symmetry

MDPI

Article

Patent Automatic Classification Based on Symmetric Hierarchical Convolution Neural Network

Huiming Zhu [1,2,*], Chunhui He [1,*], Yang Fang [1], Bin Ge [1], Meng Xing [3] and Weidong Xiao [1]

[1] Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha 410073, China; fangyang12@nudt.edu.cn (Y.F.); gebin@nudt.edu.cn (B.G.); wdxiao@nudt.edu.cn (W.X.)
[2] Department of Economics and Trade, Changsha Commerce&Tourism College, Hunan 410116, China
[3] Department of Information and Communication, Army Academy of Armored Forces, Beijing 100000, China; xingmykx@163.com
* Correspondence: zhuhuiming_1983@163.com (H.Z.); xtuhch@163.com (C.H.)

Received: 17 December 2019; Accepted: 13 January 2020; Published: 21 January 2020

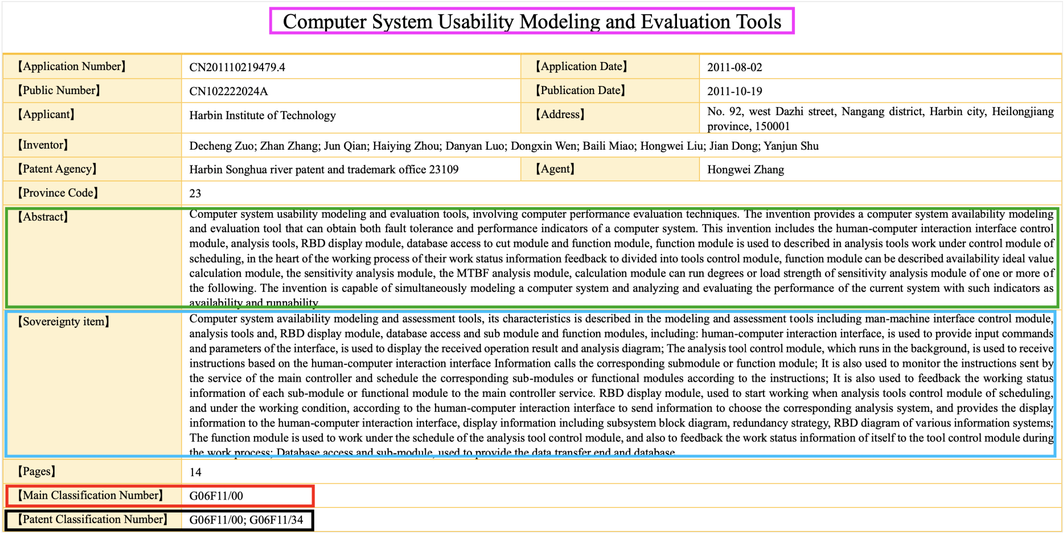
Abstract: With the rapid growth of patent applications, it has become an urgent problem to automatically classify the accepted patent application documents accurately and quickly. Most previous patent automatic classification studies are based on feature engineering and traditional machine learning methods like SVM, and some even rely on the knowledge of domain experts, hence they suffer from low accuracy problem and have poor generalization ability. In this paper, we propose a patent automatic classification method via the symmetric hierarchical convolution neural network (CNN) named PAC-HCNN. We use the title and abstract of the patent as the input data, and then apply the word embedding technique to segment and vectorize the input data. Then we design a symmetric hierarchical CNN framework to classify the patents based on the word embeddings, which is much more efficient than traditional RNN models dealing with texts, meanwhile keeping the history and future information of the input sequence. We also add gated linear units (GLUs) and residual connection to help realize the deep CNN. Additionally, we equip our model with a self attention mechanism to address the long-term dependency problem. Experiments are performed on large-scale datasets for Chinese short text patent classification. Experimental results prove our proposed model's effectiveness, and it performs better than other state-of-the-art models significantly and consistently on both fine-grained and coarse-grained classification.

Keywords: patent classification; short-text classification; symmetric hierarchical CNN

1. Introduction

As an effective technology carrier, patent documents have been widely used in real-life applications. According to the World Intellectual Property Organization's (WIPO) report on intellectual property statistics, the number of global patent applications is rapidly increasing [1]. With such a dramatic growth of patent applications, relying on manual work to complete patent classification takes a lot of time and money, and is thus unable to meet the real needs. Therefore, lots of patent automatic classification (PAC) algorithms have been proposed. Some works focus on patent text representations, e.g., in order to identify a technical patent topic from a patent abstract; the authors of [2] combined semantic role labeling (SRL) information with framework rules of semantics, which improves the performance in SRL systems by obtaining knowledge from patents. Some works focus on the extraction of semantic features from patent texts, e.g., the authors of [3] extracted keywords related to technical information from patent documents, constructed keyword collections, and obtained the layer of keywords according to the level of information. Additionally, machine learning has drawn more attention for the PAC system, e.g., the authors of [4] developed an automatic patent quality analysis and classification system SOM-KPCA-SVM, using data mining methods to identify and classify the quality of new patents in a timely manner. However, most of the studies are highly dependent on feature engineering, which are unable to quickly and automatically classify the ’sections’ and ’classes’ into large-scale patent datasets tagged by international patent classification (IPC) numbers. Moreover, with the gradual refinement of patent categories and the increase of the number of categories, these PAC algorithms have an obvious reduction in classification accuracy.

