Structural Patent Classification Using Label Hierarchy Optimization

Mengting Gui[1]*, Shufeng Hao[2] * , Chongyang Shi[1] † , Qi Zhang[3]
[1]School of Computer Science and Technology, Beijing Institute of Technology
[2]College of Computer Science and Technology (College of Data Science),
Taiyuan University of Technology
[3]Department of Computer Science and Technology, Tongji University
{mt_gui ,cy_shi}@bit.edu.cn, haoshufeng@tyut.edu.cn, zhangqi_cs@tongji.edu.cn

Abstract

Patent classification is a fundamental step in the patent examination process, directly impacting the efficiency and quality of substantive review. Existing methods mostly focus on general texts like titles and abstracts, thus ignoring the key technical content claims and the corresponding citation relationships. Meanwhile, these approaches treat labels as independent targets, failing to exploit the semantic and structural information within the label taxonomy. To address these problems, we propose a **C**laim **S**tructure based **P**atent **C**lassification model with **L**abel **A**wareness (CSPC-LA). The method first utilizes the citation relationship of patent claim texts to construct the citation graph and the co-reference graph. Then structural graph learning is used on both graphs to mine the internal logic of patent claims. Finally, we optimize the tree hierarchy of IPC labels and employ tree propagation learning to enhance the patent representation. Extensive experiments on the latest patent classification dataset from USPTO demonstrate that the proposed method is more effective than the state-of-the-art baselines.

1 Introduction

With the rapid advancement of globalization and informatization, innovation has become a key driving force behind social progress and economic growth. As emerging technologies continue to flourish and industrial upgrading accelerates, the number of patent applications, an important indicator of innovation, has been increasing globally. Although this trend reflects the vitality of technological development, it also places significant pressure on patent examination systems. Among these processes, patent classification serves as a critical first step (United States Patent and Trademark Office, 2022), whose accuracy and efficiency are crucial.

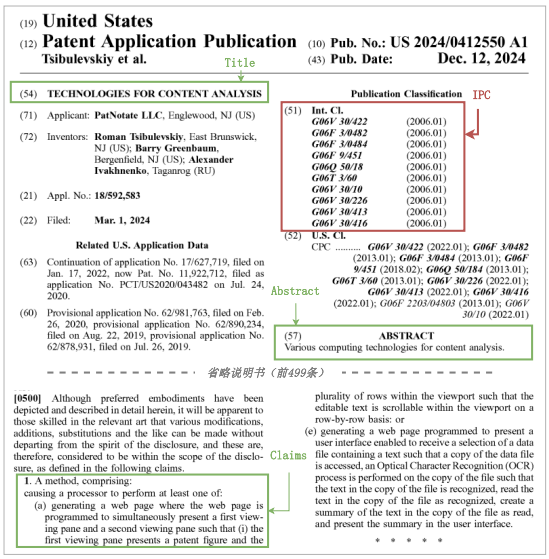


Figure 1: An overview of patent document

Patent document classification is inherently a complex multi-label text classification task. Although deep learning has achieved remarkable progress in this field, existing methods mostly rely on short texts such as titles and abstracts, ignoring the fine-grained semantic information embedded in the claims section (Suzuki and Takatsuka, 2016). Meanwhile, the structural complexity, cross-domain nature, and rich technical content in patent documents pose challenges for patent detection. As shown in Figure 1, patents have rich metadata (Lupu et al., 2013), mainly consisting of structured information (such as application number and inventor) and unstructured text (including title, abstract, description, and claims). The title and abstract briefly describe the invention, and the description provides detailed background and implementation methods, while the claims clearly define the legal scope of protection. Claims are essential for determining patent infringement and evaluating novelty. Claims include independent claims and de-

*Equal contribution.
†Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 100–114
November 4-9, 2025

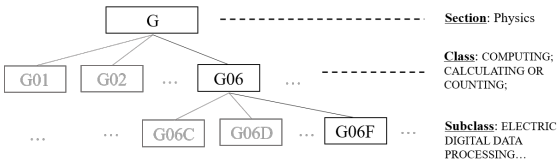


Figure 2: Examples of IPC Layers

pendent claims. Independent claims describe the complete scope of protection through a preamble and a feature part, and dependent claims further refine technical features based on the independent claims, forming a multi-level protection system. Furthermore, the large number of classes in patent taxonomies (WIPO, 2024b), their highly imbalanced distribution, and the lack of publicly available datasets containing claim texts and citation information further hinder the development of effective patent classification models.

Current patent classification systems, such as ECLA, USPC, FI/F-Term, are independently developed and used for patent examination and classification within their respective countries. Thus, we select the international patent classification IPC system (WIPO, 2024c) as our study object. The internationally adopted system features a hierarchical tree structure. It divides all scientific and technical fields into eight sections based on the basic function and application of the invention. As shown in Figure 2, each section is further subdivided into classes and subclasses, with codes becoming progressively detailed, providing increasingly fine-grained domain information. The large number of categories poses a challenge to patent classification.

To address these challenges, we first construct a novel patent classification dataset based on the IPC system from USPTO's 2024 patent application data. Each sample in the dataset not only contains titles and abstracts, but also includes complete claim texts and citation information. Then, we propose a claim structure and label aware patent classification model. The method first utilizes the citation relationship of patent claim texts to construct the citation graph and the co-reference graph. Then, patent structural graph learning is used on both graph structures to mine the internal logic of patent claims. Finally, we optimize the hierarchical tree of IPC labels and employ propagation tree learning to enhance the patent representation. Extensive experimental results demonstrate the effectiveness of the proposed method on the novel constructed patent classification dataset. The work contains three contributions:

- We propose a novel method that models the citation structure among patent claims, enabling the model to explicitly capture the structural relationships and better represent the technical content of patents;
- We construct a hierarchical tree of IPC labels and incorporate structural priors to model inter-class dependencies, thereby alleviating the difficulties caused by large-scale and imbalanced label distributions;
- We build and release a real-world challenging patent classification dataset based on USPTO's 2024 patent application data, which includes complete claim texts and citation information, providing a valuable resource for future research.

2 Related Work

Patent Classification. Existing patent classification approaches can be broadly categorized into traditional methods and deep learning-based methods. Early traditional methods treat patents as plain text and apply rule-based or classical machine learning techniques using handcrafted features such as keyword frequency and bag-of-words models (Larkey, 1999; Fall et al., 2003). With the advent of deep learning, researchers began using convolutional neural networks (CNN) and pre-trained language models to enhance patent text representations and classification performance (Li et al., 2018; Lee and Hsiang, 2019; Srebrovic and Yonamine, 2020; Yang et al., 2019; Haghighian Roudsari et al., 2022). Several approaches aim to overcome the limitations of standard attention mechanisms, especially the computational and memory overhead in capturing long-range dependencies, which is consistent with the problems encountered in patent classification (Zaheer et al., 2020).

