot vector of class a under the i-th task, and $\hat{y}_{i,a}$ represents the predicted class.

Finally, we simultaneously optimized $\mathcal{L}_G$ and $\mathcal{L}_L$ to generate better legal case node embeddings for downstream classification tasks by jointly optimizing the graph representation learning model and the legal judgment classification model. At the same time, the classification loss is used to guide feature fusion in graph representation learning. Therefore, the final loss can be expressed as:

$$\mathcal{L} = \mathcal{L}_L + \mathcal{L}_G. \tag{28}$$

We summarize the pseudocode for the law-fact heterogeneous graph representation iterative learning framework Algorithm 1. Initially, the parameters of the graph representation learning model and the legal judgment classification model are initialized, and the node content feature embeddings are pre-trained using the TF-IDF-Glove model [47]. Subsequently, the neighbor graph is generated using random walk sampling, and the graph representations for the nodes are computed within each mini-batch. Next, the joint loss for the graph representation learning model and the classification model is calculated, and the model parameters are updated through backpropagation. After each iteration, the node features are updated to enhance the representation capability. Finally, the optimized model parameters are used to predict the probability distribution of legal judgments, thereby inferring the outcomes of legal judgment tasks. Note that during the proposed model training process, the word embeddings obtained from the TF-IDF-Glove model are frozen, which significantly accelerates the convergence of the model parameters.

Algorithm 1 Law-fact heterogeneous graph representation iterative learning framework.

Input: Law-fact heterogeneous graph G, initialization features $\mathbf{x}_t(v)$ and $\mathbf{x}_s(v)$ of each node, embedding dimension d, learning rate for graph representation model α_G, learning rate for legal judgment classification model α_J, number of epochs m.
Output: Probability distribution for legal judgment prediction $P_{prediction}$
1 Initialize the parameters of the graph representation learning model Θ_G and the legal judgment classification model Θ_J;
2 **for** *each epoch* $l = 1$ *to* m **do**
  2.1 **for** *each MiniBatch in MiniBatches* **do**
    2.1.1 Generate the neighbor graph $G_n(v)$ by type neighborhood random walk sampling in Section 5.1.1;
    2.1.2 Generate the graph representation γ_v^m for each node using the current graph model parameters Θ_G, initialization features $\mathbf{x}_t(v)$,$\mathbf{x}_s(v)$ and the neighbor graph $G_n(v)$ and the last round output $\gamma_v^{(m-1)}$;
    2.1.3 Calculate the loss for the graph representation learning model $\mathcal{L}_G$ and the multi-class classification loss $\mathcal{L}_J$ for legal judgments using the updated node embeddings γ_v;
    2.1.4 Update the parameters Θ_G and Θ_J using their respective learning rates α_G and α_J by backpropagating the combined loss $\mathcal{L}_G + \mathcal{L}_J$;
  end
end
3 Return Legal judgment probability distribution $P_{prediction}$;

6. Experiments

6.1. Experimental setup

6.1.1. Dataset

To demonstrate the effectiveness of our approach, we use two datasets: CAIL-small and CAIL-big, both derived from the China Artifi-

Table 2
The CAIL dataset details.

Dataset	CAIL-small	CAIL-big
Training set	106,081	1,516,148
Testing set	22,825	257,426
Law articles	99	118
Charges	119	130
Penalty terms	11	11

cial Intelligence and Law Challenge [48]. CAIL-small is a widely adopted benchmark dataset constructed by selecting a balanced subset of legal cases from the full corpus, while CAIL-big refers to the larger original version containing the complete set of available cases. These datasets consist of cases with factual descriptions, matching law articles, charges, and subsequent penalty provisions. The detailed statistics of the datasets are shown in Table 2. In our experiments, we primarily use CAIL-small for model development and ablation studies due to its balanced distribution, and CAIL-big to evaluate generalization performance on large-scale, real-world data.

(1) Samples with case descriptions of less than 10 words were removed.
(2) Cases containing only a single statutory provision and offense were screened.
(3) Only those crimes and laws with a frequency of more than 100 are retained.

6.1.2. Metrics

To evaluate the performance in multi-class classification, we adopted four widely recognized metrics, following the mainstreamed methodologies [2,12,49]. Specifically, these are accuracy (Acc.), macro precision (MP), macro recall (MR), and macro-F1 score (F1). The calculation of MP, MR, and the macro-F1 score is based on the macro average, ensuring that each class is given equal importance.

6.1.3. Baselines

For a better comparative test, we chose several state-of-the-art approaches that perform well on legal judgment prediction.

1. TF-IDF + SVM [50] constructs textual feature information through TF-IDF and uses Support Vector Machines (SVMs) as a classifier.

2. CNN [51] extracts text feature information by convolutional neural networks with multiple filters and softmax as a classifier.

3. RCNN [52] combines RNN and CNN to improve the classification of texts.

4. HARNN [53] constructs a network architecture including word level and sentence level levels, applies bidirectional recurrent neural network at each level to process sequence information, and combines attention mechanism to calculate the importance weights of words to sentences and sentences to documents respectively, and obtains the representation with attention information by weighted summation. Finally, documents are classified by full connection layer and classifier.

5. FLA [15] invokes an attention neural network to simultaneously address the crime prediction task and the related law extraction task within a cohesive framework. By leveraging the extracted related laws, the precision of crime prediction is enhanced.

6. TOPJUDGE [27] formalizes the dependencies among subtasks as directed acyclic graphs, and is a topological multi-task learning framework, which incorporates multiple subtasks and DAG dependencies into judgment and prediction.

7. Text-GCN [54] converts text into a graph structure, then uses graph convolutional networks to aggregate neighbor node information to update node feature representation, and finally summarizes node feature and realizes text classification through a classifier.

8. MPBFN-WCA [12] constructs a network with multi-perspective feature extraction capability and two-way feedback mechanism, extracts relevant features of legal cases from different angles, and uses two-way feedback interaction information to train the network to achieve more accurate prediction of legal judgment results.

9. LADAN [2] is an end-to-end legal judgment framework constructed by graph neural network. The law article communities are constructed by similarity, the distinguishing features of law articles are obtained by an attention method, and the legal judgments are classified by softmax after being spliced with the descriptive features of facts.

10. NeurJudge [49] uses the situational awareness module to capture contextual details and environmental factors related to the case, combines the situational information with legal knowledge and case data, and realizes the accurate prediction of the outcome of the legal judgment through the training and learning of the model.

11. R-former [28] formalizes LJP as a node classification problem based on the global consistency graph. It uses mask transformer networks for node coding, graph convolutional networks in classifiers for label propagation, and optimizes them by variational expectation maximization.