Here we first introduce the detailed information of a patent. As shown in Figure 1, a patent document is a special text, which contains invention name, applicant, inventor, abstract, claims, classification number and other contents. In addition to the applicants and inventors, the rest of them usually show strong characteristics of domain knowledge, which are suitable for constructing the features for PAC algorithms. Therefore, the classification efficiency can be effectively improved by selecting appropriate features. In Figure 1, the purple box represents the title of the patent, the green box represents the abstract, the blue box represents the sovereign item, the red box represents the main classification number, and the brown box represents the patent classification number (all classification numbers). This paper mainly studies the fine-grained classification tasks of the main classification number, instead of analyzing all patent classification numbers.



Computer System Usability Modeling and Evaluation Tools

【Application Number】	CN201110219479.4	【Application Date】	2011-08-02
【Public Number】	CN102222024A	【Publication Date】	2011-10-19
【Applicant】	Harbin Institute of Technology	【Address】	No. 92, west Dazhi street, Nangang district, Harbin city, Heilongjiang province, 150001
【Inventor】	Decheng Zuo; Zhan Zhang; Jun Qian; Haiying Zhou; Danyan Luo; Dongxin Wen; Baili Miao; Hongwei Liu; Jian Dong; Yanjun Shu		
【Patent Agency】	Harbin Songhua river patent and trademark office 23109	【Agent】	Hongwei Zhang
【Province Code】	23		
【Abstract】	Computer system usability modeling and evaluation tools, involving computer performance evaluation techniques. The invention provides a computer system availability modeling and evaluation tool that can obtain both fault tolerance and performance indicators of a computer system. This invention includes the human-computer interaction interface control module, analysis tools, RBD display module, database access to cut module and function module, function module is used to described in analysis tools work under control module of scheduling, in the heart of the working process of their work status information feedback to divided into tools control module, function module can be described availability ideal value calculation module, the sensitivity analysis module, the MTBF analysis module, calculation module can run degrees or load strength of sensitivity analysis module of one or more of the following. The invention is capable of simultaneously modeling a computer system and analyzing and evaluating the performance of the current system with such indicators as availability and runnability		
【Sovereignty item】	Computer system availability modeling and assessment tools, its characteristics is described in the modeling and assessment tools including man-machine interface control module, analysis tools and, RBD display module, database access and sub module and function modules, including: human-computer interaction interface, is used to provide input commands and parameters of the interface, is used to display the received operation result and analysis diagram; The analysis tool control module, which runs in the background, is used to receive instructions based on the human-computer interaction interface Information calls the corresponding submodule or function module; It is also used to monitor the instructions sent by the service of the main controller and schedule the corresponding sub-modules or functional modules according to the instructions; It is also used to feedback the working status information of each sub-module or functional module to the main controller service. RBD display module, used to start working when analysis tools control module of scheduling, and under the working condition, according to the human-computer interaction interface to send information to choose the corresponding analysis system, and provides the display information to the human-computer interaction interface, display information including subsystem block diagram, redundancy strategy, RBD diagram of various information systems; The function module is used to work under the schedule of the analysis tool control module, and also to feedback the work status information of itself to the tool control module during the work process; Database access and sub-module, used to provide the data transfer end and database		
【Pages】	14		
【Main Classification Number】	G06F11/00		
【Patent Classification Number】	G06F11/00; G06F11/34		

Figure 1. The example structure of patent CN102222024A [5].

In classification tasks, if the classifier is trained using the whole content of the patent documents, it would be too time-consuming with a much higher complexity of the model. Therefore, it needs to make a balance between the accuracy and efficiency in practical applications. Based on the analysis of patent text, it is found that the training of the automatic patent classifier can be completed by extracting a few representative contents of patent documents, which can improve the PAC efficiency while maintaining the accuracy of PAC. Relevant research results show that satisfactory accuracy can be achieved by extracting the contents of patent abstract [6] or claims [7] in the tasks of patent classification. Therefore, this paper uses the titles and abstracts of patent documents to form the input data for training, and then uses the corresponding ’sections’ or ’classes’ labels to construct the target samples for PAC.

Specifically, to realize the fast PAC task, we use the word embedding technique to pre-process the input text. First we adopt the jieba Chinese segmentation model (https://github.com/fxsjy/jieba) to segment the text data, and then apply the gensim [8] framework to project the words into low-dimensional vectors. Then we introduce a symmetric hierarchical convolution neural network (CNN) framework to encode the word embeddings for classification. Conventional CNN is only able to capture a fixed size context; however, our hierarchical CNN can easily enlarge the context by stacking

CNN layers over each other symmetrically from the head and tail of the input. Through stacking CNN layers symmetrically, hierarchical representations on the input are created, that is, lower layers will process close words in sequences while higher layers process distant words [9], and using symmetric structure enables the proposed model to preserve both the history and future information. We equip our framework with gated linear units (GLUs) [10] and residual connection [11] to realize a deep CNN architecture. We also utilize the self-attention mechanism to address the long-term dependency problem when dealing with the long input sequence. This can help our model focus on the significant parts in the sequence for classification.