To better capture the structural characteristics of patent documents, some methods focus on the use of patent text and labels. Risch proposed PatentMatch(Risch et al., 2020a), containing claim pairs in patent applications, laying the foundation for studying patent classification. Risch Pujari first proposed TMM (Pujari et al., 2021), combining Transformer and hierarchical algorithms for patent classification. Recent studies have explored incorporating both intra-patent texts and inter-patent relationships (Shalaby et al., 2018). Some works

utilize citation networks, metadata, and multiview graph structures to model cross-patent relationships (Li et al., 2007; Zhu et al., 2015; Fang et al., 2021; Hamid Bekamiri and Jurowetzki, 2024). Others combine label hierarchies with contrastive learning to learn fine-grained instance-label associations (Risch et al., 2020b; Li et al., 2022; Liu et al., 2024; Pujari et al., 2022).

Although these methods have advanced the modeling of structural and relational information in patent classification, limited attention has been paid to the internal citation relations among patent claims, which are crucial for capturing the logical and semantic dependencies within patents.

Hierarchical Multi-label Text Classification. Considering the hierarchical nature of patent taxonomies, patent classification can also be formulated as a hierarchical multi-label text classification (HMTC) problem. Based on how label hierarchies are handled, HMTC methods can be classified into local, global, and hybrid approaches.

Local approaches include Local Classifier per Node (LCN), Local Classifier per Parent Node (LCPN), and Local Classifier per Level (LCL). LCN assigns a binary classifier to each node, leading to high parameter complexity. LCPN constructs classifiers for each parent node, capturing hierarchical dependencies. LCL trains a classifier per level to achieve better granularity. However, local methods often suffer from error propagation.

Global approaches treat hierarchical classification as a flat multi-label task, modeling all categories in a unified framework. Some adopt hierarchical label encoders to capture inter-label dependencies (Zhou et al., 2020; Risch et al., 2020b). Others enhance textual representations with syntactic cues derived from the label hierarchy (Zhu et al., 2023).

Hybrid approaches combine the fine-grained modeling of local methods with the comprehensive view of global ones. Peng(Peng et al., 2018) proposed a deep learning model based on a graph CNN, which uses graph convolution operations to convolve the word graphs. Representative techniques include hierarchical attention-based RNNs coupled with global classifiers (Huang et al., 2019), multi-granularity document representations (Jiang et al., 2019), and joint modeling of global-local hierarchical features (Zhang et al., 2022).

Despite these efforts, few studies have jointly modeled the structural taxonomy of patent labels with intra-claim citation relationships, limiting the model's ability to capture both semantic and hierarchical dependencies in patent classification.

3 Method

3.1 Problem Definition

Patent classification Task could be formulated as a hierarchical multi-label text classification task, which focuses on the classification results of the subclass labels. For the i-th patent S_i, its text D_i consists of three parts: the title T_i, abstract A_i, and the set of claims $C_i = \{c_j\}_{j=1}^{n}$, which are used for further processing like constructing citation graphs. The label hierarchy is defined as $T = (Y, E, H)$, where $Y = \{Y^1, Y^2, Y^3\}$ denotes nodes at different label levels, with Y^1, Y^2 and Y^3 denote the section labels, the class labels and the subclass labels, repectively; E indicates edges among labels, and H contains textual representations of each label node. Given the patent text D_i and the label hierarchy T, the goal of our model is to predict a set of subclass labels $Y_i = \{y_i \mid y_i \in Y^3\}$, with $|Y_i| \geq 1$. The model aims to learn a classifier f that maps the patent text D_i and label hierarchy T to the corresponding label set, $f(D_i, T) \rightarrow Y_i$.

The goal of this work is to leverage information from the claim structure and the label hierarchy to support patent classification. The overall framework is shown in Figure 3. The method contains two modules: structural text representation and label aware representation learning. The structural text representation module explores internal textual information by mining structural relationships through claims' citations, and the label aware representation learning module leverages the hierarchical and semantic structure of labels to enrich patent representations.

3.2 Structural Text Representation

Claim Selection. In the drafting of patents, applicants often elaborate on independent claims through multiple dependent claims to broaden the scope of protection. While this strategy enhances legal defensibility, it also results in an excessive number of claims. Some claims are short and have low information density, often containing a lot of templated or general language, which can introduce irrelevant patterns into the text representation model, increase representation noise and thus introduce redundancy and pose challenges for direct modeling. What's more, during application, they often need to modify claims, like deleting or

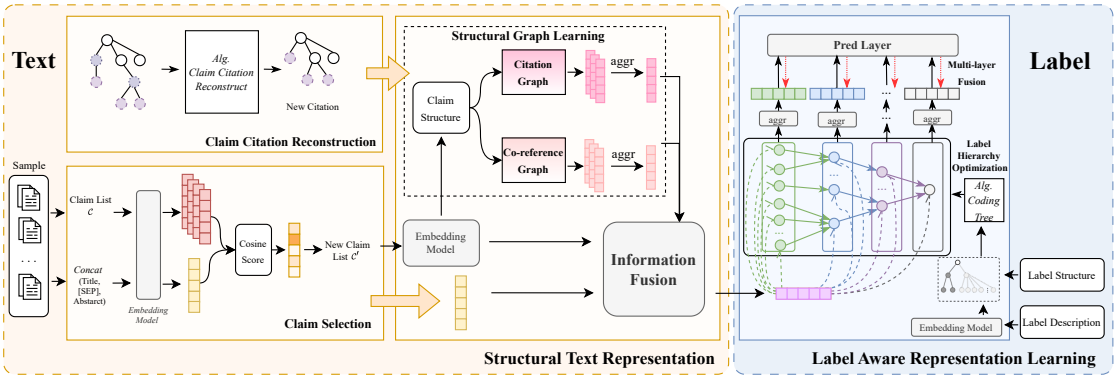


Figure 3: The overall framework of CSPC-LA. The module contains structural text representation and label aware representation learning.

changing, which results in invalid information in the claims. At the same time, patent titles and abstracts are typically written by professionals and offer high information density and summarization quality. To improve claims quality, we propose a claim selection method guided by the title and abstract. Specifically, we concatenate the title and abstract into a joint query, encode both the query and each claim by using BERT-for-patents, and compute cosine similarity to select the top K most relevant claims. This selection process helps retain the most topic-relevant content, providing more focused input for subsequent structural modeling.

Claim Citation Reconstruction. During the previous stage, a representative subset of claims was selected, significantly reducing the overall length of the claim list. However, this selection may result in the disconnection of certain citation links between claims. Consequently, some claims become isolated during the subsequent graph construction process, limiting their contribution to the overall semantic representation of the patent. To address this, we reconstruct the missing citation links using both the original claim citation graph and the selected claim subset. Consider the new claims list is $\mathcal{C}'$. Specifically, for each claim c, we locate its referenced claim c_1 in the original citation graph. If c_1 is not included in $\mathcal{C}'$, we recursively trace upward to find the next-level citation $f(c_1)$, and so on, until either a referenced claim within the list is found or the root node is reached. The citation target of c is then updated accordingly.

$$f(c) = \begin{cases} c_1, & \text{if } c_1 \in \mathcal{C}' \\ f(c_1), & \text{if } c_1 \notin \mathcal{C}' \text{ and } c_1 \neq \text{Root} \\ \text{Root}, & \text{otherwise} \end{cases} \quad (1)$$

After reconstructing the citation relationships, a set of claims that cite the same parent claim is referred to as a *co-reference dependency group*. These claims elaborate on the same subject from different perspectives, thereby complementing its various features. Based on the reconstructed citation structure, co-reference dependencies are established by connecting all child claims that share the same parent. Formally, for any pair of claims i and j, if they both cite the same parent claim, an edge is added between them in the adjacency matrix. That is, if $B_{ij} = 1$, then claim i and j both reference the same parent claim.