12. CL4LJP [55] formulates the task as a supervised contrastive learning problem. It distinguishes similar law articles within the same chapter and similar charges under related articles, enabling the model to capture fine-grained differences. It jointly optimizes contrastive objectives and prediction tasks to enhance label alignment and overall performance.

13. LKEPL [56] integrates external legal knowledge into a prompt learning framework by extracting legal entities and knowledge at the article level, projecting them into the transformer space and injecting them into the BERT attention mechanism. It combines knowledge-enhanced inputs with task-specific prompt learning to improve performance and interpretability, particularly under a few-shot scenarios.

6.1.4. Experiment settings

We use PyTorch with the Adam algorithm [57], setting dropout to 0.3. Since our work involves Chinese, we use the Jieba participle tool. We employ minibatch training to avoid memory overload and utilize the GLOVE model for word embedding to obtain embedding-based distances. To ensure efficient training, we stop the process if the validation loss does not decrease for 20 consecutive validations.

We do not sample sentence nodes here because they only serve as springboard nodes for meta-paths. We use the Adam optimizer to minimize the learning objective, setting the learning rate of the graph representation learning model to 1×10^{-3} and the learning rate of the LJP verdict model to 5×10^{-4}. All other parameters follow the default values specified in the original model provided in this paper.

6.2. Main performance comparison

The proposed model demonstrates significant advantages in resource utilization and computational efficiency, with experimental measurements indicating a model size of 0.6GB, a per-sample inference latency of merely 7.5ms, and a single-epoch training time of 150s on small-scale datasets. As opposed to traditional full-batch gradient descent optimization, we implement subgraph-based batch gradient descent-a methodology that jointly optimizes the graph representation learning model and multi-task classification model by partitioning the complete neighbor graph into batch-sized subgraphs, wherein the batch parameter requires dynamic adjustment according to dataset scale to mitigate out-of-memory risks. Experimental results demonstrate that compared to conventional training approaches, this subgraph-based batch strategy achieves 60 % acceleration in both inference speed and training throughput, validating its effectiveness for computational efficiency optimization.

Next, we present a comparative analysis to demonstrate the effectiveness of the LF-HGRILF model. We evaluate its performance against several state-of-the-art approaches using the CAIL-small and CAIL-big datasets. The results are shown in Tables 3 and 4, respectively.

Table 3
Results on the CAIL-small dataset.

Method	Law articles				Charges				Terms of penalty			
	Acc.	MP	MR	F1	Acc.	MP	MR	F1	Acc.	MP	MR	F1
TF-IDF + SVM [50]	76.5	43.2	40.1	39.7	79.8	45.9	42.7	42.8	33.3	27.7	25.0	24.6
FLA [15]	77.7	75.2	74.1	72.8	81.0	79.1	77.9	76.8	36.3	30.8	28.2	27.8
CNN [51]	78.6	75.9	74.6	73.6	82.2	81.6	79.7	78.8	35.2	33.0	29.1	29.7
RCNN [52]	79.1	76.6	75.1	74.2	82.5	81.9	79.7	79.1	35.5	33.8	30.4	30.3
HARNN [53]	79.7	75.1	76.5	74.7	83.4	82.2	82.3	80.8	36.0	34.5	31.0	31.2
TOPJUDGE [27]	79.8	79.5	73.4	73.3	82.0	83.1	79.3	79.0	36.1	34.5	32.5	29.2
Text-GCN [54]	79.8	79.7	73.4	73.4	82.3	83.2	79.2	79.0	36.0	34.7	32.5	29.2
MPBFN-WCA [12]	79.1	76.3	76.0	74.8	82.1	82.3	80.7	80.7	36.0	31.9	28.6	29.9
LADAN [2]	82.3	80.8	81.6	81.8	84.8	83.3	82.8	82.9	39.4	36.9	33.3	34.1
NeurJudge [49]	85.3	84.9	84.4	83.2	85.8	84.6	83.9	83.2	39.0	37.0	33.8	34.0
R-former [28]	86.9	85.3	83.4	84.3	86.1	83.6	84.4	83.6	38.8	37.4	35.2	35.3
CL4LJP [55]	83.7	79.7	76.2	84.9	83.9	82.6	81.7	81.8	39.5	38.7	**36.4**	**37.4**
LKEPL [56]	84.1	81.3	79.5	86.4	86.6	85.1	82.3	82.9	40.2	**39.2**	35.6	36.8
LF-HGRILF	**91.1**	**87.3**	**84.6**	**85.6**	**90.5**	**88.6**	**86.7**	**87.4**	**40.2**	37.5	36.0	35.8

Table 4
Results on the CAIL-big dataset.

Method	Law articles				Charges				Terms of penalty			
	Acc.	MP	MR	F1	Acc.	MP	MR	F1	Acc.	MP	MR	F1
TF-IDF + SVM [50]	89.9	67.8	59.6	61.2	85.8	69.7	62.0	64.5	54.1	38.7	37.5	39.2
FLA [15]	93.2	75.2	65.2	66.5	92.4	76.2	68.2	70.0	57.6	44.0	38.7	41.5
CNN [51]	95.2	75.5	76.7	75.5	95.0	82.2	78.9	81.0	55.4	45.2	36.5	39.7
RCNN [52]	96.2	76.8	76.2	77.5	95.6	86.5	79.2	81.7	55.2	45.4	37.6	39.8
HARNN [53]	97.0	76.8	77.8	76.8	95.3	87.3	79.3	81.7	55.6	45.6	38.8	39.7
TOPJUDGE [27]	95.4	75.4	75.2	75.2	95.2	87.8	79.2	81.2	57.0	47.2	40.1	40.1
Text-GCN [54]	96.2	74.2	76.7	76.2	96.5	87.3	79.4	82.0	57.2	47.5	42.1	41.0
MPBFN-WCA [12]	96.5	75.1	78.3	78.5	96.2	85.9	79.5	81.9	57.8	48.2	43.2	43.5
LADAN [2]	96.7	80.2	81.2	82.5	96.5	88.1	82.8	83.7	58.0	49.6	44.5	44.5
NeurJudge [49]	96.8	83.2	82.4	83.1	95.8	87.2	83.2	83.7	58.0	48.5	46.1	46.3
R-former [28]	96.6	84.4	83.2	83.5	96.1	87.7	82.6	84.0	57.0	**50.1**	46.5	**47.0**
CL4LJP [55]	95.4	86.4	78.2	80.2	94.2	87.3	79.0	80.8	57.1	46.3	42.8	43.1
LKEPL [56]	96.5	85.8	77.2	80.4	96.6	**88.1**	79.1	81.6	**58.9**	48.6	43.2	44.1
LF-HGRILF	**97.2**	**86.5**	**85.4**	**85.2**	**97.1**	87.7	**83.2**	**84.3**	58.0	49.2	**47.0**	46.8