The main contributions of this study are as follows:

1. To improve the efficiency of automatic patent classification, the model is trained by combining the short text of patent name and abstract.
2. After pre-processing the input data by word embedding method, for further classification, we introduce a hierarchical CNN framework to create the representations of the input text.
3. We conduct the PAC experiments in five ’sections’ and 50 ’classes’ on large-scale real Chinese patent datasets. The experiment results verify that our model based on hierarchical CNN outperforms other advanced alternatives consistently and statistically significantly on both fine-grained and coarse-grained classification.

2. Related Work

2.1. International Patent Classification System

The International Patent Classification List is a complete patent classification system [12], developed by the WIPO. Since its publication from the last century, it has made outstanding contributions to the management of global patent documents. The use of IPC number for retrieval does not rely on keywords, meanwhile, it makes it possible for patent documents of various countries to obtain a unified classification number. Currently, IPC has 12 languages including Chinese, English, and German. IPC is a hierarchical structure, consisting of section, class, sub-class, group, and sub-group. At each sub-level of the hierarchy, the number of categories is multiplied by about 10, so the IPC contains approximately 72,000 categories in total [13]. For China’s IPC number, it is slightly different from the international IPC number. According to different technical fields, the section number is divided into eight sections from A: Human necessities to H: Electricity. Taking the Chinese patent IPC number G06F11/00 in Figure 1 as an example, G, G06, G06F and G06F11/00 represent the corresponding section, Class, Subclass and Group, respectively. Most works try to classify specific sections and classes in IPC automatically, whereas the researches on the classification task of all sections and classes are still limited.

The ultimate goal of the PAC study is to enable computer systems to accurately find IPC tags for patent applications, just like patent examiners. However, due to the complexity of the IPC system, the current existing PAC systems are not ideal.

2.2. PAC Algorithms

To build a PAC system, TRIZ (the Russian acronym of theory of inventive problem solving) oriented personalized classification method was proposed based on subject model [14]. The task of automatic patent classification can be accomplished by combining different feature items with machine learning algorithm and using feature dimension reduction to optimize the classifier. Grouping similar patents according to TRIZ user’s invention principle was proposed to solve the problem of multi-label classification and class imbalance [15]. An automatic patent classification method was proposed based on the integration of multi-feature and multi-classifier [16]. Different dictionaries and paragraph vector features are used to train Bayesian and support vector machine (SVM) classifiers respectively, with the final patent classification completed with the feature-category matrix. M3-SVM attempted to Decompose the problem of large-scale unbalanced patent classification into a set of relatively small and more balanced sub-problems, and min-max modular support vector machine is then used to handle the subproblems [17]. Later, an artificial intelligence-assisted patent decision-making method was proposed [18]. A patent classification system was developed by combining expert screening and hybrid genetic support vector machine (HGA-SVM) model.

In recent years, with the improvement of hardware performance and the popularity of deep learning algorithms, some researches have applied CNN to text categorization tasks. CNN model was used to classify short text sentences [19]. Considering that short text often encounters data sparsity and fuzziness problem in representation due to the lack of context, a short text classification method was proposed combining semantic clustering and CNN [20]. This method introduces a fast semantic clustering algorithm through embedding layer, and uses supervised pattern to detect multi-scale semantic units, using external knowledge. Because of the complex and time-consuming process of text categorization, Facebook's open-source sentence categorization method and word feature learning model were proposed, Fasttext, on the field of Chinese text categorization and verified its effectiveness [21]. Compared with the current widely-used text classification methods, Fasttext can greatly reduce the computing time while guarantee the classification results.

However, those CNN based models like Fasttext can not effectively comprehend the text content, comparing with recurrent neural network (RNN) that processes a text word by word [22]. RNN still suffers from efficiency problem, so we propose a hierarchical CNN to balance the efficiency and effectiveness of the model.

3. CNN Framework

CNN has been widely used in the areas of speech recognition, image processing and natural language processing, et al. We construct a hierarchical CNN Model PAC-HCNN based on the Tensorflow platform [23]. The whole PAC-HCNN framework consists of six Layers: Input layer, preprocessing layer, word embedding layer, hierarchical convolution layer, fully connected layer, and output layer. The framework overview of the proposed framework is shown in Figure 2.

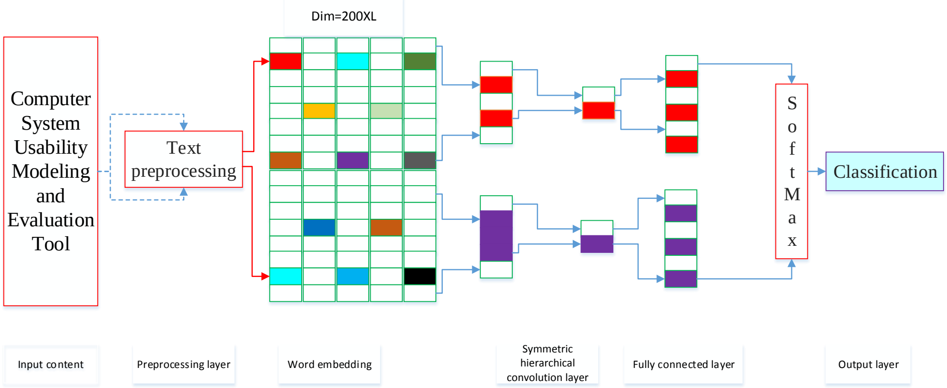


Figure 2. Framework of the automatic classification method via the symmetric hierarchical convolution neural network (PAC-HCNN).