Consequently, two types of adjacency matrices are obtained through this relation reconstruction: $\mathbf{P}$ for the citation graph, and $\mathbf{B}$ for the co-reference graph.

Structural Graph Learning. With the citation structure reconstructed, we obtain a complete set of interconnected claims, enabling structural modeling based on claim relationships. To effectively capture the internal structure of the claims section, we construct a graph that incorporates both hierarchical and parallel semantic relationships. Specifically, *citation relations* represent the semantic progression from parent to child claims, reflecting the hierarchical refinement of patent protection. In contrast, *co-reference relations* (claims sharing the same cited parent) capture semantic diversity across parallel claims that elaborate on different aspects of a common theme. Based on these insights, we construct two separate graphs: a *citation graph* and a *co-reference graph*. Each is modeled independently using a dedicated Graph Attention Network (GAT). The citation graph employs a bottom-up attention aggregation mecha-

nism to propagate semantic refinements, while the co-reference graph emphasizes lateral information fusion and contrastive semantic modeling among sibling claims.

To ensure that isolated nodes can retain their own features during representation learning, we add self-loops to the adjacency matrix as follows:

$$\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I} \tag{2}$$

Each graph structure (i.e.citation or co-reference) is processed by L stacked GAT layers to capture multi-hop structural dependencies:

$$\mathbf{h}^{(l)} = \text{GATLayer}(\mathbf{h}^{(l-1)}, \tilde{\mathbf{A}}), \quad \mathbf{h}^{(0)} = \mathbf{X} \tag{3}$$

Then, we incorporate residual connections and layer normalization to enhance model stability:

$$\mathbf{h}^{(l)} = \text{LayerNorm}(\tilde{\mathbf{h}}^{(l)} + \mathbf{h}^{(l-1)}) \tag{4}$$

Finally, the final representation is obtained by average pooling over all nodes:

$$\mathbf{h} = \frac{1}{N_l} \sum_{i=1}^{N_l} \mathbf{h}_i^{(l)} \tag{5}$$

Information Fusion. The mentioned structured representations of patents focuses on modeling fine-grained claims text structures, while the granularity of information covered by different patent text components varies: titles and abstracts cover more general information, which is conducive to classification at a higher level; The claims specify the object of the patent and its feature set, which helps to focus on the fine-grained features of the patent. In order to enable the model to adaptively focus on different parts of information in classification tasks, an adaptive information fusion module is used to fuse the representations of various parts of the patent, defined as:

$$\mathbf{h} = \text{Concat}(\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3, \mathbf{h}_4) \tag{6}$$

where h_1 and h_2 represent the text information of patents, such as title and abstract; moreover, h_3 and h_4 represent the structure information from claims citations, as the citation relationship and the co-reference relationship. Then, the enhanced representation is obtained by fusing the above different information:

$$\mathbf{h}' = \sum_{i=1}^{4} w_i \cdot \mathbf{h}_i \tag{7}$$

$$w_i = \text{Softmax}(\mathbf{W}_i \mathbf{h_i} + \mathbf{b}_i) \tag{8}$$

where $\mathbf{W}_i$ and $\mathbf{b}_i$ are the learnable weight and bias parameters, respectively.

3.3 Label Aware Representation Learning

Label Hierarchy Learning. Starting from the existing IPC structure, its hierarchical system poses certain limitations for model learning. First, the sample distribution is highly imbalanced, with some classes having abundant samples while many tail classes contain only a few, which adversely affects the model's generalization ability. Second, the label structure itself exhibits imbalance characteristics: some parent classes aggregate a large number of child classes, forming dense paths, whereas others are relatively sparse. This structural disparity leads to two main issues: (1) tail classes suffer from sparse paths, making them easily obscured within the hierarchy; (2) head paths are overly reinforced, causing structural bias that restricts the model's ability to capture fine-grained semantic relations among labels. Therefore, it is necessary to reconstruct and optimize the IPC hierarchy to alleviate the negative effects caused by distribution imbalance. Inspired by previous work (Zhu et al., 2023), we employ structural entropy (Cheng et al., 2018) to measure the redundancy of the hierarchy and compress the IPC structure into a tree of specified height, defined as:

$$H(G) = -\sum_{\alpha \in T} \frac{g_\alpha}{vol(G)} \log \frac{vol(\alpha)}{vol(\alpha^-)} \tag{9}$$

where α represents for a non-root node, α^- represents the parent node of non-root node α, g_α represents for the node's degree, and $vol(G)$ represents for the total degree of the node. $vol(\alpha)$and $vol(\alpha^-)$ represent the subset total degrees of node α and node α^-. Structural entropy measures the balance of a graph by considering the difference in degree between different nodes. The larger its value, the greater the difference between its subsets, which indicates a more imbalanced structure.

The algorithm takes the original IPC structure $T = (V_T, E_T, H_T)$ and the target height H as inputs, and mainly involves three steps: tree construction, tree compression, and cross-level link correction. First, all nodes are placed on the same level, and an initial binary tree is built by iteratively merging node pairs that yield the greatest reduction in structural entropy. Second, the tree is compressed to height H by removing nodes with minimal structural entropy. Third, cross-level links are corrected. Ultimately, a coding tree with height H is obtained. More details of this algorithm could be found in Appendix A.

Propagation Tree Learning. Based on the previously constructed encoding tree, we design a propagation mechanism that interacts the patent representation with the hierarchical label structure. During propagation at each layer, a node not only transmits its own features but also fuses with the patent representation $\mathbf{h}$ to enhance context awareness.

To provide initial semantic grounding for IPC nodes, we initialize leaf nodes using short label descriptions embedded by BERT-for-patents. For a node x at layer l, we concatenate its representation $\mathbf{e}_x^{(l)}$ with the patent representation $\mathbf{h}$ and project it via a linear transformation:

$$\mathbf{z}_x^{(l)} = \text{Concat}(\mathbf{e}_x^{(l)}, \mathbf{h}) \tag{10}$$

$$\tilde{\mathbf{e}}_x^{(l)} = \mathbf{W}^{(l)}\mathbf{z}_x^{(l)} + \mathbf{b}^{(l)} \tag{11}$$

In upper layers, each parent node aggregates features from its children using a uniform-weighted sum:

$$\mathbf{e}_x^{(l+1)} = \sum_{i \in \mathcal{C}(x)} w_i \tilde{\mathbf{e}}_i^{(l)}, \quad w_i = \frac{1}{|\mathcal{C}_(x)|} \tag{12}$$

This process is repeated layer by layer. After each aggregation, we perform another interaction with $\mathbf{h}$ to avoid loss of fine-grained context, ensuring that representations at all levels are semantically aligned with the patent.

