Research on Deep Learning-Based Automatic Classification Methods for Chinese Patents (Postprint)

Authors: Lyu Lucheng, Han Tao, Zhou Jian, Zhao Yajuan

Date: 2023-04-01T00:00:00+00:00

Abstract

[Purpose/Significance] Aimed at the objective demand for massive patent classification in current domestic patent examination and patent intelligence analysis, this study designs seven deep learning-based automatic patent classification methods and compares their classification performance, thereby contributing to the improvement of both efficiency and effectiveness in patent classification.

[Method/Process] To address the limitations of traditional machine learning methods, seven deep learning models—including Word2Vec+TextCNN, Word2Vec+GRU, Word2Vec+BiGRU, and Word2Vec+BiGRU+TextCNN—are designed based on deep learning technologies such as Word2Vec, CNN, RNN, and Attention mechanisms, with consideration for patent text sequential features, contextual features, and key classification features. Using Chinese patents as a case study, the "section" of the IPC main classification number is selected as the classification criterion, and the performance of these seven models is compared against three traditional classification models in Chinese patent classification tasks.

[Results/Conclusion] Empirical results demonstrate that deep learning methods which consider sequential features, contextual features, and enhanced key features achieve superior classification performance for Chinese patent classification.

Full Text

Research on Chinese Patent Automatic Classification Methods Based on Deep Learning

Lü Lucheng[1],[2], Han Tao[1],[2], Zhou Jian[3], Zhao Yajuan[1],[2]

[1]National Science Library, Chinese Academy of Sciences, Beijing 100190
[2]Department of Library, Information and Archives Management, School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100190
[3]Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190

Abstract: [Purpose/Significance] To address the objective demand for massive patent classification in current domestic patent examination and patent intelligence analysis work, this study designs seven deep learning-based patent automatic classification methods and compares their classification effects, thereby contributing to the improvement of patent classification efficiency and effectiveness. [Method/Process] Aiming at the defects of traditional machine learning methods, and based on deep learning technologies such as Word2Vec, CNN, RNN, and Attention mechanisms, this study considers patent text word order features, contextual features, and key classification features to design seven deep learning models including Word2Vec+TextCNN, Word2Vec+GRU, Word2Vec+BiGRU, and Word2Vec+BiGRU+TextCNN. Using Chinese patents as examples and selecting the "Section" of the IPC main classification number as the classification basis, this study compares the performance of these seven models with three traditional classification models on Chinese patent classification tasks. [Result/Conclusion] Empirical research results demonstrate that deep learning methods considering word order features, contextual features, and reinforced key features achieve superior classification performance for Chinese patent classification.

Keywords: patent automatic classification; deep learning; word embedding; patent text mining
Classification Number: G254.11
DOI: 10.13266/j.issn.0252-3116.2020.10.009

2 Research Status of Patent Automatic Classification

Patent automatic classification is the process by which computers automatically assign one or several patent classification numbers to patents based on specific rules, metadata, or text content features. From the perspective of classification systems, research can be divided into two categories: classification based on existing patent classification systems and classification based on personalized classification systems. Studies based on existing patent classification systems primarily use international universal classification systems such as IPC [7-9], USPC [10], ECLA [11-12], and FI/F-term [13-14] as the basis for classification. Research based on personalized classification systems mainly uses classification systems built upon classical theoretical frameworks like TRIZ or customized according to specific requirements, such as the work by C. He [15-16], Hu Zhengyin [17], and Zhai Jiqiang [18] on patent classification for TRIZ-based design, Liu Longfan et al. [19] on automatic classification based on a functional basis classification system for product innovation design, and X. Zhang [20] on automatic

classification based on a classification system for the electric vehicle domain (expert-defined).

From the perspective of classification methods, research can be categorized into three types: rule-based classification, citation relationship-based classification, and text content mining-based classification. In rule-based classification, for example, C. He [16] used association rule mining to identify category rules and build automatic classifiers. In citation relationship-based classification, S. Chang et al. [21] clustered patents based on citation relationships and interpreted the technologies involved in clusters to construct classification systems, while K. Lai et al. [22] established classification systems using factor analysis based on co-citation relationships of foundational patents. Research based on text content mining is more numerous and continues to receive attention, which will be discussed in detail below.

Automatic classification based on patent text content mining belongs to the text classification task in Natural Language Processing (NLP). The classical approach employs machine learning methods to identify potential classification features through feature engineering, and then uses algorithms such as Bayesian classifiers, SVM, and logistic regression for automatic classification. Commonly used features are bag-of-words features, which represent patent texts as word frequency vectors using the Bag-of-Words model [8,23]. However, the simple word frequency representation introduces noise from high-frequency function words (such as function words and conjunctions). Later, the method of using Term Frequency-Inverse Document Frequency (TF-IDF) to replace word frequency in the original vector was widely adopted, as seen in the work of Jia Shanshan et al. [24] who extracted TF-IDF features from patent applications to train Naive Bayes, SVM, and AdaBoost classifiers for IPC classification prediction. Additionally, new features are continuously being introduced to improve classification performance, such as S. Verberne et al. [25] who added semantic triple information of feature words to improve classification accuracy, J. Stutzki et al. [26] who introduced geographic data features of patent applicants and used KNN and one-versus-rest SVM classifiers for patent classification, and S. Lim et al. [27] who extracted features from titles, abstracts, claims, technical fields, and background technology information to improve patent text classification effectiveness.