As shown in Figure 2, we first pre-process the original text content at the input level. We use jieba segmentation model to segment the input text into separate words or phrases, meanwhile, removing the stop words. Then we adopt the gensim framework to project these words or phrases into low-dimensional vector, that is, the word embedding layer generates the representation of the valid words after pre-processing. We will illustrate the pre-process procedure in detail in Section 4.2. We denote the i-th effective word in the input content as x_i. Assuming that the input content contains m valid words, then the entire input sequence is $\mathbf{x} = (x_1, \ldots, x_m)$.

Before applying on convolutional layer, we add position embeddings to the above word embeddings. For the input sequence $\mathbf{x} = (x_1, \ldots, x_m)$, we first embed it as a low dimensional vector $\mathbf{u} = (u_1, \ldots, u_m)$, where $u_j \in \mathbb{R}^f$ and f is the dimension of the vector. We then embed the absolute position of input elements $\mathbf{p} = (p_1, \ldots, p_m)$ where $p_j \in \mathbb{R}^f$. By doing so, it assigns our model the ability to be sensitive to the part being processed of the input. We combine both embeddings to construct the final input embeddings $\mathbf{e} = (u_1 + p_1, \ldots, u_m + p_m)$.

In convolution layer, we introduce a symmetric hierarchical structure to flexibly encode the input embeddings. We represent the output of $l - th$ layer as $\mathbf{h^l} = (h_1^l, \ldots, h_m^l)$. A non-linearity and a convolution with one dimension form a layer. With kernel width being k in a layer, the output of the layer h_i^1 will process k elements of the input. Through stacking blocks over each other symmetrically from head and tail part of the input sequence, the length of input elements could be enlarged. For instance, if we stack six blocks using a kernel width k of 5, 25 elements of the input will be processed. The non-linearity could be used to adjust the proposed model to handle only a few elements or the whole sequence when needed. Unlike the chain structure of RNN, our CNN framework with multiple and symmetric layers is able to parallelly represent a sequence with long range, meanwhile keeping the history and future information simultaneously. Specifically, given a input sequence having n elements, our symmetric hierarchical CNN needs $O(\frac{n}{k})$ operations, while $O(n)$ operations are needed for RNN model. Additionally, in our CNN based framework, we have fixed non-linearities and kernel numbers, while for RNN model, n operations are needed to process the first words, and a single set of operations are needed for the last word [24]. It is obvious that comparing traditional CNN, our symmetric and hierarchical framework is not that efficient, since our model has to stack several layers while in traditional CNN, only one layer is adopted to process a sequence. Therefore, our framework actually makes a balance between efficiency and effectiveness.

In each convolution kernel, parameters are $W \in \mathbb{R}^{2f \times of}$, $b_w \in \mathbb{R}^f$. $X \in \mathbb{R}^{o \times f}$ denotes the input, in which o denotes the number of input elements and f is the dimension. After mapping the input via the convolution layer, we could obtain the output $Y \in \mathbb{R}^{2f}$. Note that the dimension of the output is twice of that of the input. Afterwards, our model will feed the o output elements to the following layers. For non-linearities, we apply them on the output using GLUs [10], denoted as $Y = [A\ B] \in \mathbb{R}^{2f}$:

$$v([A\ B]) = A \otimes \sigma(B), \tag{1}$$

where $A, B \in \mathbb{R}^f$ are the inputs of GLUs and $\otimes$ represents the element-wise multiplication. $\sigma(B)$ denotes the activation function which could decide to handle which part of the inputs during the time being. Residual connections [11] are also adopted so as to form a deeper CNN structure by connecting the input and the output of each CNN layer:

$$h_i^l = v(W^l[h_{i-o/2}^{l-1}, \ldots, h_{i+o/2}^{l-1}] + b_w^l) + h_i^{l-1}. \tag{2}$$

We then leverage a self-attention mechanism to further encode the text as such mechanism is able to handle the long-term dependency problem by assigning higher weights on the words important for final classification. Specifically, to calculate the attention in each layer, given $\mathbf{h}^l = \{h_1^l, h_2^l, ..., h_n^l\}$, we first compute the relativity score between every input element as follows:

$$f(\mathbf{h}^l) = \tanh(\mathbf{h}^l W_{ai}^l \mathbf{h}^{l^T} + b_{ai}^l)). \tag{3}$$

Then the attention weight between every could be computed as:

$$A_w^l = \text{softmax}(f(\mathbf{h}^l)). \tag{4}$$

Afterwards the final representation vectors of each word is represented as:

$$\mathbf{g}^l = A_w^l h = g_1^l, g_2^l, ..., g_n^l \tag{5}$$

After all the operations above, the final probability of a patent belonging to a class could be generated as follows:

$$p = \mathrm{softmax}(W_f g_i^L + b_f) \in \mathbb{R}^T, \tag{6}$$

where W_f denotes the weights in the fully connected layer, L is the final layer and b_f is the bias.

