Scientometrics
https://doi.org/10.1007/s11192-018-2905-5

DeepPatent: patent classification with convolutional neural networks and word embedding

Shaobo Li[1,2] · Jie Hu[1,3] · Yuxin Cui[3] · Jianjun Hu[2,3]

Received: 15 October 2017

Abstract
Patent classification is an essential task in patent information management and patent knowledge mining. However, this task is still largely done manually due to the unsatisfactory performance of current algorithms. Recently, deep learning methods such as convolutional neural networks (CNN) have led to great progress in image processing, voice recognition, and speech recognition, which has yet to be applied to patent classification. We proposed DeepPatent, a deep learning algorithm for patent classification based on CNN and word vector embedding. We evaluated the algorithm on the standard patent classification benchmark dataset CLEF-IP and compared it with other algorithms in the CLEF-IP competition. Experiments showed that DeepPatent with automatic feature extraction achieved a classification precision of 83.98%, which outperformed all the existing algorithms that used the same information for training. Its performance is better than the state-of-art patent classifier with a precision of 83.50%, whose performance is, however, based on 4000 characters from the description section and a lot of feature engineering while DeepPatent only used the title and abstract information. DeepPatent is further tested on USPTO-2M, a patent classification benchmark data set that we contributed with 2,000,147 records after data cleaning of 2,679,443 USA raw utility patent documents in 637 categories at the subclass level. Our algorithms achieved a precision of 73.88%.

Keywords Patent classification · Text classification · Convolutional neural network · Machine learning · Word embedding

Mathematics Subject Classification 94-02

JEL Classification Y

✉ Jie Hu
jason_houu@gmail.com

✉ Jianjun Hu
Jianjunh@cse.sc.edu

[1] Key Laboratory of Advanced Manufacturing Technology of Ministry of Education, Guizhou University, Guiyang 550025, China

[2] School of Mechanical Engineering, Guizhou University, Guiyang 550025, Guizhou, China

[3] Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29208, USA

Introduction

Patent documents are important intellectual resources, from which valuable technical knowledge such as creative designs and technical knowhow can be obtained for engineering design. It can also help design engineers to understand detailed concepts and underlying component technology (Li et al. 2012). Companies usually use patents as an effective way to protect their intellectual property (IP) and new products' market domination (Trappey et al. 2012). Patents have been a kind of strategic resource for information and knowledge management. More and more technology companies use the huge amount of technical information in patent databases to enhance their research and development (R&D) activities such as new product development (Li et al. 2013), technology transfer (Lemley and Feldman 2016), technology innovation (Lee et al. 2012), technology forecasting (Altuntas et al. 2015) and mergers and technology acquisitions (M&A) analysis (Park et al. 2013, 2017), etc. Since the state-of-the-art technical knowledge can often be found in related patent documents, taking full advantage of patent information allows one to track the progress and frontier of related technologies, which may help to avoid reinventing the wheel and can thus save research cost and shorten research time. Moreover, patent information usually stays current as most patent applications are published 18 months after the first filing, irrespective of their country of origin (Wagner and Wakeman 2016).

With the rapid development in various technology areas, the number of patents has increased dramatically in recent years. In total, innovators filed about 3.1 million patent applications worldwide in 2016, up 8.3% from 2015. For each year since 2003, with the exception of 2009, patent applications have grown every year (WIPO 2018).The enormous increase in the number of patent applications is creating significant challenges for the entire patent system and all patent information users. In reality, patent classification is typically a task which is undertaken almost exclusively by patent experts and patent examiners. The most fundamental task of patent analysis. Besides that, patent classification faces several challenges. Firstly, the IPC taxonomy is a complicated hierarchical structure. Each patent must be assigned one or more sub-group level labels. Secondly, the distribution of patents among categories are highly unbalanced with about 80% of all the documents classified in about 20% of the categories. Furthermore, patent documents are often lengthy and full of technical and legal terminologies, which increases the challenge to efficiently analyze even for domain experts. Automated technologies are thus strongly needed to assist patent analysts in patent processing and analysis.

One of the most frequent tasks of patent processing at a patent office is patent classification, which classifies patent texts into various categories defined in advance by the researchers (Korde and Mahender 2012). The efficiency and accuracy of patent analysis will be greatly improved if we can use artificial intelligence technology to speed up the patent classification process. In the past few years, a number of different algorithms and models have been proposed for patent classification such as the k-Nearest Neighbor (k-NN) (Fall et al. 2003), support vector machine (SVM) (Wu et al. 2010; Fall et al. 2003; D'Hondt et al. 2013), Naïve Bayes (NB) (Fall et al. 2003; D'Hondt et al. 2013), *k*-means clustering (Kim et al. 2008), and artificial neural network (ANN) (Trappey et al. 2006; Guyot et al. 2010), but all with limited success. And especially, most of these algorithms are trained and evaluated with relatively old datasets (at least 10 years ago) while in reality we are faced with millions of new patents. Recently, a few researchers have applied the breakthrough deep learning techniques such as recursive neural networks (RNN) (Luong et al.

2013) and convolutional neural networks (CNNs) (Zhang et al. 2015; Kim 2014) to solve the text classification problem. However, to the best of our knowledge, there is no study that has applied deep neural networks to solve the large-scale patent classification task.