Classic machine learning-based patent automatic classification relies on researchers manually constructing features to achieve better classification results. However, feature representation methods such as the bag-of-words model lose semantic information like word meaning and word order in patent texts. For example, two documents in the same category may fail to be accurately classified due to different word usage. In recent years, with the rise of deep learning technology and its continuous application in patent intelligence research, a series of research achievements have emerged in the scenario of patent automatic classification based on patent text content mining. For instance, Ma Shuanggang [28] designed an automatic classification method

combining Denoising Autoencoder (DAE) and SVM algorithms based on deep learning theory, and verified classification effectiveness on six IPC categories in the computer domain. Hu Jie et al. [29] proposed a patent text classification model based on convolutional neural networks and random forest algorithms for English mechanical patent text classification. Ma Jianhong et al. [30] constructed a bidirectional LSTM (Long Short-Term Memory) model based on attention for training and classification testing on mechanical physics patent texts with 100 patent application effect categories as class labels. S. B. Li et al. [31] proposed a DeepPatent method based on convolutional neural networks and word embedding technology for automatic classification of IPC subclasses for English patents. Xiao Lizhong et al. [32] invented a classification method for security domain Chinese patent texts using Word2Vec and LSTM models, which achieved significant accuracy improvements on security domain Chinese patent test sets. In summary, domestic and international research on patent classification method improvements and applications based on deep learning technology has achieved some results. However, these studies basically compare the classification effects of improved specific deep learning methods with traditional machine learning methods, and have not yet formed a hierarchical method optimization logic system.

Therefore, this study addresses the objective demand for large-scale domestic patent document classification in current patent examination and patent intelligence analysis work. Targeting the defects of traditional methods and considering patent text word order features, contextual features, and key classification features, this study introduces deep learning technology to hierarchically and systematically design seven patent deep learning classification methods. Using Chinese patents as examples and selecting the "Section" of the IPC main classification number as the classification basis, this study compares the performance of ten automatic classification methods on Chinese patent classification tasks to analyze and evaluate the effectiveness of deep learning technology for patent automatic classification and to assist patent classification work.

3 Method Design

3.1 Related Concepts

Text vector representation and classification models are fundamental to text classification. The following explains the relevant text vector representation methods and basic classification models selected for this study.

3.1.1 Text Vector Representation The classical approach to text vector representation uses the Vector Space Model (VSM), which represents text as a vector composed of real-valued components, where components can be represented by word frequency or TF-IDF values. Since word frequency cannot represent word importance while TF-IDF can evaluate a word' s importance to a document in a corpus, the TF-IDF vector representation approach is currently

more widely used. Although the vector space model has the advantages of being clear, explicit, and interpretable, it suffers from the problem that vector dimensionality increases with vocabulary size and vectors are highly sparse. Additionally, it cannot handle semantic issues like synonyms and near-synonyms. For example, the three terms "术语抽取" (term extraction), "语义挖掘" (semantic mining), and "太阳能电池" (solar cell) represent three feature dimensions in TF-IDF feature vectors, and although these word features have certain semantic similarity relationships, they cannot be measured in TF-IDF.

To address this, Google' s Word2Vec technology, introduced in 2013, can use low-dimensional continuous distributed vectors to represent word semantics and effectively characterize similarity relationships between semantically similar words such as synonyms and near-synonyms, thus offering higher usability in text vector representation. This study adopts the Word2Vec-based word vector method for patent text vector representation in deep learning models and uses the TF-IDF-based method as the text vector representation approach for control models.

3.1.2 Basic Models (1) ANN Model. The ANN model is a basic fully connected layer model in neural networks, including three layers: input layer, hidden layer, and output layer, with full connections between layers. The ANN model can map continuous distributed vector representations to the label space of patent texts, essentially performing further feature transformation on patent text word vector representations to highlight and fuse relevant features.

(2) TextCNN Model. TextCNN is a representative model that applies convolutional neural networks to NLP tasks [33]. It combines convolutional neural networks with the N-gram concept from language models, using convolutional kernels of different sizes to extract contextual features at different dimensions from text vectors, then performs feature enhancement through max pooling operations to improve text feature extraction capability and text classification effectiveness.

Assuming a text word vector representation $X = \{x_1, x_2, ..., x_m\}$, where $x_i \in \mathbb{R}^d$, TextCNN consists of three stages: convolutional layer, pooling layer, and fully connected layer, as shown in Figure 1 [Figure 1: see original paper]. The input layer is x_i, representing the word vector of a patent text. The convolutional layer combines the N-gram concept and uses four convolution kernel sizes of $2 * d$, $3 * d$, $4 * d$, and $5 * d$ to extract local features from $x_{1:m}$ at different dimensions. The formulas are as follows:

$$c_i = f(w * x_{i:i+h-1} + b) \quad (1)$$

$$C = [c_1, c_2, ..., c_{m-h+1}] \quad (2)$$

where w represents the convolution kernel parameters, h represents the convolution kernel height, $w \in \mathbb{R}^{h*d}$, b is the bias term, $b \in \mathbb{R}$, $f(\cdot)$ is the ReLU

activation function, and C is an output of the convolutional layer, $C \in \mathbb{R}^{m-h+1}$.

Next, a max pooling layer is used to enhance features, i.e., $\hat{C} = \max(C)$. Finally, the pooling layer results are concatenated and passed through a fully connected layer to obtain the TextCNN output, or Softmax classification can be performed directly through the pooling layer output.