4. Data Collection and Preprocessing

4.1. Data Collection

We have two datasets: (1) Published dataset of small-scale Chinese patent documents with more than 3000 original Chinese patents; (2) Data crawled from the published Chinese National Knowledge Infrastructure (CNKI) patent database (http://cnki.scstl.org/kns55/brief/result.aspx?dbPrefix=SCPD) using the open source Python web crawler framework Scrapy [25], with about 48,000 original patent documents collected in total. The entire dataset contains about 51,000 patent documents. After the data is deduplicated, denoised, and limited within section D/E/F/G/H, 29,457 valid patent documents were obtained. In our experiments, the above 29,457 documents were divided with 80% for training, 15% for validation, and 5% for independent testing.

4.2. Data Pre-Processing

The eight types of patent documents from A to H sections are automatically tagged with the original input content. The automatic tagging process is divided in three ways.

1. The first way is to use patent names as the original input for classification. We use regular expression to directly extract the name and the corresponding classification number from a patent document, i.e., forming name–sections (e.g., H) tag data.
2. The second way is to use patent abstracts as the original input for classification. We use regular expression to directly extract the abstract and the corresponding classification number from a patent document, i.e., forming abstract–sections (e.g., H) tag data.
3. The third way is to use patent names and abstracts as the original inputs for classification. We use regular expression to directly extract the name, abstract and the corresponding classification number from a patent document, thereby forming the name and abstract–section (e.g., H)/Class(e.g., H01) tag data.

Based on the above processing steps, we can obtain all the tagged corpus. Before training the model, the original text needs to be pre-processed. The pre-processing part mainly includes the following three parts:

1. Firstly, the original text of the input is processed as a word segmentation task. In the experiment, the word segmentation of the text is completed by jieba participle module of the Python version.
2. Secondly, the common stop words collected on the Internet and the high-frequency words with low degree of discrimination in the patent literature are combined to form a stop word list, which is used to remove the stop word after the word segmentation, meanwhile reducing the feature dimension.
3. Thirdly, a vectorized representation of words is implemented by combining word embedding techniques. In the experiments, the open source gensim [8] word vector training tool is used to pre-train all the experimental datasets, obtaining the corresponding embeddings.

5. Experimental Results and Analysis

5.1. Datasets and Evaluation Metrics

5.1.1. Datasets

According to Section 4.1, we have 29,457 valid patent documents containing D/E/F/G/H labels in total. According to the pre-processing step in Section 4.2, we construct 4 sub-datasets for experiments.

1. Dataset$_{\text{\_name}}$ is used to examine the performance of algorithms on section classification which only use the patent names as input.
2. Dataset$_{\text{\_abstract}}$ is used to examine the performance of algorithms on section classification that only use patent abstracts as input.
3. S_Dataset$_{\text{\_name+abstract}}$ is used to examine the performance of algorithms on section classification that combines the patent name and abstract as input.
4. C_Dataset$_{\text{\_name+abstract}}$ is used to examine the performance of algorithms on class classification that combines the patent name and abstract as input.

The basic information of the four datasets is shown in Table 1.

In Table 1, the terms '#Train', '#Val', '#Test' and '#Label' represent the number of documents in training set, the number of documents in validation set, the number of documents in test set and the number of label tags, respectively.

Table 1. Datasets statistic information.

Dataset Name	#Train	#Val	#Test	#Label
Dataset$_{\text{\_name}}$	23,565	4418	1474	5
Dateset$_{\text{\_abstract}}$	23,565	4418	1474	5
S_Dataset$_{\text{\_name+abstract}}$	23,565	4418	1474	5
C_Dataset$_{\text{\_name+abstract}}$	23,565	4418	1474	50

5.1.2. Evaluation Metrics

We use Precision, Recall, and F-measure (F_1) to evaluate the performance of the patent classification algorithms. Precision represents the accuracy of the classification, and Recall represents the recall rate of the classification. Precision and Recall rates are two widely used metrics in information retrieval. To evaluate the algorithm's performance in a comprehensive manner, we also adopt F-Measure metric, which is the weighted harmonic averaging result of Precision and Recall. The F1 metric is defined as follows:

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall}. \tag{7}$$

A higher F_1 indicates a better performance. With the number of experimental samples getting larger and the distribution of the dataset getting more balanced, the credibility of the evaluation indicators will correspondingly become higher. Finally, by calculating the average of each indicator, we can obtain the overall performance of the classification algorithm on the entire sampled dataset.

5.2. Experimental Results on Coarse-Grained Classification

In order to examine which part of the patent document is the most valuable part for patent classification, experiments are performed on three different datasets Dateset$_{\text{\_name}}$, Dateset$_{\text{\_abstract}}$ and S_Dateset$_{\text{\_name+abstract}}$ using PAC-HCNN model. We only classify the patent section with only 5 targets, so it is a coarse-grained classification. Hyper-parameters that we set up are shown in Table 2.