The performance of patent classification algorithms depends on many factors such as the choice of machine learning algorithms, selection of text features, or preparation of training and test data sets. Among all, the selection of relevant features and their encoding is the key factor that affect a model's classification performance (Blum and Langley 1997). Most approaches for text classification use the Bag-of-Word (BoW) (D'Hondt et al. 2013), Term Frequency–Inverse Document Frequency (TF–IDF) (Azam and Yao 2012) or N-grams model to represent the text (D'Hondt et al. 2013). These methods represent each document by the contained words, ignoring the semantics of the words and disregard word order in the original document. Empirical results showed that a fairly simple set of features can be used to get decent classification performance (D'hondt et al. 2012, 2013). However, we can achieve significant performance gain by using carefully constructed features based on thorough understanding of the task. To address the limitation of word frequency features, Word vector (Mikolov 2013b) has been proposed and demonstrated that it can capture meaningful syntactic and semantic regularities and can identify content and subsets of content. It makes it possible to learn high dimensional word vectors practically and can be used to represent a large amount of data precisely. Word embedding features have since been applied to more and more text classification tasks (Kim 2014).

Currently, there is no common automatic classification system used by patent offices and no comprehensive approach to automatically classifying the entire set of patents available in patent office due to scalability issue (Meireles et al. 2016). In this study, we propose DeepPatent, a deep neural network model combined with word embedding to address the patent classification problem with a large corpus. Our algorithm exploits the automated hierarchical feature extraction of convolutional neural networks and powerful modeling power of deep neural networks to achieve the competitive patent classification result. It first effectively transforms word tokens into feature vectors. Then, lexical level vectors are concatenated to form the text level dense matrix. Meanwhile, text level features are learned using a convolutional approach. Finally, the features are fed into a sigmoid function to predict the patent category label out of 637 categories.

The rest of this paper is organized as follows: In "Background and related works" Section, we summarized the core background knowledge of patent classification and reviewed related literature. In "Methods" Section, we proposed our deep neural network model for patent classification, describe the word representation scheme and the details of our convolutional neural network training. In "Experiments" Section, we described how to prepare and select patent text datasets for experiments, and how to use skip-gram to pre-train the entire dataset to get the 200-dimension word vector set. Besides, a series of experiments on the word vector set and the patent text were done. Furthermore, we conducted a series of classification experiments of English patent documents at the subclass level (637 classes) of the International Patent Classification (IPC) and evaluated the results as the CLEF-IP evaluation campaigns. Our results showed that our approach appears more effective than the performance of traditional machine learning algorithms evaluated on the large dataset composed of 2 million patents. In "Conclusions" Section, we drew our conclusions and presented our future study directions.

Background and related works

In this section, we first give a brief overview of current patent classification schemes in typical patent information processing and describe the International Patent Classification hierarchy. Then we summarize the literature related to this area, including text feature extraction and patent classification algorithms.

Patent classification scheme

When a patent application is ready to be published and made open to the public, one or more classification codes must be assigned to the patent document based on its textual content for the purpose of efficient management and retrieval. Several patent authorities maintain their own classification hierarchies, such as the International Patent Classification (IPC) organized by the World Intellectual Property Organization (WIPO), the Cooperative Patent Classification (CPC) organized by the United States Patent and Trademark Office (USPTO) and the European Patent Office (EPO) and the United States Patent Classification (USPC) organized by the USPTO.

Among all the classification schemes, IPC is the most popular patent classification scheme used worldwide in more than 100 countries to classify their national patent applications. Moreover, the IPC is available in more than ten languages, such as Chinese, English, German, Japanese, Korean, Russian, etc. IPC was established in 1971 based the on Patent Cooperation Treaty (PCT) which was concluded in 1970. Specifically, the IPC taxonomy contains 8 sections, 130 classes, 640 subclasses, 7400 main groups and approximately 72,000 sub-groups. In IPC taxonomy, the section part was represented by capital letter from A to H, including (A) “Human Necessities”; (B) “Performing Operations; Transporting”; (C) “Chemistry; Metallurgy”; (D) “Textiles; Paper”; (E) “Fixed Constructions; (F) “Mechanical Engineering; Lighting; Heating; Weapons; Blasting”; (G) “Physics”; (H) “Electricity”. And the second level of IPC taxonomy is a class which

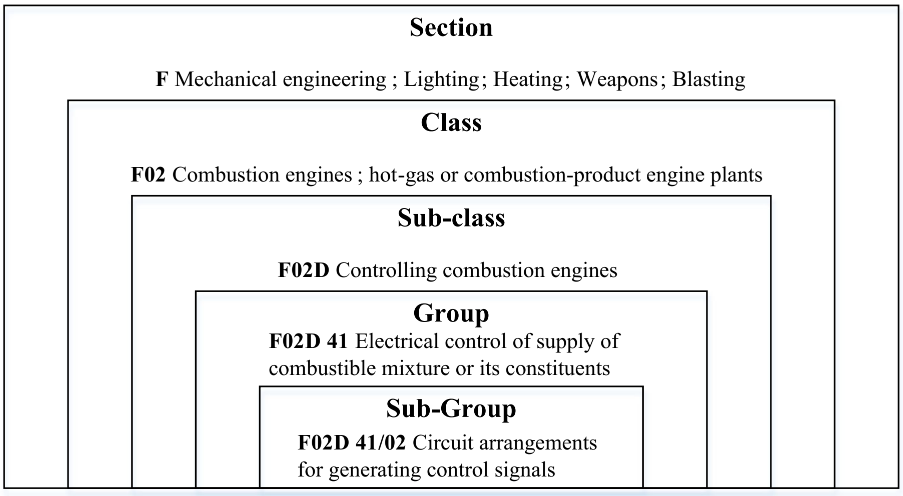

Section
F Mechanical engineering ; Lighting; Heating; Weapons; Blasting
Class
F02 Combustion engines ; hot-gas or combustion-product engine plants
Sub-class
F02D Controlling combustion engines
Group
F02D 41 Electrical control of supply of combustible mixture or its constituents
Sub-Group
F02D 41/02 Circuit arrangements for generating control signals

Fig. 1 An example of IPC

was represented by a numeral. Then the following level is sub-class, group, and subgroup. Figure 1 shows an example of IPC.