On the CAIL-small dataset, the LF-HGRILF model obtains the top performance in the three tasks. Specifically, it surpasses the second-best method, R-former, in F1 for law article and charge prediction, with increases of 1.3 % and 3.8 %, respectively. On the CAIL-big dataset, the LF-HGRILF model yields the best performance in terms of most metrics. Specifically, LF-HGRILF achieves the highest accuracies of 97.2 % and 97.1 %, and the top F1 scores of 85.2 % and 84.3 %, in law article and charge prediction, outperforming all other methods tested. However, in the task of penalty term prediction, R-former demonstrates superior accuracy performance on the CAIL-big dataset. A plausible explanation is that R-former incorporates explicit numerical modeling mechanisms, enabling it to capture quantitative legal information more effectively. Conversely, CL4LJP and LKEPL exhibit better generalization on the CAIL-small dataset. This may be attributed to the fact that CL4LJP integrates multi-angle feature fusion with a multi-label classification paradigm for sentence-level prediction, while LKEPL leverages a hierarchical legal knowledge embedding framework to model interrelations among multiple penalty terms. Specifically, the multi-label classification strategy of LKEPL allows it to capture contextual dependencies between relevant legal articles, thereby enhancing the accuracy of penalty term co-prediction in small-sample scenarios.

Comparison with traditional machine learning methods: The LF-HGRILF model outperforms traditional machine learning methods, such as TF-IDF combined with SVM. This improvement is attributed to the deep learning model's superior feature extraction capabilities and its enhanced semantic understanding, particularly in processing unstructured data and modeling long texts.

Comparison with case description-based models: When compared to models that utilize only case descriptions, such as CNN, RCNN, HARNN, and Text-GCN, the LF-HGRILF model demonstrates superior performance. This indicates that incorporating topological dependencies between subtasks significantly enhances the accuracy of legal judgment prediction. Additionally, models like LADAN and NeurJudge, which address the confusion of similar law articles, highlight the importance of resolving such ambiguities to improve LJP accuracy.

These findings underscore the LF-HGRILF model's effectiveness in enhancing the accuracy of legal judgment prediction by leveraging the latent relationship between law texts and mitigating the confusion of similar law articles.

6.3. Ablation study

To further illustrate the effectiveness of the Law Fac heterogeneous graph, law article distinction network, and iterative framework, we conducted ablation experiments on the LF-HGRILF model using the CAIL-small dataset.

6.3.1. Module ablation

First, we focus on individual modules in LF-HGRILF. We removed specific modules from LF-HGRILF to assess their impact on performance (as shown in Table 5). We consider the following variations: (1) LF-HGRILF without LADM and iteration: This variant removes both the law article distinction module and the feature iteration operation. (2) LF-HGRILF without LADM: This variant removes the law article distinction module, meaning the features of similar law articles are not differentiated between the law article node vectors. (3) LF-HGRILF without

Table 5
Ablation analysis of LF-HGRILF on CAIL-small.

Method	Law articles				Charges				Terms of penalty			
	Acc.	MP	MR	F1	Acc.	MP	MR	F1	Acc.	MP	MR	F1
LF-HGRILF	91.1	87.3	84.6	85.6	90.5	88.6	86.7	87.4	40.2	37.5	36	35.8
w/o iteration & LADM	85.7	82.9	77.5	78.9	84.8	83.2	78.7	79.7	34.8	28.8	29.1	27.2
w/o LADM	86.2	84.1	79	80.3	85.4	84.1	80.3	81	35.5	34.8	28.4	27.4
w/o iteration	88.1	85.2	80.7	82.1	88.6	85.7	82.3	83.4	36.5	34	29.1	29.3

Table 6
Ablation analysis of LF-HetG on CAIL-small.

Method	Law articles				Charges				Terms of penalty			
	Acc.	MP	MR	F1	Acc.	MP	MR	F1	Acc.	MP	MR	F1
LF-HGRILF	91.1	87.3	84.6	85.6	90.5	88.6	86.7	87.4	40.2	37.5	36	35.8
w/o keywords & S-S edges	86.5	83.9	79.3	80.6	85.3	84.3	80.1	81.2	36.8	33.6	31.6	30.1
w/o keywords	87.9	83.5	79.9	81.1	87.9	84.6	81.6	82.6	38.1	36.2	31.5	30.6
w/o S-S edges	87.4	85.0	81.6	82.6	87.0	85.2	83.2	83.8	36.8	33.8	30.9	30.7

S-S edges refer to Sentence-Sentence edges.

iteration: This variant eliminates the feature iteration operation, which means that each iteration uses the initialization vector without updating it based on previously learned features. The results indicate a significant performance reduction when both the law article distinction module and the iteration strategy are removed. Specifically, the F1 scores for crime prediction and law article prediction decrease by 5.9 % and 5.4 %, respectively. When only one module is removed, the F1 scores for crime prediction drop by 4.6 % (Baseline + Iteration) and 2.3 % (Baseline + LADM), and the F1 scores for legal fact prediction decrease by 4 % (No LADM) and 2.2 % (No Iteration). These findings highlight the importance of the law article distinction module in extracting distinguishing features of legal facts within the LF-HGRILF.

6.3.2. Graph component ablation

Next, we focus on the ablation study of heterogeneous graph components to demonstrate the importance of the law-fact heterogeneous graph. For that, we consider the following variations: (1) LF-HetG without keyword nodes and sentence-sentence edges (S-S edges): In this variant, we remove keyword nodes and sentence-sentence edges from the law-fact heterogeneous graph and follow the same learning procedure in LF-HGRILF. (2) LF-HetG without keyword nodes: This variant first extracts keywords to build sentence-sentence edges and then removes all keyword nodes from the graph, meaning redundant words are not filtered, and keywords corresponding to specific legal facts are not screened out. (3) LF-HetG without sentence-sentence edges: This variant excludes edges between sentences in the law-fact heterogeneous graph.

Table 6 lists the results of the ablation experiments on LF-HetG. The findings indicate that removing keyword extraction and edges between sentences from the law-fact heterogeneous graph significantly decreases model performance. Specifically, the F1 score for crime prediction and law article prediction decreased by 6.6 % and 4.5 %, respectively. When only keyword extraction or edges between sentences are removed, the F1 score for crime prediction decreases by 4.1 % and 2.9 %, respectively, and the F1 score for law article prediction decreases by 4 % and 2.5 %. These results underscore the critical role of keyword extraction in providing rich semantic information and fine-grained connections. Additionally, edges between sentences help capture the semantic information in the legal fact and facilitate the rapid integration of new cases into the legal fact heterogeneous graph.