Multi-level Fusion. This module aggregates multi-layer hierarchical and semantic features to form a global representation. For each layer l, node embeddings $\mathbf{h}_i^l$ are averaged to obtain a layer-wise summary vector:

$$\mathbf{H}^l = \frac{1}{m}\sum_{i=1}^{m} \mathbf{h}_i^l \tag{13}$$

These representations encode information at various levels of abstraction and are subsequently used to generate auxiliary predictions, facilitating gradual and guided learning.

3.4 Classification

Class Prediction. In the prediction stage, the global representations $\mathbf{H}^l$ obtained from each layer are first averaged to ensure consistent dimensionality. These averaged vectors are then passed through a shared linear layer to produce per-layer classification outputs.

$$\mathbf{H} = \sum_{l=1}^{L} \beta_l \mathbf{H}^l, \beta_l = \frac{1}{L} \tag{14}$$

$$y = \text{sigmoid}(\mathbf{W}^l\mathbf{H} + \mathbf{b}^l) \tag{15}$$

Loss Function. Due to the uneven long tail distribution of real patent datasets (Lafond and Kim, 2019), Focal Loss (Lin et al., 2017) and Dice Loss (Milletari et al., 2016) are employed to emphasize low-frequency samples and enhance the model's sensitivity to long-tail classes. Thus, we use Focal Loss and Dice Loss for each layer prediction, defined as:

$$L_l = \mathcal{L}_{\text{FL}}(\hat{y}, y) + \mathcal{L}_{\text{Dice}}(\hat{y}, y) \tag{16}$$

The Focal Loss and Dice Loss could be defined as:

$$\mathcal{L}_{\text{FL}}(\hat{y}, y) = -\sum_{i=1}^{C} \alpha_i (1 - \hat{y}_i)^\gamma y_i \log(\hat{y}_i) \tag{17}$$

$$\mathcal{L}_{\text{Dice}}(\hat{y}, y) = 1 - \frac{2\sum_{i=1}^{C} \hat{y}_i y_i}{\sum_{i=1}^{C} \hat{y}_i + \sum_{i=1}^{C} y_i} \tag{18}$$

where C represents for the whole classes of IPC, and $\hat{y}$ represents the real labels. And in Focal Loss, α and γ are hyper-parameters, and in our paper, we use the default settings of $\alpha = 0.25$ and $\gamma = 2.0$.

In addition, hierarchical feature aggregation is used to derive the global representation $\mathbf{H}^l$ at each layer, which is then mapped to the classification space to generate per-layer predictions.The final global prediction score is a weighted combination of the per-layer predictions. During training, two types of supervision are applied: individual supervision on each layer to encourage multi-granularity learning, and a global prediction loss aligned with the final training objective, defined as:

$$\mathcal{L} = \mathcal{L}_{layers} + \mathcal{L}_{global} \tag{19}$$

$$\mathcal{L}_{layers} = \frac{1}{L}\sum_{l=1}^{L} L_l \tag{20}$$

where $\mathcal{L}_{global}$ is the global prediction loss function.

4 Experiment and Results

4.1 Experimental Settings

Dataset. We construct a patent dataset by extracting patent documents from the USPTO's 2024 raw application data, which is called USPTO-2024. Although we use USPTO data, we do not rely on

USPTO data because our model uses only the title, abstract and claims, which are the core components in patent documents globally.The extracted information includes full claims and their citation structure, titles, abstracts, and IPC labels. The dataset statistics are shown in Table 1, more details could be find in Appendix B.1.

type	train	dev	test
numbers	234598	29325	29325
title	8.17	8.19	8.18
abstract	113.69	113.52	113.58
claims	1923.48	1927.97	1923.12
claim number	16.61	16.66	16.55

Table 1: Base Information of Dataset USPTO-2024, except for numbers, all other entries show the average number of words or claims.

Implementation Details. We conduct all experiments on a NVIDIA 4090 GPU and train all compared models on our dataset, following the original paper settings for each baseline, except for the batch size. In our model, we set the batch size to 8 and train for 1 epoch. The learning rate is set to $1e{-}3$ for the classification head and $1e{-}5$ for the other modules. To address data imbalance, we employed a simple mixed sampling strategy that helps stabilize training. Due to memory constraints, we select $K = 15$ claims per patent, with a maximum length of 256 tokens per claim. The GAT module is configured with $H = 3$ layers and 4 attention heads. For label hierarchy processing, we set the encoding tree height to 3 and adopt average pooling and hierarchical supervision for aggregation.

Baselines. We compare our model against traditional text classification approaches such as TF-IDF+LR and BiLSTM, as well as patent-specific models including DeepPatent (Li et al., 2018), BERT-for-patents (Srebrovic and Yonamine, 2020), and PatentSBERTa (Hamid Bekamiri and Jurowetzki, 2024). Furthermore, we compare with typical hierarchical multi-label classification baselines, such as HARNN (Huang et al., 2019) and HiTIN (Zhu et al., 2023). To demonstrate the robustness of our method, we include comparisons with Qwen1.5 (Bai et al., 2023), an open-source LLM, which poses a significant challenge in patent classification. More experimental setting details could be shown in Appendix B.2.

Evaluate Metrics. To comprehensively evaluate the model's capability, especially its attention to long-tailed categories, we adopt macro-level metrics. Specifically, we report TOP@1 and TOP@5 precision, recall and F1-score as evaluation criteria.

4.2 Main Results

As the results shown in Table 2, through comprehensive comparison, we observe that CSPC-LA achieves state-of-the-art performance on real-world patent datasets. Unlike previous research, our proposed method—based on claim-to-claim citation relationships—leverages more fine-grained representations of patent claims. Specifically, CSPC-LA selects more valuable claims and incorporates structural information from the patent content. Additionally, it emphasizes the interaction between label structure and textual content. These strategies enable CSPC-LA to effectively capture fine-grained features, thereby improving performance on low-frequency (long-tailed) categories.

Traditional text classification models, such as TF-IDF with logistic regression and BiLSTM achieve decent macro precision but suffer from low recall. This is largely due to their bias toward learning prominent features from the majority classes while overlooking rare class signals. Among patent-specific classification models, BERT-for-patents, pre-trained on patent corpora, captures contextual semantics effectively, leading to improved recall and F1 scores. In contrast, PatentSBERTa enhances rare class predictions through semantic neighbor retrieval but struggles with fine-grained representations and sparse neighbor availability. In hierarchical multi-label classification, HARNN lacks explicit modeling of label hierarchy and struggles with long patent texts, resulting in weak macro-level performance. HiTIN improves rare class detection via hierarchical modeling and distribution priors, but its precision is limited by majority-class dominance and underutilized fine-grained features.