(3) GRU/BiGRU Model. GRU is a variant of recurrent neural networks, similar to LSTM, and is a special recurrent neural network structure [34-35]. Standard RNN units contain only one tanh layer for repeated learning, which leads to the vanishing or exploding gradient problem. To solve these problems, gated recurrent neural networks such as LSTM and GRU were proposed. This study selects GRU over LSTM because experiments show that GRU and LSTM have similar effects, but GRU has fewer training parameters (three gate units in LSTM reduced to two gate units in GRU), making it relatively easier to train and less prone to overfitting.

GRU sequentially computes text vectors through shared-parameter GRU units and uses the hidden vector from the final step as the representation of the original text vector for classification. The GRU unit contains two gates: update gate and reset gate, as shown in Figure 2 [Figure 2: see original paper]. The GRU unit calculation formulas are as follows:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \quad (4)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \quad (5)$$

$$\tilde{h}_t = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \quad (6)$$

$$h_t = (1 - z_t)h_{t-1} + z_t \tilde{h}_t \quad (7)$$

where h_{t-1} is the output of the GRU unit at time $t-1$, x_t is the input at time t, z_t is the output of the update gate, W_z, U_z, and b_z are the weights of the update gate, r_t is the output of the reset gate, W_r, U_r, and b_r are the weights of the reset gate, W_h, U_h, and b_h are the weights of the output gate, and h_t is the output of the GRU unit at time t.

Assuming a text word vector representation $X = \{x_1, x_2, ..., x_m\}$, after encoding through the GRU layer, the hidden representation of the vector text is $H = \{h_1, h_2, ..., h_m\}$, where:

$$h_t = \text{GRU}(h_{t-1}, x_t), \quad t \in [1, m] \quad (8)$$

h_t represents the hidden representation at step t.

GRU considers the context information above a word when encoding sentences, but often the words below a word also play a role in word encoding. Therefore, bidirectional recurrent neural networks are considered for sentence encoding. BiGRU builds upon GRU by encoding sentences in both forward and

backward directions through GRU units. The parameters inside the forward and backward encoding GRU units are not shared. The text vector after encoding through the BiGRU layer obtains the hidden vector representation $H = \overrightarrow{H} \oplus \overleftarrow{H} = \{h_1, h_2, ..., h_m\}$, where:

$$\overrightarrow{h}_t = \text{GRU}(x_t, \overrightarrow{h}_{t-1}), \quad t \in [1, m] \quad (9)$$

$$\overleftarrow{h}_t = \text{GRU}(x_t, \overleftarrow{h}_{t+1}), \quad t \in [m, 1] \quad (10)$$

Word order information is an important feature of text. Since GRU recurrent neural networks can model and represent word order information through sequential modeling of text vectors, they are introduced into patent text classification. As GRU only considers the word order features above the text vector, while BiGRU considers contextual word order features, both are considered in this study to investigate classification effectiveness.

(4) Attention Mechanism. The Attention mechanism originated in the visual image domain and was later applied to the NLP domain, continuously achieving new progress [36]. After continuous improvement, various variants have been formed, but the core idea is basically to highlight features with greater impact on results by assigning different weight coefficients to vectors. The Attention method adopted in this study follows this basic idea: assuming the original text word vector representation $X = \{x_1, x_2, ..., x_m\}$, after obtaining the hidden representation at each step $H = \{h_1, h_2, ..., h_m\}$ through recurrent neural networks, a weight vector a_t is assigned to the hidden representation obtained at each step of the recurrent neural network, with the formula as follows:

$$u_t = \tanh(W_a h_t + b_a) \quad (11)$$

$$a_t = \frac{\exp(u_t^T u_w)}{\sum_t \exp(u_t^T u_w)} \quad (12)$$

$$c_t = a_t h_t \quad (13)$$

where u_t is the hidden representation of h_t, $U = \{u_1, u_2, ..., u_m\}$, a_t is the normalized probability weight calculated through the hidden representation u_t, and $C = \{c_1, c_2, ..., c_m\}$ is the text vector based on weighted representation obtained through probability weights and the original hidden representation of the recurrent neural network.

The Attention mechanism assigns different weights to different positions of word vectors or other hidden representation vectors in the text through a self-learned weight matrix, aiming to highlight key features and ignore useless features, making the model pay more attention to parts that have greater impact on results. The Attention mechanism is not limited by sentence length and can highlight key features in long sentences. Therefore, this study introduces the Attention mechanism into the classification model.

3.2 Classification Model Design

Based on the above text vector representation methods and basic models, and considering the relatively structured writing characteristics of patent texts, this study designs seven deep learning models and three classical machine learning models as controls to evaluate the classification effectiveness of deep learning models, as shown in Table 1 .

Table 1 Ten Patent Automatic Classification Models Designed in This Study

Classical Machine Learning Models	Deep Learning Models
TFIDF+LR	Word2Vec+ANN
TFIDF+DT	Word2Vec+TextCNN
TFIDF+RF	Word2Vec+ATT
	Word2Vec+GRU
	Word2Vec+BiGRU
	Word2Vec+BiGRU+TextCNN
	Word2Vec+BiGRU+ATT+TextCNN

3.2.1 TFIDF+Classical Machine Learning Models "TFIDF+Classical Machine Learning Models"are the baseline control models designed in this study. They use TF-IDF feature vectors of patent texts as input and employ three classical classification models—Logistic Regression (LR), Decision Tree (DT), and Random Forest (RF)—to train patent text automatic classifiers.