6.4. Parameter sensitivity analysis

In this section, we conduct a comprehensive analysis to evaluate the performance of our model under various parameter configurations. We focus on the impact of the number of neighboring nodes for each node type by establishing distinct sampling ranges for different node categories and comparing the resulting outcomes. The results are illustrated in Fig. 5.

As depicted in Fig. 5(a), augmenting the quantities of neighboring nodes linked to keywords and cases result in a mountain-like trend of F1-scores for law article prediction. The scores attain the peak when the numbers of neighboring nodes for keywords and cases are set to 15 and 6, respectively. Notably, the model achieves optimal performance when the count of neighbors for law articles is 3. This suggests that expanding the number of keywords and case neighbors allows the central node to assimilate more pertinent information. However, exceeding a certain threshold introduces sampling noise nodes and increases model complexity, adversely affecting performance. Conversely, while increasing the number of law article neighbors enhances the case's capacity to accurately identify the actual law articles, it also raises the difficulty of the model's recognition process. Additionally, it was determined that the number of potentially confounding law articles should not exceed three.

Further investigation into the influence of varying hidden layer dimensions on model performance reveals that setting the hidden layer dimension to 512 yields the most favorable results, as depicted in Fig. 5(c).

In addition, we examine the effect of negative sampling ratios within the context of graph representation learning. As presented in Fig. 5(b), increasing the negative sampling ratio gradually enhances the performance of the LJP task. This indicates that negative samples effectively optimize node embeddings, increasing the spatial separation between unrelated nodes. The model achieves peak performance when the quantities of positive and negative samples are equivalent.

Finally, as shown in Fig. 5(d), when the shared word threshold of sentence edges is set to 4, the F1 scores of different subtasks demonstrate better performance, leading us to set this threshold to 4 in subsequent experiments. Additionally, we conduct a parameter search for σ_L and σ_F with testing values ranging from 0.6 to 0.9.

To analysis the thresholds of similarity, we also explore different threshold combinations and their effects on the three subtasks. The results are illustrated in Fig. 6. When the similarity threshold for law articles is 0.6 and the case similarity threshold is 0.8, all three subtasks achieve relatively high F1 scores and accuracy. Therefore, the similarity threshold between facts σ_F and the similarity threshold between law articles σ_L are set to 0.8 and 0.6, respectively.

6.5. Case study

In the legal field, a model must not only predict accurate judgment results but also explain the rule behind its predictions to convince legal professionals. Fig. 7 shows a neighbor entity graph of a legal case, which

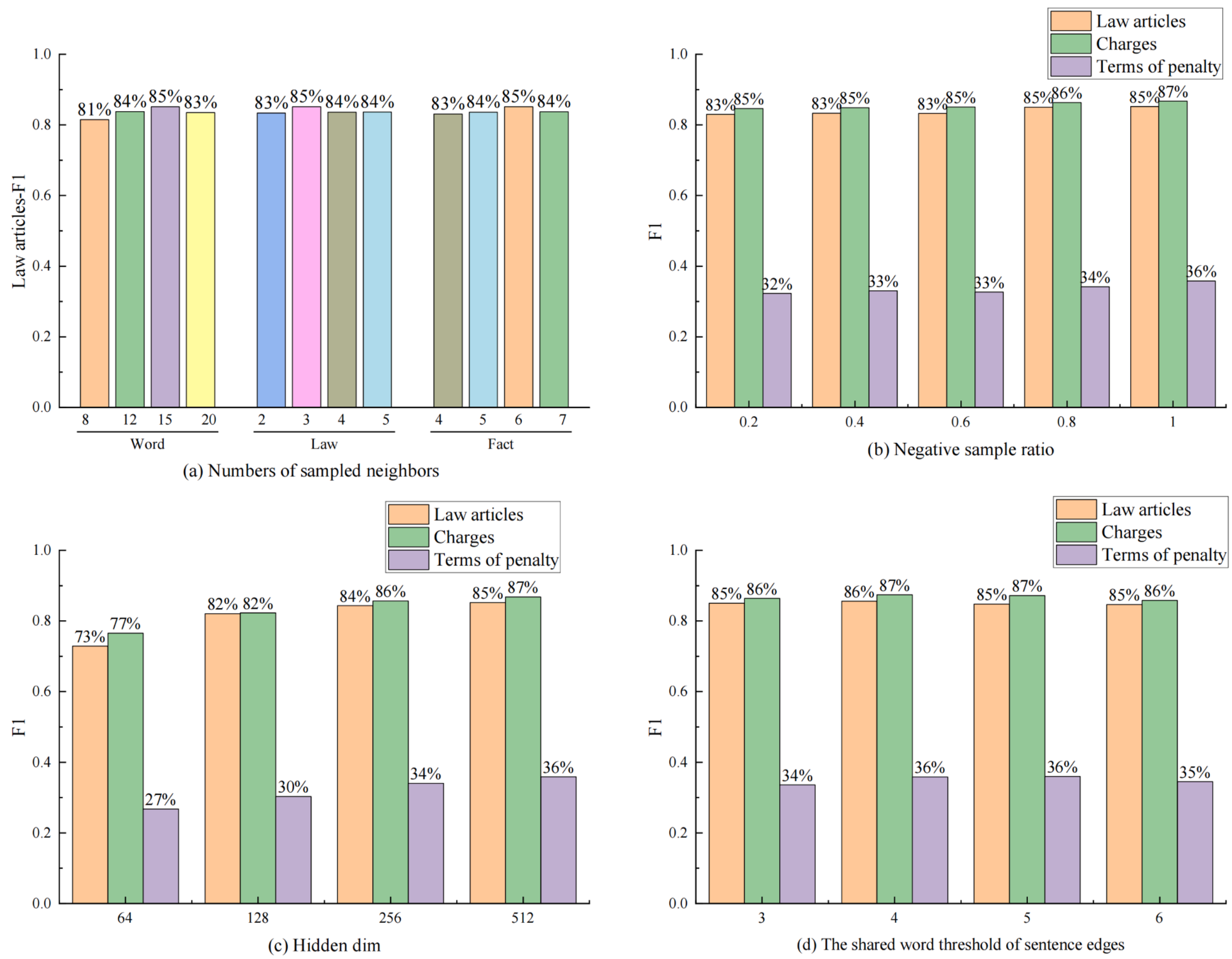


Fig. 5. Experiments and performances with different model parameters.