The patent coding process currently is done manually in most patent offices around the world. Text classification algorithms are needed to automate this tedious information processing step, which involves the design of representation scheme of patent texts, the selection and design of the classifier algorithms, and the preparation and training of the prediction models.

Each IPC code consists of Arabic numerals and letters of the Latin alphabet. To illustrate the IPC hierarchy, Fig. 1 describes the class label "F02D 41/02" and its ancestors. The "F02D 41/02" class label, which groups circuit arrangements for generating control signals, fall under section F "Mechanical Engineering; Lighting; Heating; Weapons; Blasting", class F02 "Combustion engines; hot-gas or combustion-product engine plants", subclass F02D "Controlling combustion engines", group F02D 41 "Electrical control of supply of combustible mixture or its constituents", sub-group F02D 41/02 "Circuit arrangements for generating control signals".

Text feature extraction

One of the main steps of text classification is text representation or feature extraction, which is also a fundamental problem in information retrieval. Different text representation schemes extract different text features, which have big influence on text classification accuracy. Lewis (1992) proposed phrases as index terms for text classification. It was found that different syntactic phrase indexing schemes led to different text classification performance. The bag-of-words model (D'Hondt et al. 2013) is another standard approach for text encoding, which represents a document by the words' occurrences, ignoring their ordering and grammar in the original document. Empirical results show that (*n*-gram) phrase encoding contains more information than single-word scheme and could lead to better classification performance. However, longer phrases may lead to the explosion of the resulting dataset size. For example, in the Web 1T 5-gram (Brants and Franz 2006) dataset, Google provides the webpage text dataset with the encoding length ranging from unigrams to five-grams. When the length of phrases grows to five, the size of data has exploded to approximately 1 terabyte. Li et al. (2012) proposed a POS and stemming model to count the term more accurately and used a classical TF-IDF algorithm to represent patent documents. They first used Part-Of-Speech (POS) tagging method to identify the part of speech of words in a sentence. After word segmentation and POS tagging, the sentences are then tokenized. In order to count the term frequency accurately, they employed a stemming algorithm to stem all words to their original forms. Beyond that, they also count the term frequency of the corpus and a specific patent to calculate TF–IDF. D'hondt et al. (2013) used four kinds of text representations (unigram, bigram, Stanford, and AEGIR) to represent the patent documents and compared the impact of adding statistical phrases and linguistic phrases. They found that extending statistical phrases and/or linguistic phrases brought significant improvement in classification performance when compared to the unigram baseline. Specifically, the best classification performance appears when all kinds of text representation are combined. These studies indicate that we could represent a text using the BOW model to get decent performance. But due to the loss of information, *n*-grams features cannot encode complex expressions. A major limitation of such traditional BOW representation models is that in these methods, words as treated as discrete atomic symbols, which provide no useful information regarding the relationships that may exist between the individual symbols. This means that the model can leverage very little of what it has learned about a word when it is processing data about a related word. Representing words as unique, discrete ids furthermore leads to data sparsity.

Using vector representations can overcome some of these obstacles. In vector space models (VSMs), words are represented (embedded) in a continuous vector space where semantically similar words are mapped to nearby points. One of the preeminent VSMs text encoding approach is the emerging word vector (Mikolov et al. 2013b), which has been shown to be able to capture meaningful syntactic and semantic regularities and identify text contents and subsets of the content. Word vector models are developed based on the distributional hypothesis (Sahlgren 2008), which states that words that appear in the same contexts share similar semantic meaning. That also means words that occur in similar contexts may have similar embeddings. This encoding approach makes it possible to learn high dimensional word vectors practically and to represent a large amount of data precisely and could be used to improve text classification results. Zhang et al. (2015) proposed a method for sentiment classification based on word vector and SVM. They use word vector to cluster similar features in the selected domain (Chinese language) and put the features into an SVM algorithm, which achieved superior performance in sentiment classification. Matt Taddy (Taddy 2015) applied word vector to 2 million sentences from Yelp reviews and used a classifier to class the sentences into categories. They found that it is a simple, scalable, interpretable, and effective option for text classification. Furthermore, it performs equally well or better than other complex custom-designed algorithms. This paper aims to explore the potential of word vector text representation approach for the large scale patent classification problem.

Patent text classification

Many algorithms have been used to classify patent documents, among which ANN, SVMs, and kNN are the most commonly used algorithms in the field of automated patent classification (Benzineb and Guyot 2011). But, it’s hard to draw a clear conclusion with regard to their performances from previous research because there are no standard data sets used among these studies in patent analysis. Furthermore, the difference in task definitions such as mono-category versus multi-category classification at different levels in the category hierarchy and the choices of data sets also make objective evaluation and comparison of existing methods a challenging task. Here we just survey some leading results based on the reported performance, which should be interpreted in the context of their test datasets. The accuracy scores among different studies are not necessarily comparable.