3.2.2 Word2Vec+ANN "Word2Vec+ANN"is a classification model designed to distinguish synonyms and near-synonyms in patent texts to achieve better automatic classification performance. Assuming a patent text word vector representation $X = \{x_1, x_2, ..., x_m\}$, where $x_i \in \mathbb{R}^d$, the Word2Vec+ANN model calculation formulas are simply described as follows:

$$X = \text{Flatten}(X) \quad (14)$$

$$H = \tanh(W_h X + b_h) \quad (15)$$

$$O = \text{softmax}(W_o H + b_o) \quad (16)$$

$$\hat{y} = \arg\max(O) \quad (17)$$

where Flatten$(\cdot)$ is the vector flattening operation that transforms high-dimensional vectors into one-dimensional vectors, H represents the hidden layer output, O represents the output layer output, $\hat{y}$ represents the label predicted by the model, and W and b are network weight parameters.

3.2.3 Word2Vec+TextCNN The Word2Vec+ANN model directly flattens patent text vectors, a process that loses much semantic information such as text context and word order, and cannot leverage the feature extraction and representation capabilities of deep learning. To extract and enhance local contextual features, this study proposes the "Word2Vec+TextCNN" model.

Assuming a patent text word vector representation $X = \{x_1, x_2, ..., x_m\}$, where $x_i \in \mathbb{R}^d$, the Word2Vec+TextCNN model calculation formulas are simply described as follows:

$$C_j = \text{Conv1d}(X, j), \quad j \in [2, 5] \quad (18)$$

$$P_j = \text{Maxpooling}(C_j), \quad j \in [2, 5] \quad (19)$$

$$O_{\text{conv}} = P_2 \oplus P_3 \oplus P_4 \oplus P_5 \quad (20)$$

$$O = \text{softmax}(W_o O_{\text{conv}} + b_o) \quad (21)$$

$$\hat{y} = \arg\max(O) \quad (22)$$

In $\text{Conv1d}(X, j)$, X represents the input patent text word vector to TextCNN, j represents the convolution kernel size, $\text{Maxpooling}(\cdot)$ represents the max pooling operation, $\oplus$ represents vector concatenation, W_o and b_o represent the network parameters of the output layer, and $\hat{y}$ represents the category predicted by the model.

3.2.4 Word2Vec+GRU The Word2Vec+TextCNN model extracts and enhances features of the current word and neighboring words but does not consider the global word order features of patent texts. For NLP tasks, word order features are a very unique characteristic. For images, swapping pixel values at two positions may not have a particularly large impact on results, but for text, swapping the order of two words may cause significant changes in sentence semantics. Therefore, to address the issue of modeling sentence word order, this study proposes the "Word2Vec+GRU" model.

Assuming a patent text word vector representation $X = \{x_1, x_2, ..., x_m\}$, where $x_i \in \mathbb{R}^d$, the Word2Vec+GRU model calculation formulas are simply described as follows:

$$h_t = \text{GRU}(h_{t-1}, x_t), \quad t \in [1, m] \quad (23)$$

$$O = \text{softmax}(W_o h_m + b_o) \quad (24)$$

$$\hat{y} = \arg\max(O) \quad (25)$$

In $\text{GRU}(h_{t-1}, x_t)$, h_{t-1} represents the hidden representation at step $t-1$, x_t represents the current input, h_m represents the hidden representation at the final step, O represents the output layer output, W_o and b_o represent the network parameters of the output layer, and $\hat{y}$ represents the category predicted by the model.

3.2.5 Word2Vec+BiGRU The Word2Vec+GRU model considers forward word order features, meaning that when computing at time step t, it only considers the historical states of the previous $t-1$ steps and does not consider information after $t+1$. Therefore, using GRU for sequential modeling of patent text word vectors may be incomplete. In contrast, BiGRU performs bidirectional modeling on patent text word vectors, considering not only bidirectional word order features but also forward and backward semantic features. Based on this, this study proposes the "Word2Vec+BiGRU" model.

Assuming a patent text word vector representation $X = \{x_1, x_2, ..., x_m\}$, where $x_i \in \mathbb{R}^d$, the Word2Vec+BiGRU model calculation formulas are described as follows:

$$\overrightarrow{h}_t = \mathrm{GRU}(\overrightarrow{h}_{t-1}, x_t), \quad t \in [1, m] \quad (26)$$

$$\overleftarrow{h}_t = \mathrm{GRU}(\overleftarrow{h}_{t+1}, x_t), \quad t \in [m, 1] \quad (27)$$

$$O = \mathrm{softmax}(W_o(\overrightarrow{h}_1 \oplus \overleftarrow{h}_1) + b_o) \quad (28)$$

$$\hat{y} = \arg\max(O) \quad (29)$$

where $\overrightarrow{h}_t$ is the hidden representation at step t for forward modeling, $\overleftarrow{h}_t$ is the hidden representation at step t for backward modeling. According to the above formulas, the final hidden representation for forward modeling is $\overrightarrow{h}_m$, and the final hidden representation for backward modeling is $\overleftarrow{h}_1$. Therefore, $\overrightarrow{h}_m \oplus \overleftarrow{h}_1$ is the hidden output of the BiGRU layer, O represents the output layer output, W_o and b_o represent the network parameters of the output layer, and $\hat{y}$ represents the category predicted by the model.