4.3 Ablation Study

To analyze the contribution of each component in CSPC-LA, several model variants were constructed by replacing or removing specific modules. The effectiveness of each module was evaluated by observing the precision, recall, and F1 scores of these variants on the dataset.

Firstly, under the setting of not selecting claims but sequentially truncating a fixed number of claims (w/o Method), the performance of the model decreases, indicating that there is content with low information density or even invalidity in the original claims, which may introduce learning noise

Table 2: Performance comparison on USPTO-2024.Bold indicates the best performance; underline indicates the second best.

Models	TOP@1			TOP@5		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
TF-IDF+LR	36.52	7.94	12.10	36.38	10.07	14.55
BiLSTM	41.40	12.61	18.21	39.38	17.11	22.59
DeepPatent	37.37	8.58	13.03	35.56	11.82	16.71
Bert-for-patents	32.03	11.46	14.66	31.64	18.64	19.88
PatentSBerta	15.62	5.07	6.45	10.03	15.93	10.42
HARNN	8.68	2.06	2.80	8.45	3.17	3.83
HiTIN	39.12	17.63	22.55	37.11	25.71	28.44
Qwen1.5	<u>45.91</u>	**19.81**	**24.89**	<u>41.14</u>	<u>27.53</u>	<u>30.69</u>
CSPC-LA	**46.85**	<u>18.34</u>	<u>24.44</u>	**46.39**	**29.09**	**32.59**

Table 3: Ablation study results on USPTO-2024.

Variant	Precision	Recall	F1
w/o Method	38.30	14.13	18.96
w/o Graph	42.98	15.32	20.56
w/o gate	41.01	15.25	20.42
r/p IPC tree	35.16	12.13	16.35
r/p Cross	38.32	13.55	18.20
Full CSPC-LA	46.85	18.34	24.44

and interfere with model judgment. Secondly, removing the claim structure mapping module (w/o Graph) also resulted in a decrease in performance, indicating that this module played a key role in modeling the referencing and hierarchical dependencies between claims, effectively enhancing the internal structural expression ability of patents. Furthermore, when we preserve the graph structure but remove the gating fusion mechanism (w/o gate), the model cannot dynamically weight based on the importance of different information sources, and its performance also decreases, indicating that the information fusion strategy has a positive effect on improving the model's discriminative ability.

In addition, we compared the effectiveness of using encoding trees with the original IPC hierarchical structure and found that directly using the original IPC structure (r/p IPC tree) would introduce an imbalance in the label structure, making the model more inclined to predict labels on high-frequency paths, thereby affecting overall performance; And the encoding tree alleviates this bias to some extent by reconstructing the hierarchical structure. Finally, when we remove the cross layer interaction module (r/p Cross) from the label structure, the model also shows some degradation, indicating that relying solely on same layer label propagation will limit the depth of semantic expression and make it difficult to capture the direct semantic connection between text and high-level labels.

4.4 Label Tree Height Parameter Effect

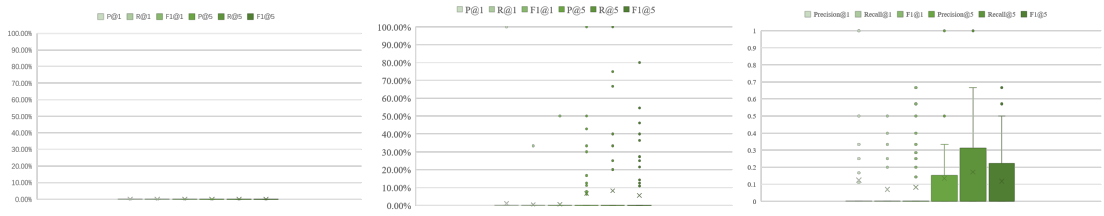
For the label tree experiments, we test encoding tree heights $H = \{2, 3, 4\}$. As for parameter H, it could be observed in Table 4 that the model performs best when the coding tree layer height H is set to 3. When $H = 2$, the label structure is excessively compressed, which may cause child nodes with low similarity to be clustered under the same parent node, resulting in poor model performance; and when $H = 4$, more parameters are introduced, which may lead to performance degradation due to insufficient training.

Table 4: Experiment of H.

	H	2	3	4
TOP@1	Precision	30.77	**46.85**	36.19
	Recall	13.56	**18.34**	14.68
	F1	17.43	**24.44**	19.28
TOP@5	Precision	32.05	**46.39**	36.45
	Recall	20.66	**29.09**	22.78
	F1	22.81	**32.59**	25.55

4.5 Tail Labels Embeddings

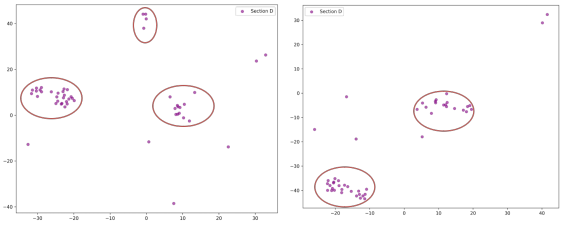
In CSPC-LA, an IPC encoding tree is constructed to enhance patent representations by integrating both the structural representation of patents and the semantic and structural information of labels. The leaf nodes of the IPC encoding tree are initialized using a pretrained model and are further optimized during the hierarchical message-passing



(a) tail-TF-IDF (b) tail-HiTIN (c) tail-CSPC-LA

Figure 4: Tail Class Performance on Bottom 100 classes using TF-IDF+LR, HiTIN, and CSPC-LA.

process. To investigate how label representations evolve through this process, we extract the label representations from the model and analyze their spatial distribution, aiming to determine whether the model has captured the latent relationships between labels. To this end, we apply t-SNE for



(a) Original label representations (Section D) (b) Label representations in CSPC-LA (Section D)

Figure 5: Label representation distributions of Section D after t-SNE dimensionality reduction

dimensionality reduction and visualize the label representations on a 2D plane. We focus on the tail class section "D", and for better visualization, all labels within the section (including section, class, and subclass levels) are colored identically. The visualizations are shown in Figure 5.

We could see that after using the proposed method, labels within the same section become more tightly clustered, reducing from three major clusters to two. This indicates that the model gradually learns the underlying relationships and discriminative features among labels during training. The representations of non-head labels become more compact and coherent, which lays a solid foundation for improving classification performance on long-tail categories.

4.6 Tail Class Study

To further evaluate the proposed model's performance on tail classes, we compare it with two representative baselines: TF-IDF, a traditional keyword-based method, and HiTIN, a hierarchical multi-label classification model. Specifically, we sort all labels by their frequency in the test set in descending order and select the bottom 100 (the rarest) labels. We then compute the precision, recall, and F1 scores for these labels under the TOP@1 and TOP@5 settings, and visualize the results using box plots to observe the models' performance on tail classes.