Li et al. (2012) proposed a two-layered feed-forward ANN and employed Levenberg–Marquardt algorithm to train the ANN for 1948 patent documents from USPC 360/324. The authors used a stemming approach, the Brown Corpus, to handle most irregular words. They achieved accuracy of 73.38% and 77.12% on two categories set. Wu et al. (2010) extracted key words from 234 patent documents and split the documents into two sets (41 correct patent documents and 193 incorrect patent documents). A hybrid genetic algorithm support vector machines (SVMs) was then applied on this data set, which reached an accuracy of 82%. They found that their HGA-SVM was able to increase the prediction accuracy by 1.7% of the SVM patent classification system. However, the significance of these two studies is limited by their small datasets. Chen and Chang (2012) presented a three-phase categorization method which contains SVM, *K*-means, and kNN algorithms. This hierarchical method employed TF–IDF to select discriminative terms to classify

21,104 patent documents (12,042 for training and 9062 for testing) down to the sub-group level and achieved 36.07% accuracy at the sub-group level. They reported that hierarchical methods work better than single level methods even though the overall performance is apparent far from being satisfactory.

Current approaches for patent classification all used the conventional text encoding approach and traditional machine learning algorithms to assign the IPC labels to patent documents. These approaches have several major limitations: (1) they use conventional text features/encoding scheme, which lead to sparsity issue for the training process; (2) the encoding approach cannot capture the complex contents; (3) they are not suitable for massive data processing (Najafabadi et al. 2015) as shown by either their experiments on small datasets or their poor performance on large datasets; (4) there lacks an effective approach to learn features from the training data automatically.

To address above issues in patent classification, we propose to a deep convolutional neural network model combined with distributed word vector encoding to achieve high-performance patent classification. Our work is inspired by an increasing number of studies that employ deep learning models for text classification. Zhang et al. (2015) built a character-level CNNs for text classification trained on several large-scale datasets, which achieved state-of-the-art or competitive results. They treated texts as a kind of raw signal at the character level and applied one-dimensional CNNs to it. Kim (2014) reported a series of experiments with CNNs on pre-train word vectors at the sentence level. He trained a simple CNN with one layer of convolution on top of word vectors obtained from an unsupervised neural language model. The results showed that CNN achieved excellent performance on multiple benchmarks.

Methods

We propose a deep learning algorithm, DeepPatent for large-scale patent classification by combining word vector representation and a well-designed convolutional neural network. Our study has two stages: at the first stage, we extract title and abstract sections from two patent corpuses USPTO-2 M and CLEF-IP 2011 (Piroi et al. 2011), and then use the skip-gram model to transform words in the extracted text into encoding real-value vectors. At the second stage, we first use the pre-trained word embedding lookup table to find the vector of each word and concatenate the vectors into a dense matrix. We then fed the matrices input into a convolutional neural network model with multi-size filters. Our CNN model includes multiple convolutional layers, max-pooling layers, and fully connected layers. Dropout regularization is used also to avoid over-fitting. The details of our DeepPatent model are described as follows:

Overview of the DeepPatent algorithm

Convolutional neural networks (CNN) have been widely using for computer vision and speech recognition in recent years, and have achieved remarkably strong results. In this work, we implemented the W2V-CNN model with an unsupervised neural language model and a simple CNN architecture. The CNN model uses the patent text vectors as the input, which is trained by the continuous skip-gram language model. The CNN model can perform both mono-label and multi-label tasks. In the case of multi-class classification, it is flexible on the number of labels with each label predicted with a probability that a given sample belongs to that labelled class.

Each word from the patent dataset was represented as a 200-dimensional global vector. All the vectors are stored in a lookup table. Thus, we can convert a given patent text description into a matrix by looking up the vector table. Besides, we pad each text to the same length. We append special <*unk*> vector to all other text matrices to make them have the same shape, which is useful because it allows us to efficiently batch our data since each sample in a batch must be of the same size for CNN training. Besides, words not present in the pre-trained vector table initialize with <*unk*> vector.

The detailed hyper-parameters are described as follows. For our DeepPatent, the number of training epoches was set to 50, the training batch size was set as 2048, and the number of input words was set to 100 when only the title and abstract sections are extracted from the entire patent document. The first layers perform convolutions over the 200-dimensional vectors using multiple convolutional filtering kernels. There are three sizes of convolution kernels in our model: 3×200, 4×200 and 5×200. We initialize each kernel with 512 filters. In the next layer, we applied the max-pool operation to the result of the convolutional layer to generate a feature vector. Next, a dropout regularization operation is applied to the feature vector. Finally, the model uses the softmax function to predict the probabilities for all categories. The sigmoid cross entropy was used here to calculate the loss between the true labels and the prediction labels. A sophisticated back-propagation algorithm ADAM (Kingma and Ba 2014) optimizer was used to reduce the loss during the training stage.

Word representation

A word representation method deals with how to represent words by continuous vectors. There is a long history of representation of words as continuous vectors (Hinton 1984; Rumelhart et al. 1986). Bengio et al. (2003) proposed a very popular model to estimate a neural network language model (NNLM), which is a feed-forward neural network with a linear projection layer and a nonlinear hidden layer. A back-propagation training algorithm is used to train a statistical language model that learns to map words into vector representations. In this paper, we use the skip-gram model proposed by Mikolov et al. (2013a) as our distributed word representation approach. This model is based on the distribution hypothesis that words in similar contexts have similar meanings. It can be used for learning high-quality word vectors from huge data sets with billions of words, and with millions of words in the vocabulary. Figure 2 shows the architecture of the skip-gram model.