3.2.6 Word2Vec+BiGRU+TextCNN The Word2Vec+BiGRU model comprehensively considers bidirectional word order features and dynamically adjusts word vectors based on contextual semantic information, but it does not extract and enhance the contextual features of the current word, which may cause some hidden key features to not be prominently highlighted, leading to suboptimal classification results. Therefore, this study combines the BiGRU model that extracts sequential features with the TextCNN model that enhances contextual features to propose the "Word2Vec+BiGRU+TextCNN" model. This model first uses BiGRU for bidirectional modeling of patent text vector representations to obtain hidden representations with dynamically adjusted word vectors based on context, then uses this hidden representation as input to TextCNN for feature extraction through convolutional neural networks and feature enhancement through the pooling layer.

Assuming a patent text word vector representation $X = \{x_1, x_2, ..., x_m\}$, where $x_i \in \mathbb{R}^d$, the Word2Vec+BiGRU+TextCNN model calculation formulas are simply described as follows:

$$\overrightarrow{h}_t = \mathrm{GRU}(\overrightarrow{h}_{t-1}, x_t), \quad t \in [1, m] \quad (30)$$

$$\overleftarrow{h}_t = \mathrm{GRU}(\overleftarrow{h}_{t+1}, x_t), \quad t \in [m, 1] \quad (31)$$

$$H = \overrightarrow{h}_1 \oplus \overleftarrow{h}_1 \oplus \overrightarrow{h}_2 \oplus \overleftarrow{h}_2 \oplus ... \oplus \overrightarrow{h}_m \oplus \overleftarrow{h}_m \quad (32)$$

$$C_j = \mathrm{Conv1d}(H, j), \quad j \in [2, 5] \quad (33)$$

$$P_j = \mathrm{Maxpooling}(C_j), \quad j \in [2, 5] \quad (34)$$

$$O_{\mathrm{conv}} = P_2 \oplus P_3 \oplus P_4 \oplus P_5 \quad (35)$$

$$O = \mathrm{softmax}(W_o O_{\mathrm{conv}} + b_o) \quad (36)$$

$$\hat{y} = \arg\max(O) \quad (37)$$

where H represents the hidden representation output by the BiGRU layer, consisting of forward and backward hidden representations.

3.2.7 Word2Vec+Attention TextCNN can capture local contextual key features, and BiGRU can model and extract sequential features. However, these two methods have a limitation: they cannot effectively capture and enhance key features over long distances. Since the Attention mechanism can highlight key features in long sentences, this study proposes the "Word2Vec+Attention" model, which obtains a feature weight matrix corresponding to word vectors through word vector training, and obtains the final text vector representation through weighted word vectors based on weights.

Assuming a patent text word vector representation $X = \{x_1, x_2, ..., x_m\}$, where $x_i \in \mathbb{R}^d$, the Word2Vec+Attention model calculation formulas are simply described as follows:

$$u_t = \tanh(W_a x_t + b_a) \quad (38)$$

$$a_t = \frac{\exp(u_t^T u_w)}{\sum_t \exp(u_t^T u_w)} \quad (39)$$

$$c = \sum_t a_t x_t \quad (40)$$

$$O = \mathrm{softmax}(W_o c + b_o) \quad (41)$$

$$\hat{y} = \arg\max(O) \quad (42)$$

where u_t is the hidden representation calculated from x_t, a_t is the weight vector obtained by normalizing the hidden representation, W and b are network parameters, and c represents the text vector representation obtained through weighting based on the Attention weight matrix.

3.2.8 Word2Vec+BiGRU+Attention+TextCNN Integrating the characteristics of the above six deep learning models, this study combines the BiGRU model that can perform bidirectional sequential modeling on patent text vectors, the TextCNN model that uses convolutional neural networks to extract local features and enhances features through pooling layers, and the Attention mechanism that can ignore distance to enhance key features, to propose the seventh deep learning model– "Word2Vec+BiGRU+Attention+TextCNN" . This model first dynamically adjusts word vectors through BiGRU, then uses the Attention mechanism to adjust the weights of the hidden representations output by BiGRU, and finally uses the adjusted hidden representations as input to TextCNN.

Assuming a patent text word vector representation $X = \{x_1, x_2, ..., x_m\}$, where $x_i \in \mathbb{R}^d$, the Word2Vec+BiGRU+Attention+TextCNN model calculation formulas are simply described as follows:

$$\overrightarrow{h}_t = \mathrm{GRU}(\overrightarrow{h}_{t-1}, x_t), \quad t \in [1, m] \quad (43)$$

$$\overleftarrow{h}_t = \mathrm{GRU}(\overleftarrow{h}_{t+1}, x_t), \quad t \in [m, 1] \quad (44)$$

$$H = \overrightarrow{h}_1 \oplus \overleftarrow{h}_1 \oplus \overrightarrow{h}_2 \oplus \overleftarrow{h}_2 \oplus ... \oplus \overrightarrow{h}_m \oplus \overleftarrow{h}_m \quad (45)$$

$$u_t = \tanh(W_a(h_t) + b_a) \quad (46)$$

$$a_t = \frac{\exp(u_t^T u_w)}{\sum_t \exp(u_t^T u_w)} \quad (47)$$

$$c_t = a_t(h_t) \quad (48)$$

$$C = \{c_1, c_2, ..., c_t\} \quad (49)$$

$$\mathrm{Conv}_j = \mathrm{Conv1d}(C, j), \quad j \in [2, 5] \quad (50)$$

$$P_j = \mathrm{Maxpooling}(\mathrm{Conv}_j), \quad j \in [2, 5] \quad (51)$$

$$O_{\mathrm{conv}} = P_2 \oplus P_3 \oplus P_4 \oplus P_5 \quad (52)$$

$$O = \mathrm{softmax}(W_o O_{\mathrm{conv}} + b_o) \quad (53)$$

$$\hat{y} = \arg\max(O) \quad (54)$$

where H is the hidden representation output by the BiGRU layer, and C is the hidden representation output by the Attention layer, which differs from the Word2Vec+Attention model that directly performs weighted processing.