Table 2. Hyper-parameter settings of the PAC-HCNN Model.

Parameter Name	Value	Parameter Name	Value
Num_Classes	5	Kernel_size	5
Num_filters	128	Vocab_size	5000
Learning_rate	1×10^{-3}	Hidden_dim	128
Batch_size	256	Num_epochs	20

The model is then tested using the corresponding testing datasets. The detailed results are shown in Table 3.

Table 3. Experiment results of PAC-HCNN on coarse-grained classification.

Dataset Name	Average P(%)	Average R(%)	Average F_1(%)
Dataset$_{\_name}$	80.3	82.1	81.2
Dateset$_{\_abstract}$	83.7	84.1	83.9
S_Dataset$_{\_name+abstract}$	**85.7**	**87.8**	**86.7**

From Table 3, we observe that only using the patent name caused the worst coarse-grained classification performance. The average P value is 80.3%, while the average R and F_1 value are 82.1% and 81.2%, respectively. Only using the patent abstract had a slightly better performance. The average P, R and F_1 value are 83.7%, 84.1% and 83.9%, respectively. Using the combination of the name and abstract had the best result, with the average P, R and F_1 values reaching 85.7%, 87.8% and 86.7%. This demonstrates that considering more information of the patent can correspondingly improve the experimental performance.

5.3. Comparing with Other Models

In order to comprehensively examine the classification performance of PAC-HCNN, we compare it with seven state-of-the-art algorithms including Fasttext, gated recurrent unit (GRU), multinomial naive Bayesian (MNB), logistic regression (LR), support vector machines (SVM), random forest (RF) and multi-layer perceptron neural networks (MPLNN). For those models using conventional CNN, we only adopt Fasttext as it achieved the best results reported in its original paper. Except for the PAC-HCNN, Fasttext and GRU, the other models are implemented using the corresponding algorithms in the Scikit-learn [26] toolkit. We conduct both coarse-grained and fine-grained classification. For fine-grained classification, we use the C_Dateset$_{\_name+abstract}$ in which patent class having 50 category labels. In all the experiments, we use the average F_1 value as a comprehensive evaluation metric. We also adopt a paired two-tailed t test with $p < 0.05$ between our model and the other models. The experimental results are shown in Figures 3 and 4.

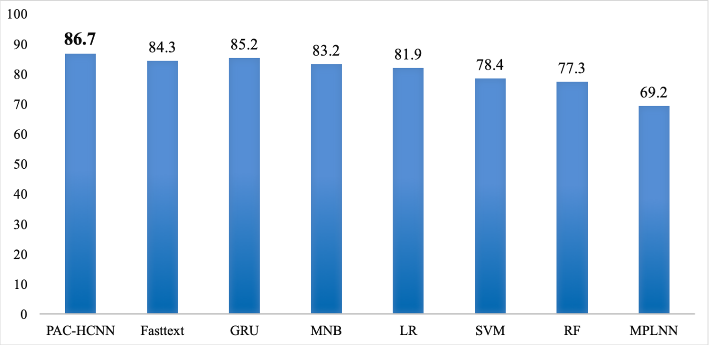


Figure 3. Average F_1 of different models on coarse-grained classification.

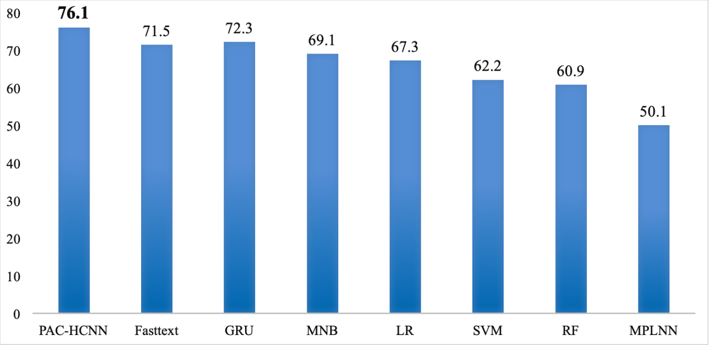


Figure 4. Average F_1 of different models on fine-grained classification.

From Figure 3, we observe that deep learning methods PAC-HCNN, Fasttext and GRU achieve better results comparing those traditional machine learning methods. GRU outperforms Fasttext which proves that RNN based model could comprehend the patent text better than traditional CNN based model. However, PAC-HCNN obtains the best result and we attribute it to the use of symmetric hierarchical CNN which promotes the patent text representation ability even better than GRU. From Figure 4, we can find the same tendency which further proves the effectiveness of PAC-HCNN. Specifically, we find that the margin between PAC-HCNN and other models are bigger on fine-grained classification. For example, on coarse-grained classification, PAC-HCNN is 2.8% better than GRU while on fine grained it is 6.4% better. It proves that our hierarchical architecture is more suitable for fine-grained task. In conclusion, PAC-HCNN outperforms other alternatives consistently and statistically significantly on both coarse-grained and fine-grained tasks.

5.4. Performance Variation on Different Sections

We conduct an additional experiment to further examine the performance of PAC-HCNN on different sections. The results are shown in Figure 5. From which we could observe that our model shows considerable difference on the classification performance on different sections, especially on section D being the worst. It is probably caused by the distribution imbalance of the sampled data in different sections of the experimental dataset.