In this model, $w_1, w_2, w_3, \ldots, w_n$. are the training words, and $c_1, c_2, c_3, \ldots, c_n$. denote their context, which can be generated according to the center word w_i, k is the number of context words. The word-context dependency relationship can be represented by a conditional probability p calculated as:

$$p = p(c_{i+k}|w_i) \tag{1}$$

The goal of the skip-gram model is to maximize the average log probability:

$$\max\left(\frac{1}{n}\sum_{i=1}^{n}\sum_{-k\leq j\leq k, j\neq 0}\log(p(c_{i+j}|w_i))\right) \tag{2}$$

A larger k result in a larger context and thus can lead to higher accuracy (Pennington et al. 2014). It also costs more time to train. When (1) is put into the softmax function, we get:

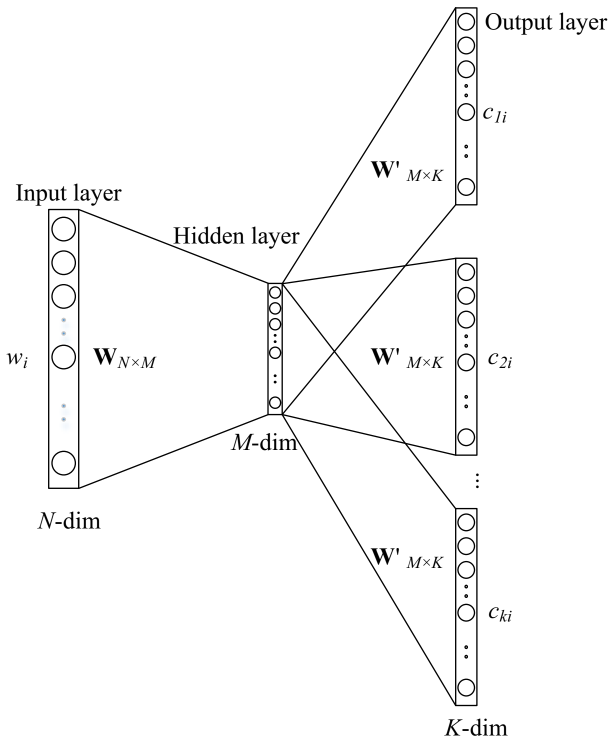


Fig. 2 The architecture of skip-gram model

$$p(c|w;\theta) = \frac{e^{v_c \cdot v_w}}{\sum_{c' \in C} e^{v_{c'} \cdot v_w}} \tag{3}$$

where C is the vocabulary set of the context, W is the set of training words $w_1, w_2, w_3, \ldots, w_n$, D is the set of C and W, v_w and $v_{c'}$ are the "input" and "output" vector representations of w. Put probability (3) into the objective function (2), we have:

$$\max\left(\sum_{(w,c)\in D} \log(p(c|w))\right) = \max\left(\sum_{(w,c)\in D} (\log e^{v_c \cdot v_w} - \log\sum_{c'} e^{v_{c'} \cdot v_w})\right) \tag{4}$$

But calculting objective function (4) is non-trivial because of the computing cost since $\log(p(\mathrm{c}|w;\theta))$ is proportional to W, which is often large (in our case 1,308,700,935 words). To address this issue, negative-sampling (Mikolov et al. 2013a) can be used to reduce the cost of computation. The main idea of negative-sampling is optimizing a different objective function. As mentioned earlier, D is the set of random (w, c) pairs that are all correct. Correspondingly, we can generate D' as the set of random (w, c) pairs that are all incorrect. Then the optimization objective function becomes:

$$\max\left(\sum_{(w,c)\in D}\log\frac{1}{1+e^{-v_c\cdot v_w}}+\sum_{(w,c)\in D'}\log\frac{1}{1+e^{v_c\cdot v_w}}\right) \tag{5}$$

Let $\delta(x)=\frac{1}{1+e^{-x}}$ then the objective function (5) can be expressed as:

$$\max\left(\sum_{(w,c)\in D}\log\delta(v_c\cdot v_w)+\sum_{(w,c)\in D'}\log\delta(-v_c\cdot v_w)\right) \tag{6}$$

Compared to the objective (4), we can easily find that objective (6) will offset cumulative items. Thus the cost for computing will be significantly reduced.

Convolutional neural network model

A convolutional neural network (CNN) (Lecun et al. 1998) is a multilayer neural network composed of a sequence of layers including the convolutional layer, the pooling layer, and the fully-connected layer. The main unique feature of CNNs is the convolution layer composed of a set of filters; each can learn to extract local features from the input using the back-propagation algorithm. By concatenating multiple convolution layers, the CNN model can be trained to learn a hierarchy of features of increasing contexts/scope. This unique capability of CNNs makes it to be one the most successful deep neural network model in deep learning. In DeepPatent, our CNN model architecture is shown in Fig. 3.

We first converted each patent text document into a dense text matrix using the pre-trained word vectors by concatenating the vectors of each word of a patent document.