3.3 Model Effectiveness Evaluation Metrics

This study selects three evaluation metrics to assess model effectiveness: precision, recall, and F1-score, using macro-averaged metrics. Macro-averaged metrics first calculate statistical metrics for each category, then compute the arithmetic mean across all categories, with formulas as follows:

$$p_k = \frac{\text{Number of samples correctly predicted as category } k}{\text{Number of samples predicted as category } k} \quad (55)$$

$$r_k = \frac{\text{Number of samples correctly predicted as category } k}{\text{Number of samples of category } k \text{ in test set}} \quad (56)$$

$$F1_k = \frac{2 \cdot p_k \cdot r_k}{p_k + r_k} \quad (57)$$

$$\text{Precision} = \frac{1}{K} \sum_{k=1}^{K} p_k \quad (58)$$

$$\text{Recall} = \frac{1}{K} \sum_{k=1}^{K} r_k \quad (59)$$

$$F1 = \frac{1}{K} \sum_{k=1}^{K} F1_k \quad (60)$$

where K represents the total number of categories, and p_k, r_k, and $F1_k$ represent the precision, recall, and F1-score for category k, respectively. Precision p_k measures the proportion of texts correctly classified into category k among all texts classified into category k; higher p_k indicates more accurate classification for category k samples. Recall r_k measures the proportion of texts correctly classified into category k among all actual texts of category k; higher r_k indicates fewer missed samples in category k. $F1_k$ comprehensively considers precision and recall, with higher values indicating better classification effectiveness for category k.

3.4 Method Implementation Process

The patent automatic classification method process designed in this study consists of five steps: dataset construction, text preprocessing, text vectorization, model training and parameter tuning, and model classification effectiveness evaluation, as shown in Figure 3 [Figure 3: see original paper]:

Figure 3 Patent Automatic Classification Process

The specific explanations for each 环节 in Figure 3 are as follows:

1. **Dataset Construction.** Extract an appropriate number of labeled patents from the patent database as the original dataset, which is divided into two parts: training set and test set. The training set is used to train

the patent automatic classification model. To better train the model, a portion of the training set is divided as a validation set to cooperate with model training. The test set is used to evaluate the trained patent automatic classification model.

2. **Text Preprocessing.** This includes three steps: word segmentation, stop word removal, and part-of-speech tagging to remove specific word classes. For Chinese, characters are the smallest character units, while words are the smallest semantic units. Therefore, to enable models to process text from a semantic perspective and achieve better results, the first step in text preprocessing is to segment the patent text portion in the dataset. The word segmentation results of patent texts contain some noise words, such as special characters or function words without actual meaning. These words are removed through stop word removal and part-of-speech tagging to remove specific word classes (only retaining content words such as nouns, verbs, and adjectives).
3. **Text Vectorization.** The TF-IDF-based vector space model is used for text vectorization of the three traditional baseline models, while the Word2Vec-based word vector concatenation method is used for text vectorization of the seven deep learning models.
4. **Model Training and Parameter Tuning.** The ten patent automatic classification models described above are used for model training and parameter optimization, with the best-performing model on the validation set being retained.
5. **Model Classification Effectiveness Evaluation.** The ten classification models are tested on the test set to evaluate their performance on precision, recall, and F1-score metrics.

4 Experimental Results and Analysis

4.1 Classification Basis and Experimental Data

This study selects the "Section" of the patent IPC main classification number as the classification basis (the meaning of each section is shown in Table 2). 80,000 patent data entries are randomly extracted from the Chinese Academy of Sciences Patent Online Analysis System as the dataset, which is divided into three parts: 50,000 entries as the training set, 10,000 entries as the validation set, and 20,000 entries as the test set. The Word2Vec word vector model is trained on over 30 million Chinese patent data based on the CBOW model, with training parameters: size=300, min_{count}=40, window=10, sample=1e-3.

Table 2 Classification Basis

Section	Category
A	Human necessities

Section	Category
B	Performing operations; transporting
C	Chemistry; metallurgy
D	Textiles; paper
E	Fixed constructions
F	Mechanical engineering; lighting; heating; weapons; blasting
G	Physics
H	Electricity

4.2 Classification Results and Analysis

This study uses Mini-batch training. After experimental analysis, the final selection is: Mini-batch sample size of 200, word vector dimension of 300, recurrent neural network output dimension of 300, convolutional neural network output dimension of 300, and convolution kernel sizes of 2, 3, 4, and 5. The model iterates until the results on the validation set converge, and the best-performing model on the validation set is retained. The results are shown in Table 3 :

Table 3 Automatic Classification Results of Models

Model	Precision	Recall	F1-Score
Classical Machine Learning Models			
TFIDF+LR	0.7805	0.7784	0.7786
TFIDF+DT	0.5759	0.5740	0.5748
TFIDF+RF	0.7156	0.7117	0.7082
Deep Learning Models			
Word2Vec+ANN	0.7300	0.7301	0.7300
Word2Vec+TextCNN	0.8103	0.8075	0.8075
Word2Vec+GRU	0.8083	0.8091	0.8081
Word2Vec+BiGRU	0.8120	0.8117	0.8114
Word2Vec+BiGRU+TextCNN	0.8220	0.8183	0.8175
Word2Vec+ATT	0.7636	0.7626	0.7622
Word2Vec+BiGRU+ATT+TextCNN	0.8230	0.8243	0.8231

Sorted by model precision, Figure 4 [Figure 4: see original paper] shows the comparison of precision, recall, and F1-score for the ten models.