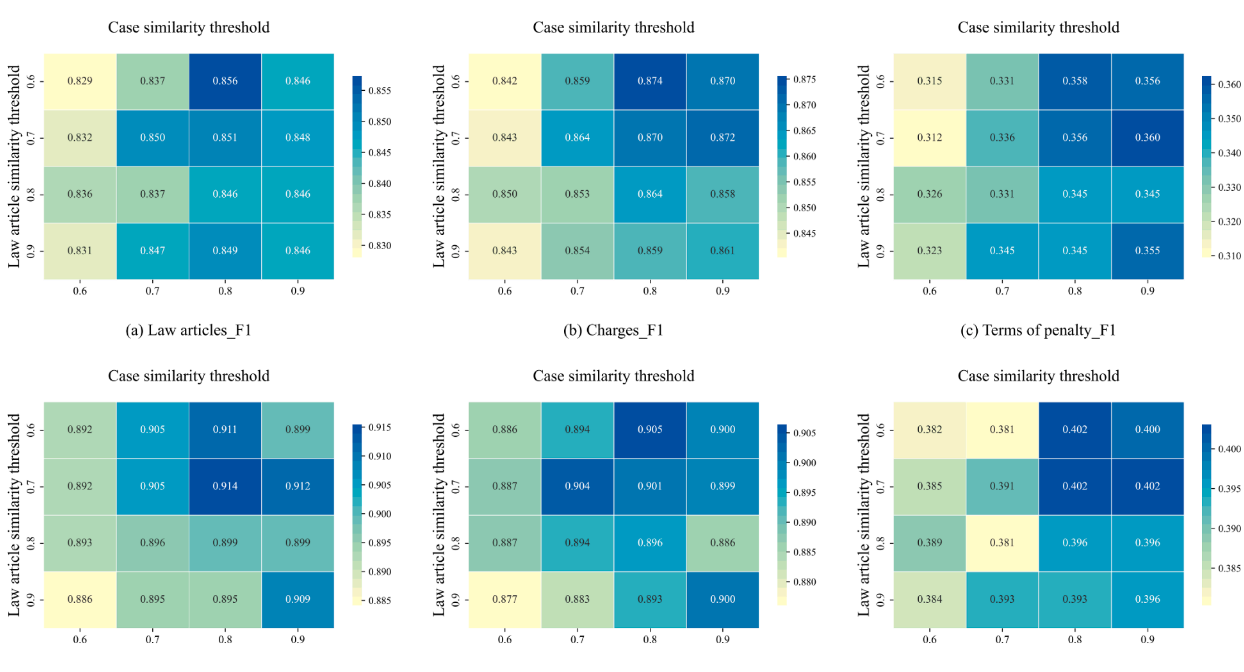


Fig. 6. Heatmaps of different similarity thresholds of law article prediction, charge prediction, and penalty term prediction. The horizontal axis denotes the case similarity threshold, while the vertical axis indicates the law article similarity threshold. A grid search strategy is adopted to identify the optimal threshold combinations. The first row shows the F1 results and The second shows the accuracy results..

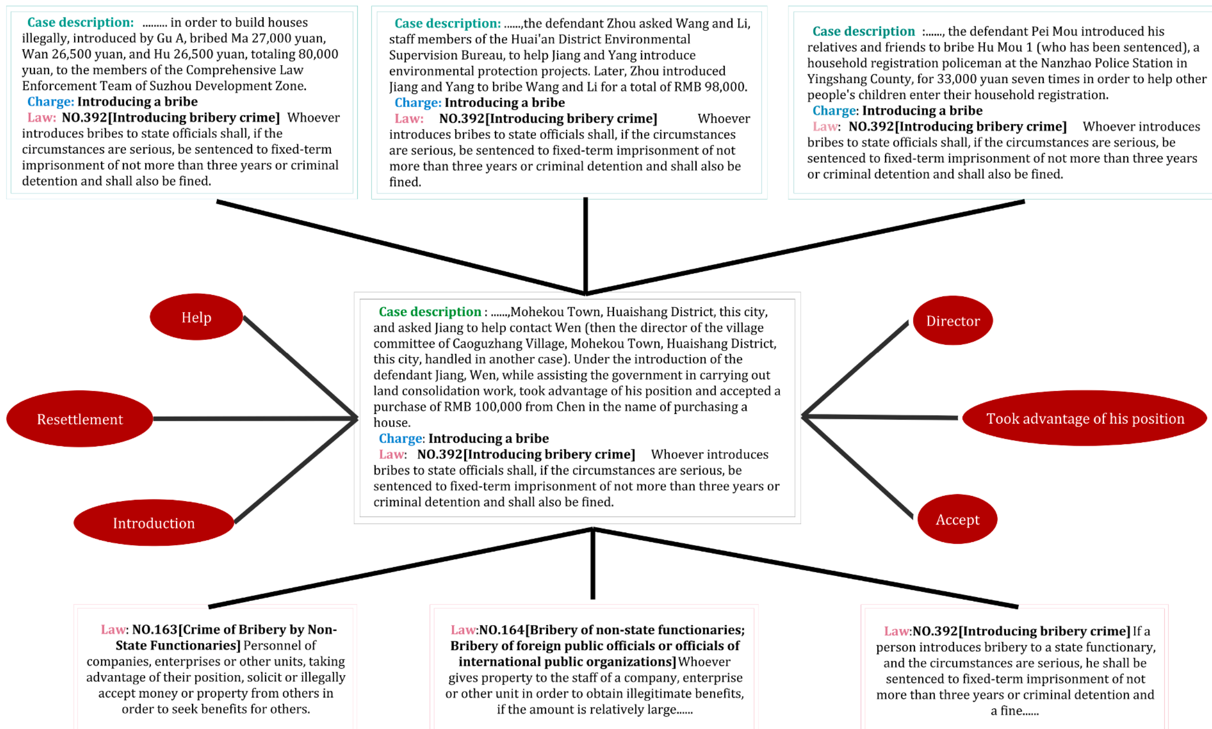


Fig. 7. An example neighbor graph for a case: the red ellipses stand for keywords, the pink boxes stand for the law articles that it refers to, and the green boxes stand for its similar legal cases.

is a sample subgraph obtained through the type proximity random walk sampling in Section 5.1.1. The red nodes represent keywords, while other nodes represent legal cases and law articles that are similar to the current case in law articles or facts. This visualization demonstrates how the model accurately identifies legal entities related to the target case through the meta-path setting and node sampling strategy. For example, in cases involving the crime of introducing bribery, the charges in the target case are consistent with neighboring cases, all related to introducing bribery. Keywords such as "introducing" and "bribery" frequently appear in these cases, highlighting their common characteristics. Additionally, neighboring law articles, such as "bribery of non-state functionaries," "bribery to non-state functionaries," and "bribery to foreign public officials and officials of international public organizations," show high similarity to the law articles related to introducing bribery. This detailed association analysis not only helps legal professionals understand the legal basis of the cases deeply but also enhances the transparency and interpretability of the model's predictions.