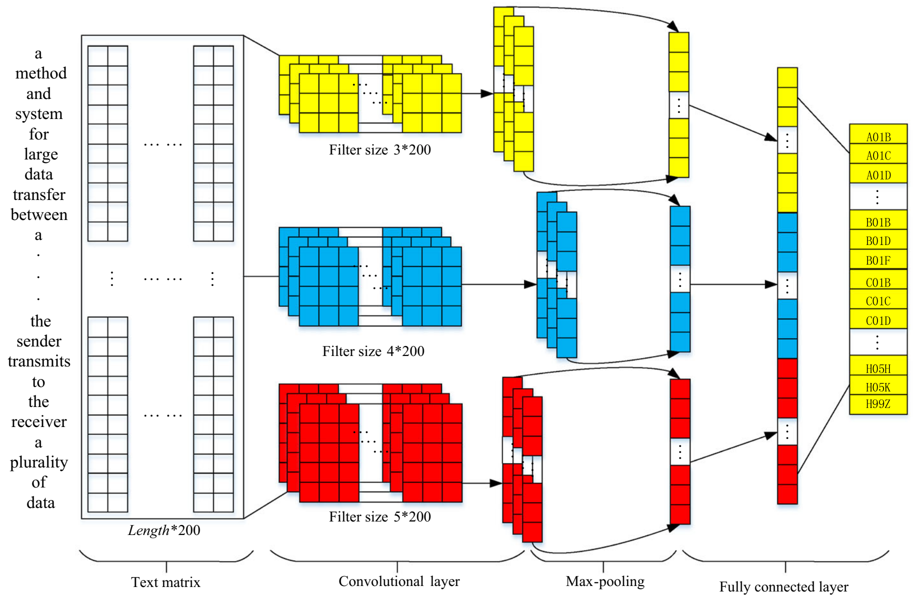


Fig. 3 The architecture of the CNN model in DeepPatent algorithm

Secondly, the algorithm performs convolutions over the dense word vector matrix using filters of different sizes. Next, we apply max-pooling operations to the result of the convolutional layers to create feature vectors, along with dropout regularization. Finally, we use the sigmoid function with cross entropy to calculate the probability outputs. The activation function used here is the Rectified Linear Units (ReLU) function.

In our model, as mentioned earlier, we use $w_1, w_2, w_3, \ldots, w_n$. to denote training words, we use $v_1, v_2, v_3, \ldots, v_n$ to denote the corresponding m-dimensional word vector. A text of length l can then be represented as:

$$V_i = v_1 \oplus v_2 \oplus v_3 \oplus \ldots \oplus v_l \tag{7}$$

Here $\oplus$. is the concatenation operator, V_i. is our model's inmatrix, here the size of V_i is $l \times m$. The next layer is the convolutional layer which exploits the party of spatial local-correlations in the text by enforcing local connectivity pattern between neurons of adjacent layers. It employs a set of filters $K = \{k_1, k_2, k_3, \ldots, k_n\}$ to the text matrix to produce feature map C_i. For each feature c_i^j. in C_i., the feature map is generated by the convolution operation of the text matrix with a linear filter, adding a bias term and applying a non-linear function. Thus, the c_i^j can be represented as follows:

$$c_i^j = f\left(\sum_{p \in m-w+1} V_i^p * k_i^j + b_i\right) \tag{8}$$

Here the size of k^i. is $w * m$ where w. is the width of convocation filters, and $f(x) = \max(0, x)$ is the ReLU activation function and b_i. is the bias term. By applying three kinds of filters (the number of filters of each kind is n) to the text matrix V_i, a feature map can be obtained as:

$$C_i = \left[c_i^1, c_i^2, c_i^3, \ldots, c_i^{3*n}\right] \tag{9}$$

We can then apply a max-pooling operation over time (Taylor et al. 2010) to the feature map c_i^j, and take the max value from c_i^j, $\hat{c}_i^j = \max\left(c_i^j\right)$ as the are corresponding to each specific filter. The idea of the pooling layer is to capture the most important features by picking the highest value to represent each feature. Finally, after the pooling layer, a fully connected layer takes all features in the previous layer and generates the probabilities distribution over each label.

Evaluation criteria for patent classification

In this study, the patent classification problem is a multi-label classification task. For each experiment, we use the followings evaluation metrics as used in CLEP-IP competition (Piroi et al. 2011) to evaluate various methods. Firstly, we predict 1, 4 and 50 classification labels for each patent document. Then we calculate $\text{Precison}_{\text{score}}$, $\text{Recall}_{\text{score}}$, and $F1_{\text{score}}$ for each prediction. The precision score is the number of correct predictions divided by the number of all returned predictions.

$$\text{Precison}_{\text{score}} = \frac{\text{correct predictions}}{\text{all predictions}} \tag{10}$$

The recall score is the number of correct predictions divided by the number of all relevant patent documents.

$$\text{Recall}_{\text{score}} = \frac{\text{correct predictions}}{\text{all relevant patent documents}} \tag{11}$$

$$F1_{\text{score}} = 2 \times \frac{\text{Precison}_{\text{score}} \times \text{Recall}_{\text{score}}}{\text{Precison}_{\text{score}} + \text{Recall}_{\text{score}}} \tag{12}$$

The $\text{Precison}_{\text{score}}$, $\text{Recall}_{\text{score}}$, and $F1_{\text{score}}$ are denoted as Precision, Recall, and F1 respectively. We use # to denote the number of topmost labels returned by the models then we can denote the measures as Precision@#, Recall@#, and F1@# respectively.