Figure 4 Comparison of Precision, Recall, and F1-Score of Ten Models

The following analysis and interpretation of the experimental results are provided:

1. **Deep learning models generally outperform classical machine learning models.** Except for Word2Vec+ANN and Word2Vec+ATT, the precision, recall, and F1 values of deep learning models are all above

0.8, while the metrics of the three classical machine learning models are all below 0.8. Since ANN has relatively weak feature representation capability and directly introducing the Attention mechanism on top of word vectors cannot well represent hidden features, Word2Vec+ANN and Word2Vec+ATT show the lowest performance among all deep learning models. However, their performance is still significantly better than TFIDF+DT and TFIDF+RF, indicating that feature extraction and enhancement on text vectors have a certain promoting effect on classification results.

2. **Considering contextual features and word order features has a positive effect on classification performance improvement.** The TextCNN model extracts and enhances contextual features of patent texts based on convolutional neural networks; the GRU model performs forward sequential modeling on patent texts, enhancing sequential features of the preceding context; the BiGRU model performs bidirectional modeling on patent texts, enhancing sequential features of both forward and backward contexts. The consideration of these features enables Word2Vec+TextCNN, Word2Vec+GRU, and Word2Vec+BiGRU to achieve metric scores above 0.8. On this basis, combining BiGRU and TextCNN models to consider bidirectional word order features while also considering contextual features achieves better model performance than using TextCNN and BiGRU alone.

3. **Introducing the Attention mechanism to enhance key features has a positive impact on classification results.** Among the ten automatic classification models, Word2Vec+BiGRU+ATT+TextCNN shows the best performance, indicating that while considering contextual features and bidirectional word order features, introducing the Attention mechanism to enhance key features can effectively improve patent text classification performance.

5 Discussion and Outlook

This study addresses the problem of Chinese patent multi-classification, designs seven patent automatic classification deep learning models based on TextCNN, GRU, Attention, and other technologies, and compares them with three traditional classical automatic classification models. The evaluation results show that deep learning models considering word order features, contextual features, and enhanced key features achieve better classification performance than traditional classification models for Chinese patent classification. Among them, the "Word2Vec+BiGRU+ATT+TextCNN" model demonstrates the optimal performance among the ten models, with the highest precision, recall, and F1-score.

In the context of the current national requirement to accelerate patent examination work, this model has reference significance and value for optimizing and improving the effectiveness of existing automatic classification methods and

tools, enhancing patent classification work efficiency, and shortening patent examination cycles.

However, this research still has room for improvement. Patent classification is a multi-label classification problem, but this study only conducted research on single-label multi-classification using the main classification number of patents. At the same time, IPC classification includes five levels: Section, Class, Subclass, Main Group, and Subgroup. More detailed category classification means a substantial increase in the number of categories, posing higher requirements for classification models. This study only conducted automatic classification research at the "Section" level, and more detailed level classification models need to be studied in the future. In addition, the methods proposed in this study can also be applied and verified on internationally published evaluation tasks and data for patent classification (such as the patent classification task in the NTCIR evaluation competition [13]), thereby expanding the influence and application scope of the research.

Deep learning technology is still developing. For example, Google' s BERT pre-trained language model released in 2018 has set new records in 11 NLP tasks, providing possibilities for further optimizing the classification performance of patent automatic classification models. In future work, we will continue to research patent automatic classification methods based on dynamic text representation models to achieve better classification performance.

References

[1] CCTV News. China's invention patent applications rank first in the world for 8 consecutive years [EB/OL]. [2019-08-02]. http://dy.163.com/v2/article/detail/EGM6VQSG0511A3UP.html.

[2] National Intellectual Property Administration. Annual status of domestic patent applications [EB/OL]. [2019-08-02]. http://www.cnipa.gov.cn/tjxx/jianbao/year2018/a/a3.html.

[3] Tian Chuang, Zhao Yajuan. A patent-industry category mapping model based on similarity—taking the International Patent Classification and National Economic Industry Classification as examples [J]. Library and Information Service, 2016, 60(20): 123-131.

[4] Lai S W, Xu L H, Liu K, et al. Recurrent convolutional neural networks for text classification [C]//Proceedings of the twenty-ninth AAAI conference on artificial intelligence. Austin: AAAI, 2015: 2267-2273.

[5] Yang Z C, Yang D Y, Dyer C, et al. Hierarchical attention networks for document classification [C]//Proceedings of the 2016 conference of the North American Chapter of the Association for Computational Linguistics: human language technologies. San Diego: NAACL, 2016: 1480-1489.

[6] Zhang X, Zhao J, LeCun Y, et al. Character-level convolutional networks for text classification [C]//Advances in neural information processing systems. Montreal: Neural information processing systems foundation, 2015: 649-657.

[7] Chen Y L, Chang Y C. A three-phase method for patent classification [J]. Information processing and management, 2012, 48(6): 1017-1030.

[8] Fall C J, Torcsvari A, Benzineb K, et al. Automated categorization in the international patent classification [J]. ACM SIGIR forum. Toronto: Association for Computing Machinery, 2003, 37(1): 10-25.

[9] Trappey A J C, Hsu F C, Trappey C V, et al. Development of a patent document classification and search platform using a back-propagation network [J]. Expert systems with applications, 2006, 31(4): 755-765.