In Fig. 8, we give the visualization in terms of the attention weight of the three confusing law articles (, 168, 392). For the case of the 393rd law article, the keywords "unit" and "bribed" carry the highest attention weights. This indicates that LF-HGRILF has identified the defendant as an organization, aligning with the subject requirements for the crime of bribery by units, and captures the act of bribery, a key element of this crime. Meanwhile, "direct supervisor" and "general manager" define the recipients of the bribe, i.e., state officials, while other keywords reveal the crime's motive. In the case under the 163rd law article, the keywords "took advantage of his position" and "request for property" carry the highest attention weights. This shows that LF-HGRILF accurately identified the defendant using their position to solicit property from the project manager, aligning with the elements of the crime of bribery by non-state personnel. "The person in charge", "Project Department", and "name" provide case background and involved personnel identities, while other keywords pertain to the specific forms and values of the bribes. In the case under the 392nd law article, the keyword "introduction" highlights the key part of the crime of introducing bribery, which involves facilitating contact between the briber and the recipient. "Took advantage of his position" indicates the recipient used their position to commit the crime. These keywords are essential elements of introducing bribery, while other keywords describe the case background and details of the criminal behavior.

To further illustrate the effectiveness of the law article distinction module, we visualize the distribution in the representation space of two groups of confusing law articles in different iterations via t-Distributed Stochastic Neighbor Embedding technology (t-SNE). The visualization results are shown in Fig. 9. In the first iteration, we obtain the representation of Articles 392, 393, and 163 (in blue). The 393rd and 163rd articles are close in the representation space, while the 392nd article maintains a distance from them. As the iteration grows, these law articles exhibit a clear trend of separation. For Articles 192, 193, and 194 (in red), the separation trend is more obvious. It shows that the iteration framework of LF-HGRILF contributes to continuously refining the representations of law articles and enhancing the distinction of confusing law articles.

7. Conclusion

In this study, we proposed the Law-Fact Heterogeneous Graph Representation Iterative Learning Framework (LF-HGRILF) to improve the accuracy and interpretability of legal judgment prediction. Our approach constructs a heterogeneous graph that incorporates legal facts, sentences, keywords, and law articles, enabling the model to better capture semantic nuances and structural dependencies. To address the ambiguity caused by similar law articles, we designed a law article distinction module based on reverse attention. Furthermore, we introduced an iterative graph learning strategy to enhance information propagation and representation refinement.

We evaluated our model on the large-scale CAIL 2018 dataset, which contains over 1.5 million real-world criminal cases with annotated judgment results. Experimental results demonstrate that our method outperforms conventional baselines and shows improved performance

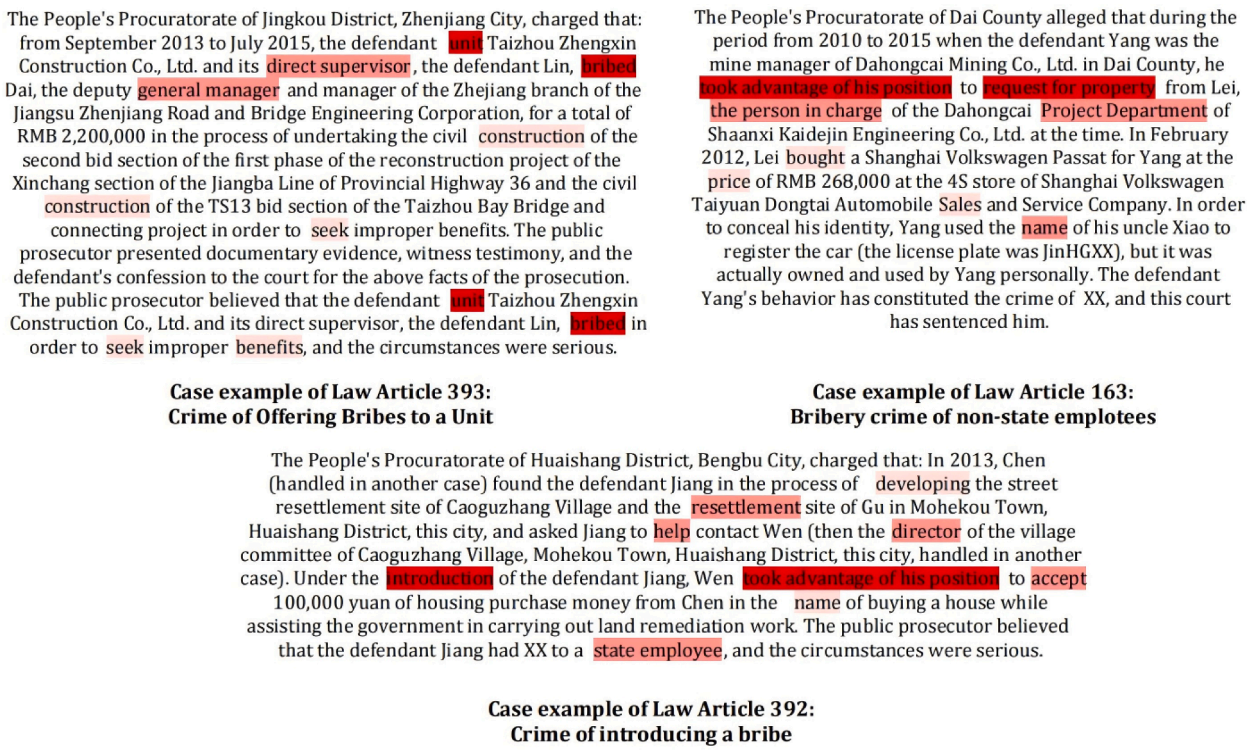


Fig. 8. Attention visualization of legal cases in terms of 393, 163, and 392.