Experiments

In this section, we introduce our benchmark datasets and experimental results. First, we describe the preparation of USPTO-2M, a large-scale benchmark pant classification dataset that we contribute to the community, which is used to evaluate the real-world performance of patent classifiers. Then, the skip-gram scheme for pre-training the entire data to get 200-dimension word vectors are evaluated. Finally, a series of experiments are conducted on the USPTO-2M and CLEF-IP datasets to evaluate and compare the performance of our DeepPatent algorithm with that of other existing algorithms.

Preparation of benchmark datasets

To evaluate the real-world performance of patent classification algorithms, two large-scale benchmark datasets are included in this study: USPTO-2M and CLEF-IP. The USPTO-2M dataset is a large-scale dataset that we prepared for benchmark studies of patent classification algorithms, of which the raw patent data are obtained from the online website of the United States Patent and Trademark Office (USPTO). The bulk data contains the full texts of all patents since 1976. There are over 9.4 million records in the dataset. We collected 2,679,443 utility patent documents in 637 categories at the subclass level in last 10 years, namely USPTO-2M. Figure 4 shows the statistic of the number of published patents from

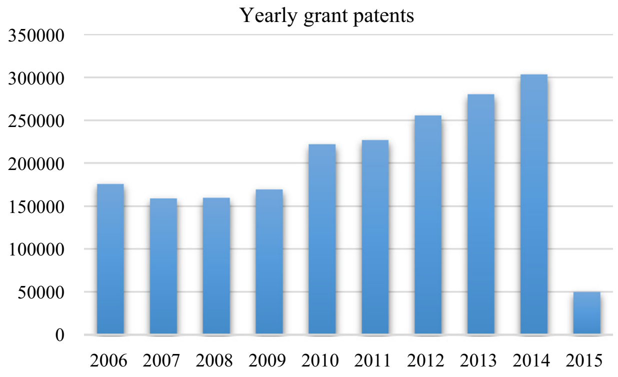
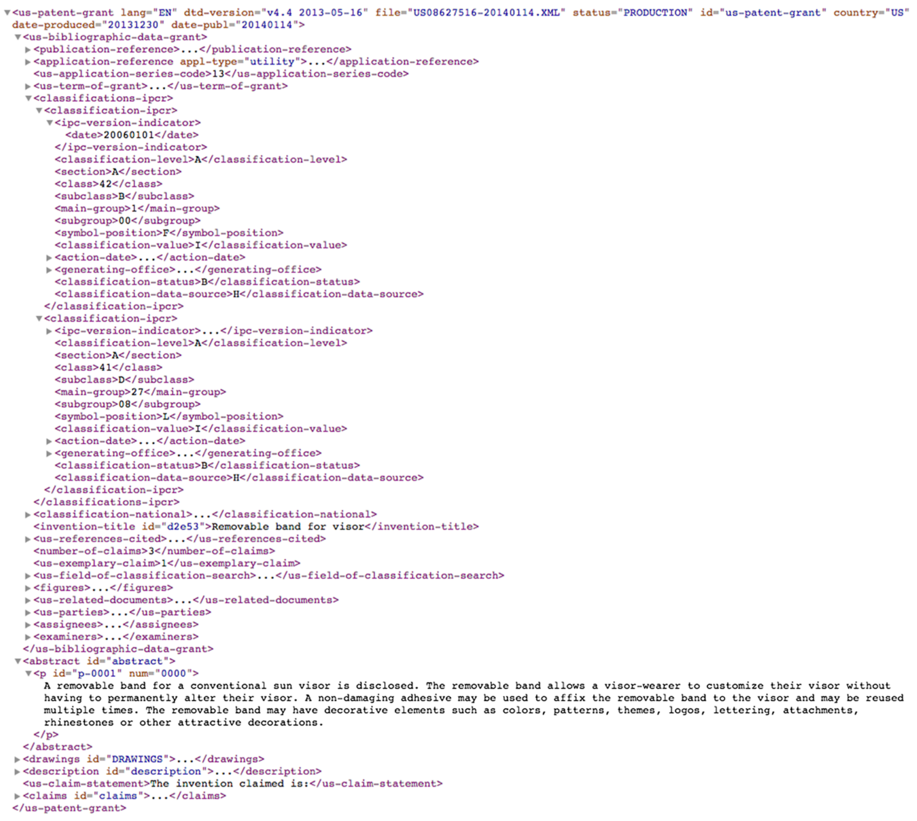


Fig. 4 Yearly distribution of the patents in the USPTO-2M dataset

2006 to 2015. Similar to the CLEF-IP classification task, we extract all the subclass labels from the raw data as our training and test labels.

In total, about 2.7 million patent applications are filed worldwide in 2014, up 4.5% from 2013. For each year since 2003 except for 2009, the number of patent applications has been increasing every year.