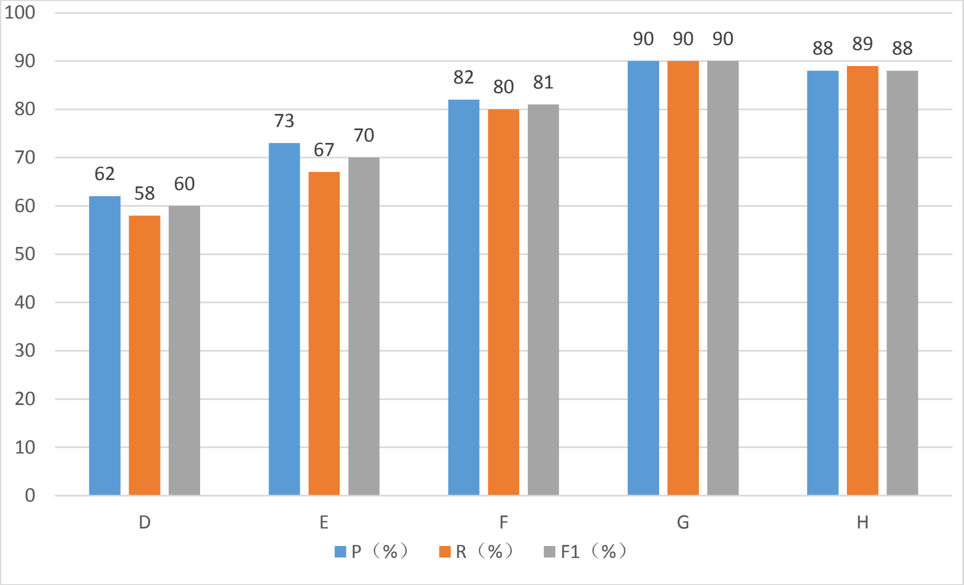


Figure 5. Experimental results of the PAC-CNN model on the dataset of five sections.

5.5. Computation Cost

We evaluate the computation cost of our model, GRU and Fasttext, as GRU and Fasttext achieved comparative results which are much better than other alternatives. We extract a subset of C_Dateset$_{\text{name+abstract}}$ with 100 samples. The device we use to evaluate the efficiency issue is GTX-1080 GPU. Table 4 presents the experimental results. From Figures 3 and 4 and Table 4, we observe that our model not only outperforms GRU model which is based on RNN, but also being more efficient. We attribute it to the use of our hierarchical CNN which can parallelly process the input sequence to improve the efficiency, meanwhile boosting the performance via the deep structure stacking convolutional layers. However, our model is not that efficient comparing Fasttext, while the gap is not very large. This is due to that our model uses deeper nerual networks than Fasttext, achieving much better results but sacrificing acceptable time cost.

Table 4. The computation cost.

Model	Times
PAC-HCNN	1873
GRU	2837
Fasttext	1674

6. Discussion

Our model has a lot of potential and real-life applications. Accurately classifying the patents into a fine-grained field could help researchers find the most related patents and filter out other noise. For example, in a medical research, it could offer a shortcut access to a patent focusing on a certain disease or a certain operation.

7. Conclusions

In this paper, we propose a symmetric hierarchical CNN framework to realize IPC classification. We first extract the patent name and abstract to form the training datasets and leverage the word embedding technique to transform the text into low-dimensional vectors. Afterwards we propose hierarchical architecture stacking convolutional layers to represent the input word sequence; meanwhile promoting efficiency by comparing those RNN based model. We also incorporate the GLU mechanism and residual connection with the framework to realize a deep CNN. The self-attention mechanism is also introduced to handle the long-term dependency problem. We then conduct extensive experiments on real-life patent datasets, the experimental results verify the effectiveness and efficiency of our model for IPC number classification.

Author Contributions: H.Z. conducted the conceptualization and proposed a new methodology, she also wrote the paper and conducted the experiments; C.H. conducted the formal analysis, coding and data curation; Y.F. mainly reviewed and edited the paper; M.X. conducted results validation; B.G. conducted the project administration; W.X. was the supervisor and helped the funding acquisition. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by NSFC under grants Nos. 61872446, 61902417, 71971212 and 71690233, and NSF of Hunan province under grant No. 2019JJ20024.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Neuhäusler, P.; Rothengatter, O.; Frietsch, R. Patent Applications-Structures, Trends and Recent Developments 2018. Technical Report, Studien Zum Deutschen Innovationssystem. 2019. Available online: https://www.econstor.eu/bitstream/10419/194274/1/1067672206.pdf (accessed on 15 January 2020).