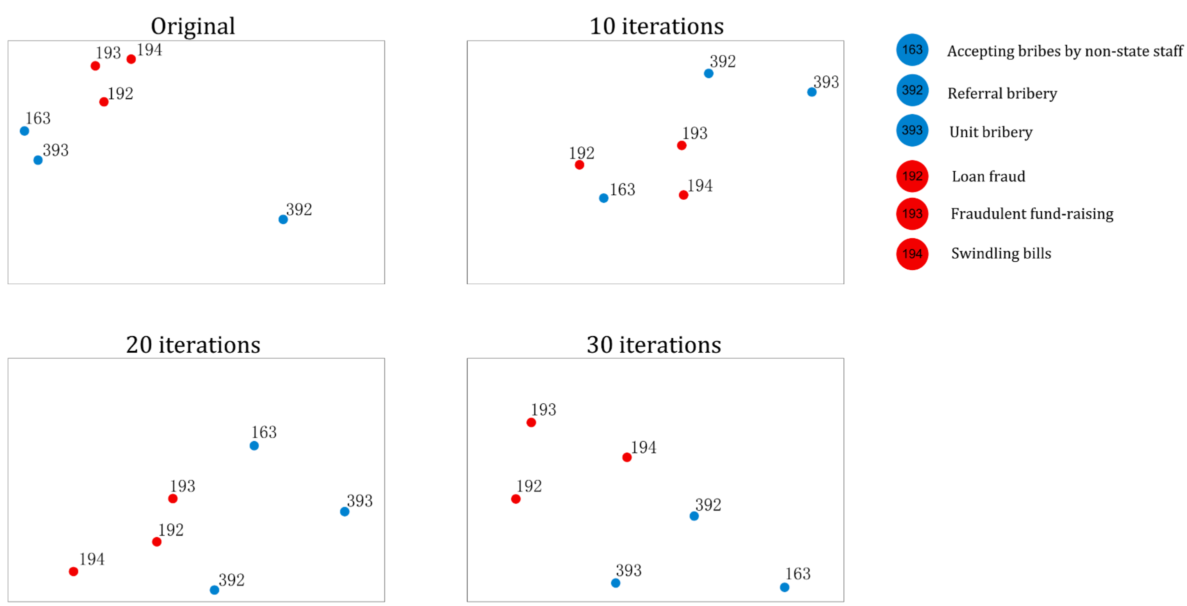


Fig. 9. Spatial visualization of a part of law article representations in different iterations via t-SNE.

in handling challenges such as law article similarity and penalty term prediction. We also visualized intermediate graph representations to enhance the interpretability of our framework.

In future work, we plan to explore more expressive graph structures by incorporating continuous variables such as the specific penalty duration and integrating external legal knowledge. This would further strengthen the model's ability to capture fine-grained legal relationships and improve its generalization in diverse legal scenarios.

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used ChatGLM to improve the readability and language of the manuscript. After using this tool/service, the authors reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

CRediT authorship contribution statement

Yinying Kong: Writing – review & editing, Supervision, Methodology, Formal analysis, Conceptualization; **You-Gan Wang:** Writing – review & editing, Supervision, Methodology, Formal analysis, Conceptualization; **Haodong Deng:** Writing – original draft, Methodology, Visualization, Validation, Software, Resources, Data curation; **Zhanhao Xiao:** Writing – original draft, Methodology, Visualization, Investigation, Formal analysis; **Yuke Zhang:** Validation, Investigation, Data curation.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61906216), the Guangdong Provincial Department of Education Innovation Team Project (2022WCXTD009), Guangdong Provincial Key Laboratory Project of Intellectual Property and Big Data (2018B030322016), Guangdong Provincial Department of Education Scientific Research Project-Key Field Special Project (2024ZDZX2088), Guangdong Province Graduate Education Innovation Plan Project (2024SFKC_042 and 2024ANLK_067), Hunan Province Science and Technology Innovation Program (2022RC4028), Guangzhou Science and Technology Program (2024A04J4676), and Guangdong Polytechnic Normal University Project (22GPNUZDJS19).