[10] Hode A I B, Bortri, Wanka G. The Rose-Gurevitz-Fox approach applied for patent classification [J]. European journal of operational research, 2006, 173(3): 815-826.

[11] Krier M, Francois Z. Automatic categorisation applications at the European patent office [J]. World patent information, 2002, 24(3): 187-196.

[12] Koster C H A, Seutter M, Beney J. Multi-classification of patent applications with Winnow [C]//International Andrei Ershov memorial conference on perspectives of system informatics. Berlin: Springer Berlin Heidelberg, 2003: 546-555.

[13] Iwayama M, Fujii A, Kando N. Overview of classification subtask at NTCIR-5 patent retrieval task [C]//Proceedings of NTCIR-5 workshop meeting. Tokyo: NTCIR, 2005.

[14] Kim J H, Choi K S. Patent document categorization based on semantic structural information [J]. Information processing and management, 2007, 43(5): 1200-1215.

[15] He C, Loh H T. Grouping of TRIZ inventive principles to facilitate automatic patent classification [J]. Expert systems with applications, 2008, 34(1): 788-795.

[16] He C, Loh H T. Pattern-oriented associative rule-based patent classification [J]. Expert systems with applications, 2010, 37(3): 2395-2404.

[17] Hu Zhengyin, Fang Shu, Wen Yi, et al. Research on patent automatic classification for TRIZ [J]. New Technology of Library and Information Service, 2015, 31(1): 66-74.

[18] Zhai Jiqiang, Wang Keqi. Chinese patent automatic classification based on TRIZ invention principles [J]. Journal of Harbin University of Science and Technology, 2013, 18(3).

[19] Liu Longfan, Li Yan, Hou Chaoyi, et al. Experimental research on patent information mining and automatic classification based on functional basis [J]. Journal of Sichuan University (Engineering Science Edition), 2016, 48(5): 105-113.

[20] Zhang X Y. Interactive patent classification based on multi-classifier fusion and active learning [J]. Neurocomputing, 2014, 127: 200-205.

[21] Chang S B, Lai K K, Chang S M. Exploring technology diffusion and classification of business methods: using the Patent Citation Network [J]. Technological forecasting and social change, 2009, 76(1): 107-117.

[22] Lai K K, Wu S J. Using the Patent Co-Citation approach to establish a new patent classification system [J]. Information processing and management, 2005, 41(2): 313-330.

[23] Li Chengxiong, Ding Yuehua, Wen Guihua. Application of SVM-KNN combined improved algorithm in patent text classification [J]. Computer Engineering and Applications, 2006(20): 193-195, 229.

[24] Jia Shanshan, Liu Chang, Sun Lianying, et al. Research on patent automatic classification based on multi-feature multi-classifier integration [J]. Data Analysis and Knowledge Discovery, 2017, 1(8): 76-84.

[25] Verberne S, D'Hondt E. Patent classification experiments with the linguistic classification system LCS in CLEF-IP 2011 [C]//CLEF 2011 working notes. Amsterdam: CLEF, 2011.

[26] Stutzki J, Matthias S. Geo-data supported classification of patent applications [C]//Proceedings of the third international ACM SIGMOD workshop on managing and mining enriched geo-spatial data. San Francisco: Association for Computing Machinery, 2016: 1-6.

[27] Lim S, Kwon Y J. IPC multi-label classification based on the field functionality of patent documents [C]//SIGIR Forum. Gold Coast: Association for Computing Machinery, 2016: 677-691.

[28] Ma Shuanggang. Research on Chinese patent text automatic classification based on deep learning theory and methods [D]. Suzhou: Jiangsu University, 2016.

[29] Hu Jie, Li Shaobo, Yu Liya, et al. Patent text classification model based on convolutional neural network and random forest algorithm [J]. Science Technology and Engineering, 2018, 18(6): 268-272.

[30] Ma Jianhong, Wang Ruiyang, Yao Shuang, et al. Patent classification method based on deep learning [J]. Computer Engineering, 2018, 44(10): 215-220.

[31] Li S B, Hu J, Cui Y X, et al. DeepPatent: Patent classification with convolutional neural networks and word embedding [J]. Scientometrics, 2018, 117(2): 721-744.

[32] Xiao Lizhong, Wang Guangzhong, Liu Yuan, et al. Classification method for security domain patent texts [P]. China: 109033402A. 2018-12-18.

[33] Kim Y. Convolutional Neural Networks for Sentence Classification [C]//Proceedings of the 2014 conference on empirical methods in natural language processing. Doha: EMNLP, 2014: 1746-1751.

[34] Cho K, Merrienboer B V, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation [C]//Proceedings of the 2014 conference on empirical methods in natural language processing. Doha: EMNLP, 2014: 1724-1734.

[35] Chung J, Gulcehre C, Cho K, et al. Empirical evaluation of gated recurrent neural networks on sequence modeling [C]//NIPS 2014 deep learning and representation learning workshop. arXiv:1412.3555. Montreal: NIPS, 2014.

[36] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need [C]//arXiv:1706.03762. Long Beach: NIPS, 2017.

Author Contributions

Lü Lucheng: Responsible for paper framework design, writing, and revision;
Han Tao: Responsible for research scheme design and optimization;
Zhou Jian: Conducted model training and effectiveness evaluation;
Zhao Yajuan: Responsible for paper topic selection and design, and provided revision suggestions.

Note: Figure translations are in progress. See original paper for figures.