2. Meng, L.; He, Y.; Li, Y. Research of Semantic Role Labeling and Application in Patent Knowledge Extraction. In Proceedings of the First International Workshop on Patent Mining And Its Applications (IPaMin 2014) Co-Located With Konvens 2014, Hildesheim, Germany, 6–7 October 2014.
3. Roh, T.; Jeong, Y.; Yoon, B. Developing a Methodology of Structuring and Layering Technological Information in Patent Documents through Natural Language Processing. *Sustainability* **2017**, *9*, 2117. [CrossRef]
4. Wu, J.; Chang, P.; Tsao, C.; Fan, C. A patent quality analysis and classification system using self-organizing maps with support vector machine. *Appl. Soft Comput.* **2016**, *41*, 305–316. [CrossRef]
5. Zuo, D.; Zhang, Z.; Qian, J.; Zhou, H.; Luo, D.; Wen, D.; Miao, B.; Liu, H.; Dong J.; Shu Y. Computer system usability modeling and evaluation tool. Chinese CN102222024A, 2011.10.19. Available online: http://dbpub.cnki.net/grid2008/dbpub/detail.aspx?dbcode=SCPD&dbname=SCPD2011&filename=CN102222024A&uid=WEEvREcwSlJHSldRa1FhcTdnTnhXM281cEdHa0o5bTQ1ZHlUd3YrcTFwND0=\protect\T1\textdollar9A4hF_YAuvQ5obgVAqNKPCYcEjKensW4IQMovwHtwkF4VYPoHbKxJw!! (accessed on 15 January 2020).
6. Liao, J.; Jia, G.; Zhang, Y. The rapid automatic categorization of patent based on abstract text.Information Studies. *Inf. Stud. Theory Appl.* **2016**, *39*, 103–105. [CrossRef]
7. Xu, D. Fast Automatic Classification Method of Patents Based on Claims. *J. Libr. Inf. Sci.* **2018**, *3*, 72–76. [CrossRef]
8. Heuer, H. Text Comparison Using Word Vector Representations and Dimensionality Reduction. *CoRR* **2016**. Available online: http://xxx.lanl.gov/abs/1607.00534 (accessed on 15 January 2020).
9. Gehring, J.; Auli, M.; Grangier, D.; Yarats, D.; Dauphin, Y.N. Convolutional Sequence to Sequence Learning. In Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017; pp. 1243–1252.
10. Dauphin, Y.N.; Fan, A.; Auli, M.; Grangier, D. Language Modeling with Gated Convolutional Networks. In Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017; pp. 933–941.
11. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]
12. Li, P. The Predicament and Outlet of International Patent Classification. *China Invent. Pat.* **2009**, *8*, 75–79. [CrossRef]
13. Jie, H.; Li, S.; Hu, J.; Yang, G. A Hierarchical Feature Extraction Model for Multi-Label Mechanical Patent Classification. *Sustainability* **2018**, *10*, 219. [CrossRef]
14. Hu, Z.; Fang, X.; Wen, Y.; Zhang, X.; Liang, T. Study on Automatic Classification of Patents Oriented to TRIZ. *Data Anal. Knowl. Discov.* **2015**, *31*, 66–74.
15. He, C.; Loh, H.T. Grouping of TRIZ Inventive Principles to facilitate automatic patent classification. *Expert Syst. Appl.* **2008**, *34*, 788–795. [CrossRef]
16. Jia, S.; Liu, C.; Sun, L.; Liu, X.; Peng, T. Patent Automatic Classification Based on Multi-Feature Multi-Classifier Integration. *Data Anal. Knowl. Discov.* **2017**, *1*, 76–84.
17. Chu, X.; Ma, C.; Li, J.; Lu, B.; Utiyama, M.; Isahara, H. Large-scale patent classification with min-max modular support vector machines. In Proceedings of the International Joint Conference on Neural Networks, IJCNN 2008, Part of the IEEE World Congress on Computational Intelligence, WCCI 2008, Hong Kong, China, 1–6 June 2008; pp. 3973–3980. [CrossRef]
18. Wu, C.; Ken, Y.; Huang, T. Patent classification system using a new hybrid genetic algorithm support vector machine. *Appl. Soft Comput.* **2010**, *10*, 1164–1177. [CrossRef]
19. Kim, Y. Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, A Meeting of SIGDAT, a Special Interest Group of the ACL, Doha, Qatar, 25–29 October 2014; pp. 1746–1751.
20. Lee, J.Y.; Dernoncourt, F. Sequential Short-Text Classification with Recurrent and Convolutional Neural Networks. In Proceedings of the NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 515–520.
21. Dai, L.; Jiang, K. Chinese Text Classification Based on FastText. *Comput. Mod.* **2018**, *5*, 35–40. [CrossRef]

22. Sun, L.; Cao, B.; Wang, J.; Srisa-an, W.; Yu, P.; Leow, A.D.; Checkoway, S. KOLLECTOR: Detecting Fraudulent Activities on Mobile Devices Using Deep Learning. *IEEE Trans. Mobile Comput.* **2020**. [CrossRef]
23. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A System for Large-Scale Machine Learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
24. Zhang, Y.; Fang, Y.; Xiao, W. Deep keyphrase generation with a convolutional sequence to sequence model. In Proceedings of the 4th International Conference on Systems and Informatics, ICSAI 2017, Hangzhou, China, 11–13 November 2017; pp. 1477–1485. [CrossRef]
25. Myers, D.W.; McGuffee, J. Choosing scrapy. *J. Comput. Sci. Coll.* **2015**, *31*, 83–89.
26. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2012**, *12*, 2825–2830.