As shown in Figure 4, the proposed CSPC-LA model outperforms the baselines in terms of average performance on the top 100 tail classes. This is mainly because real-world datasets often follow a long-tailed distribution, where some classes appear very infrequently. The TF-IDF model struggles with tail classes due to the low frequency of domain-specific terms in the overall corpus, causing it to overlook their features. HiTIN, on the other hand, exhibits better tail performance by leveraging prior knowledge of label hierarchy and learning path-based features for rare classes. In contrast, our CSPC-LA model captures the fine-grained textual details of patents and pays more attention to challenges posed by rare classes, resulting in stronger predictive ability on these difficult cases.

5 Conclusion

In this work, we present CSPC-LA, a patent classification model that leverages the structural information of patent claims and the hierarchical organization of labels. By introducing modules for claim-structure graph learning and hierarchical label modeling, our approach captures fine-grained textual signals and strengthens the alignment between patent content and label semantics, resulting in improved classification performance. Moving forward, we aim to explore entity-level structural relations within patents, viewing claim dependencies as semantic elaborations between entities in all types of claims, to further enhance the model's capacity in understanding complex patent structures.

Limitations

Although our model has achieved good results based on this, there are still certain limitations. Firstly, although we collected and processed the 2024 patent application data published by USPTO, covering newer trend information, some labels lacked samples due to year limitations. In order to achieve better learning of label information, future work will consider processing data from other years to improve data diversity. Secondly, although we have focused on the reference relationship at the sentence level in the claims, the reference relationship is essentially a more detailed elaboration of the dependent claims around a certain invention object (entity) in the independent claims. In addition, there are certain differences and similarities between patents belonging to the same label and those belonging to different labels. Comparing them from a more macro perspective can also have good results. Therefore, future work will focus on more detailed information within patents and the relationships between patents. In addition, the training speed of the model is limited by equipment and data volume, which makes its training relatively slow. In the future, parallel operation optimization will be carried out to accelerate the model training and enhance its practical application value. Finally, due to current computational constraints, we have not yet utilized large language models. In future work, we plan to actively explore their application in patent classification to further enhance performance.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (No. 62102026, 62372043).

References

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, and 1 others. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Yi Cheng, Wen Ge, and Li Xu. 2018. Quality of geographical information services evaluation based on order-relation. page 1.

C. J. Fall, A. Trcsvári, K. Benzineb, and G. Karetka. 2003. Automated categorization in the international patent classification. *Acm Sigir Forum*, 37(1):10–25.

Lintao Fang, Le Zhang, Han Wu, Tong Xu, Ding Zhou, and Enhong Chen. 2021. Patent2vec: Multi-view representation learning on patent-graphs for patent classification. *World Wide Web*, 24(5):1791–1812.

Arousha Haghighian Roudsari, Jafar Afshar, Wookey Lee, and Suan Lee. 2022. Patentnet: multi-label classification of patent documents using deep learning based language understanding. *Scientometrics*, 127(1):207–231.

Daniel S. Hain Hamid Bekamiri and Roman Jurowetzki. 2024. Patentsberta: A deep nlp based hybrid model for patent distance and classification using augmented sbert. *Technological Forecasting and Social Change*, 206:123536.

Wei Huang, Enhong Chen, Qi Liu, Yuying Chen, Zai Huang, Yang Liu, Zhou Zhao, Dan Zhang, and Shijin Wang. 2019. Hierarchical multi-label text classification: An attention-based recurrent network approach. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 1051–1060, New York, NY, USA. Association for Computing Machinery.

Jyun-Yu Jiang, Mingyang Zhang, Cheng Li, Michael Bendersky, Nadav Golbandi, and Marc Najork. 2019. Semantic text matching for long-form documents. In *The World Wide Web Conference*, WWW '19, page 795–806, New York, NY, USA. Association for Computing Machinery.

François Lafond and Daniel Kim. 2019. Long-run dynamics of the u.s. patent classification system. *Journal of Evolutionary Economics*, 29(2):631–664.

Leah S Larkey. 1999. A patent search and classification system. In *Proceedings of the fourth ACM conference on Digital libraries*, pages 179–187.

Jieh-Sheng Lee and Jieh Hsiang. 2019. Patentbert: Patent classification with fine-tuning a pre-trained bert model. *Preprint*, arXiv:1906.02124.

Huahang Li, Shuangyin Li, Yuncheng Jiang, and Gansen Zhao. 2022. Copate: A novel contrastive learning framework for patent embeddings. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, page 1104–1113, New York, NY, USA. Association for Computing Machinery.

Shaobo Li, Jie Hu, Yuxin Cui, and Jianjun Hu. 2018. Deeppatent: patent classification with convolutional neural networks and word embedding. *Scientometrics*, 117(2):721–744.

Xin Li, Hsinchun Chen, Zhu Zhang, and Jiexun Li. 2007. Automatic patent classification using citation network information: an experimental study in nanotechnology. JCDL '07, page 419–427, New York, NY, USA. Association for Computing Machinery.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Yunduo Liu, Fang Xu, Yushan Zhao, Zichen Ma, Tengke Wang, Shunxiang Zhang, and Yuhao Tian. 2024. Hierarchical multi-instance multi-label learning for chinese patent text classification. *Connection Science*, 36(1):2295818.

Mihai Lupu, Allan Hanbury, and 1 others. 2013. Patent retrieval. *Foundations and Trends® in Information Retrieval*, 7(1):1–97.

Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. 2016. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 565–571.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 world wide web conference*, pages 1063–1072.

Subhash Pujari, Jannik Strötgen, Mark Giereth, Michael Gertz, and Annemarie Friedrich. 2022. Three real-world datasets and neural computational models for classification tasks in patent landscaping. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, Abu Dhabi, United Arab Emirates.

Subhash Chandra Pujari, Annemarie Friedrich, and Jannik Strötgen. 2021. A multi-task approach to neural multi-label hierarchical patent classification using transformers. In *European Conference on Information Retrieval*, pages 513–528.

Julian Risch, Nicolas Alder, Christoph Hewel, and Ralf Krestel. 2020a. Patentmatch: A dataset for matching patent claims & prior art.

Julian Risch, Samuele Garda, and Ralf Krestel. 2020b. Hierarchical document classification as a sequence generation task. In *Proceedings of the ACM/IEEE joint conference on digital libraries in 2020*, pages 147–155.

Marawan Shalaby, Jan Stutzki, Matthias Schubert, and Stephan Günnemann. 2018. An lstm approach to patent classification based on fixed hierarchy vectors. In *Proceedings of the 2018 SIAM international conference on data mining*, pages 495–503. SIAM.

Rob Srebrovic and Jay Yonamine. 2020. Leveraging the bert algorithm for patents with tensorflow and bigquery. Technical report.

Shoko Suzuki and Hiromichi Takatsuka. 2016. Extraction of keywords of novelties from patent claims. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1192–1200, Osaka, Japan. The COLING 2016 Organizing Committee.

United States Patent and Trademark Office. 2022. Manual of patent examining procedure (mpep) section 2103: Patent examination process. https://www.uspto.gov/web/offices/pac/mpep/s2103.html. Accessed: 2025-05-20.

WIPO. 2024a. Bulk data directory.

WIPO. 2024b. International patent classification. https://www.wipo.int/classifications/ipc/en/ITsupport/Version20240101/transformations/stats.