References

[1] Z. Hu, X. Li, C. Tu, Z. Liu, M. Sun, Few-shot charge prediction with discriminative legal attributes, in: Proceedings of the 27th International Conference on Computational Linguistics, 2018, pp. 487–498.
[2] N. Xu, P. Wang, L. Chen, L. Pan, X. Wang, J. Zhao, Distinguish confusing law articles for legal judgment prediction, arXiv preprint arXiv:2004.02557 (2020).
[3] S. Bi, Y. Huang, X. Cheng, M. Wang, G. Qi, 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 12, Springer, 2019, pp. 628–639.
[4] S. Bi, Z. Ali, M. Wang, T. Wu, G. Qi, Learning heterogeneous graph embedding for chinese legal document similarity, Knowl. Based Syst. 250 (2022) 109046.
[5] F. Kort, Predicting supreme court decisions mathematically: a quantitative analysis of the “right to counsel” cases, Am. Polit. Sci. Rev. 51(1) (1957) 1–12.
[6] S.S. Nagel, Applying correlation analysis to case prediction, Tex. L. Rev. 42 (1963) 1006.
[7] J.A. Segal, Predicting supreme court cases probabilistically: the search and seizure cases, 1962–1981, Am. Polit. Sci. Rev. 78(4) (1984) 891–900.
[8] B.E. Lauderdale, T.S. Clark, The supreme court’s many median justices, Am. Polit. Sci. Rev. 106(4) (2012) 847–866.
[9] H. Wang, T. He, Z. Zou, S. Shen, Y. Li, Using case facts to predict accusation based on deep learning, in: 2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C), IEEE, 2019, pp. 133–137.
[10] X. Jiang, H. Ye, Z. Luo, W. Chao, W. Ma, Interpretable rationale augmented charge prediction system, in: Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations, 2018, pp. 146–151.
[11] H. Chen, D. Cai, W. Dai, Z. Dai, Y. Ding, Charge-based prison term prediction with deep gating network, arXiv preprint arXiv:1908.11521 (2019).
[12] W. Yang, W. Jia, X. Zhou, Y. Luo, Legal judgment prediction via multi-perspective bi-feedback network, arXiv preprint arXiv:1905.03969 (2019).
[13] Y. Feng, C. Li, V. Ng, Legal judgment prediction via event extraction with constraints, in: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022, pp. 648–664.
[14] Z. Xu, X. Li, Y. Li, Z. Wang, Y. Fanxu, X. Lai, Multi-task legal judgement prediction combining a subtask of the seriousness of charges, in: Chinese Computational Linguistics: 19th China National Conference, CCL 2020, Hainan, China, October 30–November 1, 2020, Proceedings 19, Springer, 2020, pp. 415–429.
[15] B. Luo, Y. Feng, J. Xu, X. Zhang, D. Zhao, Learning to predict charges for criminal cases with legal basis, arXiv preprint arXiv:1707.09168 (2017).
[16] S. Pan, T. Lu, N. Gu, H. Zhang, C. Xu, Charge prediction for multi-defendant cases with multi-scale attention, in: Computer Supported Cooperative Work and Social Computing: 14th CCF Conference, ChineseCSCW 2019, Kunming, China, August 16–18, 2019, Revised Selected Papers 14, Springer, 2019, pp. 766–777.
[17] S. Li, H. Zhang, L. Ye, X. Guo, B. Fang, Mann: a multichannel attentive neural network for legal judgment prediction, IEEE Access 7 (2019) 151144–151155.
[18] L. Gan, K. Kuang, Y. Yang, F. Wu, Judgment prediction via injecting legal knowledge into neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, 2021, pp. 12866–12874.
[19] Y. Le, Z. Quan, J. Wang, D. Cao, K. Li, Boldsymbol : a novel recall & ranking framework for legal judgment prediction, IEEE/ACM Trans. Audio Speech Lang. Process. (2024).
[20] H. Zhang, Z. Dou, J.R. Zhu, Contrastive learning for legal judgment prediction, ACM Trans. Inf. Syst. 41(4) (2023) 1.1–1.25.
[21] Z. Liu, J. Li, Y. Huang, N. Cui, L. Pei, Knowledge-based natural answer generation via effective graph learning, Knowl. Based Syst. 316 (2025) 113288.
[22] J. Li, K. Deng, J. Li, Y. Ren, Session-oriented fairness-aware recommendation via dual temporal convolutional networks, IEEE Trans. Knowl. Data Eng. 37(2) (2025) 923–935. https://doi.org/10.1109/TKDE.2024.3509454
[23] Z. Li, X. Wang, J. Zhao, W. Guo, J. Li, Hycube: efficient knowledge hypergraph 3d circular convolutional embedding, IEEE Trans. Knowl. Data Eng. 37(4) (2025) 1902–1914. https://doi.org/10.1109/TKDE.2025.3531372
[24] D. Zeng, E. Zha, J. Kuang, Y. Shen, Multi-label text classification based on semantic-sensitive graph convolutional network, Knowl. Based Syst. 284 (2024) 111303.
[25] W. Ai, Y. Wei, H. Shao, Y. Shou, T. Meng, K. Li, Edge-enhanced minimum-margin graph attention network for short text classification, Expert Syst. Appl. 251 (2024) 124069.
[26] H. Liu, X. Huang, X. Liu, Improve label embedding quality through global sensitive GAT for hierarchical text classification, Expert Syst. Appl. 238 (2024) 122267.
[27] H. Zhong, Z. Guo, C. Tu, C. Xiao, Z. Liu, M. Sun, Legal judgment prediction via topological learning, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 3540–3549.
[28] Q. Dong, S. Niu, Legal judgment prediction via relational learning, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 983–992.
[29] J. Wang, Y. Le, D. Cao, S. Lu, Z. Quan, M. Wang, Graph reasoning with supervised contrastive learning for legal judgment prediction, IEEE Trans. Neural Netw. Learn. Syst. (2024).
[30] P. Bhattacharya, K. Ghosh, A. Pal, S. Ghosh, Hier-spcnet: a legal statute hierarchy-based heterogeneous network for computing legal case document similarity, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1657–1660.
[31] Y. Wang, J. Gao, J. Chen, Z. Shen, H. Wang, Case recommendation algorithm of discipline inspection and supervision based on knowledge graph, in: 2021 2nd International Conference on Electronics, Communications and Information Technology (CECIT), IEEE, 2021, pp. 336–342.
[32] S. Yan, Z. Zhang, Skyper: legal case retrieval via skeleton-aware hypergraph embedding in the hyperbolic space, Inf. Sci. 682 (2024) 121162.
[33] Y. Tang, R. Qiu, H. Yin, X. Li, Z. Huang, Caselink: inductive graph learning for legal case retrieval, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024, pp. 2199–2209.
[34] J. Mo, J. Zheng, D. Bi, Establishment of a copyright regulations knowledge base and development of a case recommendation system utilizing transformer and graph convolutional network, IEEE Access (2024).
[35] P. Bhattacharya, K. Ghosh, A. Pal, S. Ghosh, Legal case document similarity: you need both network and text, Inf. Process. Manag. 59(6) (2022) 103069.
[36] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Inf. Process. Manag. 24(5) (1988) 513–523.
[37] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, Adv. Neural Inf. Process. Syst. 30 (2017).
[38] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, arXiv preprint arXiv:1710.10903 (2017).
[39] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907 (2016).
[40] C. Zhang, D. Song, C. Huang, A. Swami, N.V. Chawla, Heterogeneous graph neural network, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 793–803.
[41] Y. Dong, N.V. Chawla, A. Swami, Metapath2vec: scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 135–144.
[42] J. Pennington, R. Socher, C.D. Manning, Glove: global vectors for word representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543.
[43] J. Lu, H. Zhang, Z. Shen, K. Shi, L. Yang, B. Xu, S. Zhang, H. Lin, Multi-task hierarchical cross-attention network for multi-label text classification, in: CCF International Conference on Natural Language Processing and Chinese Computing, Springer, 2022, pp. 156–167.
[44] Q. Zhao, T. Gao, N. Guo, La-mgfm: a legal judgment prediction method via sememe-enhanced graph neural networks and multi-graph fusion mechanism, Inf. Process. Manag. 60(5) (2023) 103455.
[45] Q. Zhao, T. Gao, S. Zhou, D. Li, Y. Wen, Legal judgment prediction via heterogeneous graphs and knowledge of law articles, Appl. Sci. 12(5) (2022) 2531.
[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).

[47] M. Kamyab, G. Liu, M. Adjeisah, Attention-based CNN and bi-LSTM model based on TF-IDF and glove word embedding for sentiment analysis, Appl. Sci. 11(23) (2021) 11255.

[48] C. Xiao, H. Zhong, Z. Guo, C. Tu, Z. Liu, M. Sun, Y. Feng, X. Han, Z. Hu, H. Wang, et al., Cail2018: a large-scale legal dataset for judgment prediction, arXiv preprint arXiv:1807.02478 (2018).

[49] L. Yue, Q. Liu, B. Jin, H. Wu, K. Zhang, Y. An, M. Cheng, B. Yin, D. Wu, Neurjudge: a circumstance-aware neural framework for legal judgment prediction, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 973–982.

[50] J.A.K. Suykens, L. Lukas, P. Van Dooren, B. De Moor, J. Vandewalle, et al., Least squares support vector machine classifiers: a large scale algorithm, in: European Conference on Circuit Theory and Design, ECCTD, volume 99, Citeseer, 1999, pp. 839–842.

[51] Y. Chen, Convolutional Neural Network for Sentence Classification, Master's thesis, University of Waterloo, 2015.

[52] S. Lai, L. Xu, K. Liu, J. Zhao, Recurrent convolutional neural networks for text classification, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 29, 2015.

[53] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, E. Hovy, Hierarchical attention nNetworks for document classification, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2016, pp. 1480–1489.

[54] L. Yao, C. Mao, Y. Luo, Graph convolutional networks for text classification, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, 2019, pp. 7370–7377.

[55] H. Zhang, Z. Dou, Y. Zhu, J.-R. Wen, Contrastive learning for legal judgment prediction, ACM Trans. Inf. Syst. 41(4) (2023) 1–25.

[56] Q. Zhao, Legal judgment prediction via legal knowledge extraction and fusion, J. King Saud Univer. Comput. Inf. Sci. 37(3) (2025) 1–16.

[57] D.P. Kingma, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).