WIPO. 2024c. International patent classification(ipc). https://www.wipo.int/en/web/classification-ipc. Accessed May 2025.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and 1 others. 2020. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297.

Xinyi Zhang, Jiahao Xu, Charlie Soh, and Lihui Chen. 2022. La-hcn: label-based attention for hierarchical multi-label text classification neural network. *Expert Systems with Applications*, 187:115922.

Jie Zhou, Chunping Ma, Dingkun Long, Guangwei Xu, Ning Ding, Haoyu Zhang, Pengjun Xie, and Gongshen Liu. 2020. Hierarchy-aware global model for hierarchical text classification. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 1106–1117.

Fujin Zhu, Xuefeng Wang, Donghua Zhu, and Yuqin Liu. 2015. A supervised requirement-oriented patent classification scheme based on the combination of metadata and citation information. *International Journal of Computational Intelligence Systems*, 8(3):502–516.

He Zhu, Chong Zhang, Junjie Huang, Junran Wu, and Ke Xu. 2023. Hitin: Hierarchy-aware tree isomorphism network for hierarchical text classification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7809–7821, Toronto, Canada. Association for Computational Linguistics.

A Theroretical Analysis

In subsection 3.3, we introduced the use of structural entropy to measure redundant information in label structures, and compressed label structures by reducing structural entropy redundancy to make their content distribution more uniform. In this section, we provide a detailed introduction to the construction definition and algorithm of encoding trees. The process of building a coding tree mainly includes three types, namely *Merge*, *Delete*, and *Swap*, as shown in Figure 6.

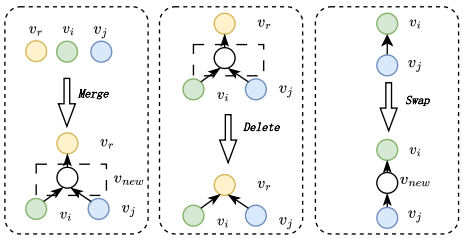


Figure 6: Operations in Constructing Coding Tree

After introducing the three basic operations, we represent the encoding tree construction algorithm. The input of this algorithm is the original IPC structure $T = (V_T, E_T, H_T)$and the height $H$to be compressed. The main steps are to construct the tree, compress the tree, and correct the cross layer connections. Firstly, place all nodes on the same layer and construct an initial binary tree structure based on structural entropy, which is merging the node pairs that minimize the structural entropy each time. Secondly, in the second step, the tree is compressed to a height of H. Then, the node with the lowest structural entropy is selected for removal, and the cross layer links are corrected in the third step. Finally, a coding tree with a height of H is formed, and the process is shown in Algorithm 1.

Algorithm 1: Encoding Tree Construction

Input: Graph $T = (V_T, E_T)$, height H
Output: Encoding tree $T' = (V_{T'}, E_{T'})$
`// Initialization`
$V^0_{T'} \leftarrow V$
while $\exists v_{root}$ *such that* $|v_r.children| > 2$ **do**
  Select the node pair (v_i, v_j) that maximally reduces structural entropy ;
  Merge nodes (v_i, v_j) ;
`// Compress the tree to height` H
while $T'.height > H$ **do**
  Select node v_i whose removal increases structural entropy the least ;
  Delete node (v_i) ;
`// Correct cross-level connections`
foreach $v_i \in T$ **do**
  if $|v_i.parent.height - v_i.height| > 1$ **then**
    Swap node $(v_i, v_i.\text{parent})$;
return T'

B Experimental Details

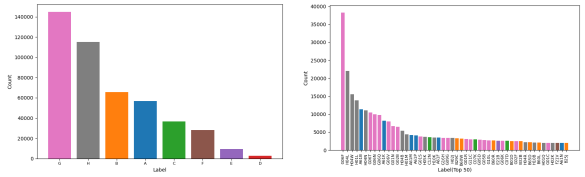
B.1 Dataset

We extracted patent information containing complete claims and their citation structures, titles, abstracts, and IPC tags from the 2024 original patent application data file (WIPO, 2024a) published by USPTO, and constructed a patent text dataset. The raw data is recorded and saved in XML format, with each file containing the patent application content for the past week, and all files organized by date. A total of 52 original compressed files were obtained through the aforementioned website. Using Python to parse XML files, the "xml. etree. ElementTree" library is mainly used to parse strings and extract information from relevant tags of valid strings, including patent number, title, abstract, claims, and IPC classification number. The claims exist in the form of nested tags and are numbered in order during the extraction process. After extraction, store it in the specified file.

Then, extract the original reference relationship of each legal patent claim. Firstly, parse the current claim number from the aforementioned XML tag. Secondly, use regular expressions to extract its reference object according to a specified pattern (such as 'claim n'). Finally, assign a reference number to each claim. When the claim is an independent claim without a reference object, set the number to -1. Based on this, the establishment of original claim citation relationship has been completed and stored in dataset in the form of a list.

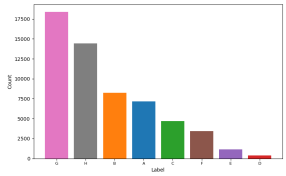
In order to gain a preliminary understanding of label distribution in the dataset, a statistical analysis was first conducted on the patent data, which included counting the sample size of the top 50 subclass level categories, as shown in Figure 7.

As shown in the figure, there are significant differences in the number of samples between different categories, and there are more samples belonging to the G and H parts of the small category,

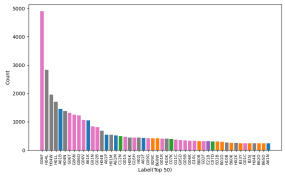


(a) train, count by section

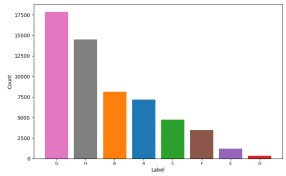
(b) train, count by subclass



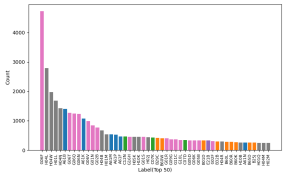
(c) dev, count by section



(d) dev, count by subclass



(e) test, count by section



(f) test, count by subclass

Figure 7: Count on USPTO-2024, the labels were colored based on the type of part to which the subclass belongs.

while there are fewer samples belonging to the D part, ranking higher and showing a certain long tail distribution trend (Lafond and Kim, 2019).

B.2 Baseline Settings

TF-IDF+LR: A logistic regression method using TF-IDF features, and it establishes patent representations based on TF-IDF, a method that measures the ability of keywords in text to distinguish categories.

BiLSTM: Use bidirectional long short-term memory networks to capture contextual sequence information. It obtains information by modeling the temporal dependencies between words in patent texts.

DeepPatent (2018) (Li et al., 2018): A method that gets patents' information by extracting key information from local n-gram features based on convolutional neural networks. It focuses on the ability to model local text structures.

BERT-for-patents(2020) (Srebrovic and Yonamine, 2020): A BERT-large model developed by Google, which is pre-trained by over 100 million patent documents, which concludes all texts of patents (abstracts, claims, descriptions). In this way, it has been endowed with rich knowledge in the field of patents

PatentSBerta (2024) (Hamid Bekamiri and Jurowetzki, 2024): Use claim text to calculate the similarity between patents, and use the classification number of neighboring patents to calculate the target patent category. Basically, it uses STS-B datasets to train a score model Roberta, which is used to train embedding model SBERT by scoring patent texts pair and making data for SBERT. And Finally, it uses neighbors' labels for predictions.

HARNN (2019) (Huang et al., 2019): Using RNN and hierarchical attention mechanism, key information is extracted from both sentence and document levels, highlighting important content in patents.

HiTIN (2023) (Zhu et al., 2023): Enhance text representation using label hierarchy information for hierarchical multi label text classification, using global classification methods. HiTIN ensures that the network can effectively learn the relationships between labels through node connection initialization, making it more suitable for scenarios with imbalanced data.

Qwen-1.5-7B (2023) (Bai et al., 2023): Qwen1.5-7B is part of Alibaba's open-source LLM series, built on the Transformer architecture with SwiGLU activation. It supports a context length of up to 32K tokens and demonstrates strong performance in multilingual understanding, generation, and coding tasks.

B.3 Parameter Effect

To investigate the impact of parameter settings on model performance, we conducted experiments in two main parts: one focusing on patent representation parameters—namely the number of claims N, the number of words per claim W, and the number of GAT layers L; and the other focusing on label tree-related parameters, including the encoding tree height H and aggregation methods (AVG, ALL and SUM). Due to device memory constraints, larger parameter values were chosen when necessary. The experimental results for both parts are shown below.

Structural Text Representation Part. In this part, we examined the influence of each parameter on patent representation without incorporating labels. We experimented with $N = \{5, 10, 15\}$, $W = \{64, 128, 256\}$, and $L = \{1, 2, 3\}$.

As shown in Table 5, the results indicate that model performance improves as the number of claims N increases, reaching the best performance at $N = 15$. This may be because a smaller N truncates many claims, losing rich information, whereas $N = 15$ approximates the average number

of claims, providing more complete content.

Table 5: Experiment of N.

	N	5	10	15
TOP@1	Precision	41.18	43.20	**45.13**
	Recall	14.76	15.64	**17.20**
	F1	19.76	21.14	**23.01**
TOP@5	Precision	38.82	41.68	**43.25**
	Recall	24.23	24.27	**26.41**
	F1	26.41	27.66	**29.33**

For the word count per claim W, as shown in Table 6, it can be observed that when $W = 128$, the performance is competitive, but overall, the results achieved in TOP@1 are better when $W = 256$. Therefore,$W = 256$ is chosen as the reserved word count for the claims.

Table 6: Experiment of W.

	W	64	128	256
TOP@1	Precision	41.18	**45.77**	45.13
	Recall	14.76	16.03	**17.20**
	F1	19.76	21.72	**23.10**
TOP@5	Precision	38.82	**43.28**	43.25
	Recall	24.23	26.40	**26.41**
	F1	38.76	**29.49**	29.33

In Table 7, regarding the number of GAT layers L, $L = 3$ yielded the highest recall and F1 scores, with minor changes in precision, indicating that deeper layers can enhance model performance.

Table 7: Experiment of L.

	L	1	2	3
TOP@1	Precision	**45.68**	45.51	45.13
	Recall	15.70	15.58	**17.20**
	F1	21.77	21.59	**23.10**
TOP@5	Precision	42.09	41.34	**43.25**
	Recall	24.79	24.51	**26.41**
	F1	28.34	28.03	**29.33**

Label Aware Representation Learning. For the label tree experiments, we tested three aggregation methods: SUM (summing representations from all layers for prediction), AVG (averaging representations from all layers), and ALL (based on AVG but with supervision applied to each layer separately).

In terms of aggregation methods, which is shown in Table 8, the ALL method performs the best because it adds independent supervision for each layer on top of global integration, allowing the model to learn useful features at different granularities, which is superior to the AVG method that only uses global supervision. In contrast, the AVG method is more stable than the SUM method because it averages the representations of each layer, avoiding the problem of the model being biased towards individual layers due to the amplification of information in one layer of the SUM method.

Table 8: Experiment of Aggregation Methods.

	M	SUM	AVG	ALL
TOP@1	Precision	36.81	39.76	**46.85**
	Recall	14.58	15.27	**18.34**
	F1	19.27	20.25	**24.44**
TOP@5	Precision	37.55	39.74	**46.39**
	Recall	23.73	24.34	**29.09**
	F1	25.93	26.95	**32.59**

C Claim Selection Case Study

In the patent application process, claims are often withdrawn due to examiner feedback or voluntary amendments. Such withdrawals are typically marked as "canceled" or "deleted". As this status may change over different patent versions, it is more reasonable to determine the final set of valid claims during the analysis phase. To illustrate our method, we analyze patent *US10029235B2*. The basic statistics are shown in Table 9, where a large portion of the original claims were withdrawn.

Table 9: Statistics of patent US10029235B2

Original claims	58
Withdrawn claims	38
Average words	24.7
1st valid claim	33. A method of . . .
1st withdrawn claim	1–32. (canceled)

After concatenating the title and abstract, we compute the semantic similarity between each claim and the input text using a pretrained model. The results are visualized in Figure 8, where a darker color indicates higher cosine similarity. The model effectively identifies withdrawn claims with low relevance, allowing us to remove them to avoid processing redundant information.

We further compare the citation structures of claims selected by two methods: a sequential strategy and a semantic-based selection strategy. Figure

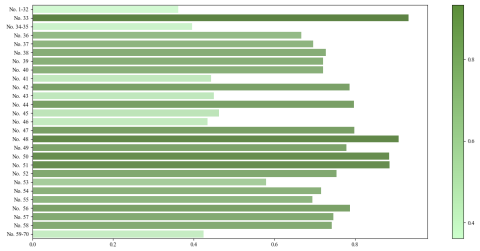


Figure 8: Cosine similarity between each claim and the concatenated input

9 visualizes the retained claims and their reference links. Nodes with orange borders denote sequentially selected claims, while yellow-filled nodes indicate those selected via the semantic method. Given a target of selecting 15 claims, the sequential method only retained 10 valid ones due to withdrawals. In contrast, the semantic method successfully selected 15 meaningful claims, including higher-level nodes with rich descriptions. Claims #36 and #37 were excluded due to their short lengths (8 and 9 words), which may limit their informativeness during embedding.

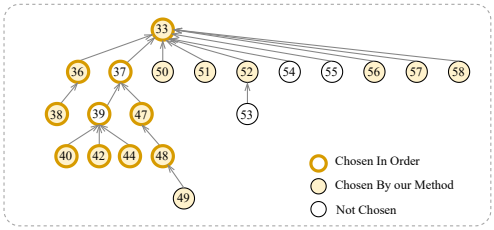


Figure 9: Citation structure of origin valid claims