A patent document usually includes bibliographic information, the title, the document number, the issued date, the patent type, classification information, a list of inventors, a list of applicant companies or individuals, abstract, claims section, and a full-text description. Figure 5 shows the detail of a sample patent. More specifically, the title of a patent indicates the name of the patent; the abstract part gives a brief technical description of the innovation; the patent type explains patent's type, and the classification part presents one or multiple class labels. The claim section's main function is to protect the inventors' right without any detailed technical information. The description section describes the process, the machine, manufacture, composition of matter, or improvement invented, a brief summary and the background of the invention, the detailed description, and a brief description of its application. The documents also contain meta-information on assignee, date of application, inventor, etc. We did not collect any meta-data in our dataset since we focus on text representation.



```
▼<us-patent-grant lang="EN" dtd-version="v4.4 2013-05-16" file="US08627516-20140114.XML" status="PRODUCTION" id="us-patent-grant" country="US"
 date-produced="20131230" date-publ="20140114">
  ▼<us-bibliographic-data-grant>
    ▶<publication-reference>...</publication-reference>
    ▶<application-reference appl-type="utility">...</application-reference>
     <us-application-series-code>13</us-application-series-code>
    ▶<us-term-of-grant>...</us-term-of-grant>
    ▼<classifications-ipcr>
      ▼<classification-ipcr>
        ▼<ipc-version-indicator>
            <date>20060101</date>
         </ipc-version-indicator>
         <classification-level>A</classification-level>
         <section>A</section>
         <class>42</class>
         <subclass>B</subclass>
         <main-group>1</main-group>
         <subgroup>00</subgroup>
         <symbol-position>F</symbol-position>
         <classification-value>I</classification-value>
        ▶<action-date>...</action-date>
        ▶<generating-office>...</generating-office>
         <classification-status>B</classification-status>
         <classification-data-source>H</classification-data-source>
       </classification-ipcr>
      ▼<classification-ipcr>
        ▶<ipc-version-indicator>...</ipc-version-indicator>
         <classification-level>A</classification-level>
         <section>A</section>
         <class>41</class>
         <subclass>D</subclass>
         <main-group>27</main-group>
         <subgroup>08</subgroup>
         <symbol-position>L</symbol-position>
         <classification-value>I</classification-value>
        ▶<action-date>...</action-date>
        ▶<generating-office>...</generating-office>
         <classification-status>B</classification-status>
         <classification-data-source>H</classification-data-source>
       </classification-ipcr>
     </classifications-ipcr>
    ▶<classification-national>...</classification-national>
     <invention-title id="d2e53">Removable band for visor</invention-title>
    ▶<us-references-cited>...</us-references-cited>
     <number-of-claims>3</number-of-claims>
     <us-exemplary-claim>1</us-exemplary-claim>
    ▶<us-field-of-classification-search>...</us-field-of-classification-search>
    ▶<figures>...</figures>
    ▶<us-related-documents>...</us-related-documents>
    ▶<us-parties>...</us-parties>
    ▶<assignees>...</assignees>
    ▶<examiners>...</examiners>
   </us-bibliographic-data-grant>
  ▼<abstract id="abstract">
    ▼<p id="p-0001" num="0000">
       A removable band for a conventional sun visor is disclosed. The removable band allows a visor-wearer to customize their visor without
       having to permanently alter their visor. A non-damaging adhesive may be used to affix the removable band to the visor and may be reused
       multiple times. The removable band may have decorative elements such as colors, patterns, themes, logos, lettering, attachments,
       rhinestones or other attractive decorations.
     </p>
   </abstract>
  ▶<drawings id="DRAWINGS">...</drawings>
  ▶<description id="description">...</description>
   <us-claim-statement>The invention claimed is:</us-claim-statement>
  ▶<claims id="claims">...</claims>
 </us-patent-grant>
```

Fig. 5 Sample patent record in the bulk data

The title, abstract, and the beginning of the description are generally considered as the most informative sections of a patent (Benzineb and Guyot 2011). D'hondt and Verberne (2010) showed that the title and abstract sections are more informative than the full-text representation of the patent document for patent retrieval. It improved the classification results on the CLEF-IP 2010 dataset. Wu et al. (2010) applied various types SVM kernel functions to various source datasets created by using the title, abstract, claim, and the description part of the English patent documents. They found that title and abstract sections provide high-quality information for patent classification.

From the raw patent dataset, we selected all the patents that contain the title, abstract, and at least one IPC label in the <*classifications-ipcr*> field. In total, there are eight different sections at the section level, 130 different classes at the main class level, 637 different subclasses at the subclass level, 72,000 different sub-group classes at the sub-groups level. On average, in our dataset, each patent has 1.34 classification labels at the subclass level. Table 1 shows essential information of USPTO-2M corpus.

After data cleaning, the dataset contains 2,000,147 patents with the title and abstract sections. To approach the realistic situation, we split the patent document based on the time axis. We used earlier patents for training and used the later patents for testing. Specifically, we used the 2006–2014 yearly data as our training data and used the 2015 patents as the test data. The longest text in a patent document has 514 words, and there are only eight words in the shortest document. However, we only chose those patents with texts of more the ten words. On average there are 118 words in a document.

To compare DeepPatent with other patent classification algorithms, we also collect the CLEF-IP dataset and conduct experiments on the dataset as the baseline of patent classification. This dataset contains 2.6 million patent documents (about 1.3 million patents, each patent can consist of one or more patent documents) from the European Patent Office (EPO) and 400,000 documents from the World Intellectual Property Organization (WIPO), which published between 1985 and 2011. These 3 million documents include three languages, English, German, and French. For CLEF-IP dataset, we extract and clean the documents which are written in English. Table 2 lists the essential information of CLEF-IP 2011 corpus.

The effectiveness of pre-trained word embedding

In this study, the continuous skip-gram (Mikolov et al. 2013b) model is used to pre-train our dataset, which can capture a large number of precise syntactic and semantic word relationships and learn high-quality distributed vector representations for real-world patent classification processing.

Our word embedding model is pre-trained as follows: firstly, we parse the XML files into text files and then extract the title and abstract sections. Next, we delete all punctuations and lowercase words in the whole text. We also delete the words whose occurrence frequency are below five. Thirdly, we set the parameter window size in the skip-gram

Table 1 The essential information of USPTO-2M corpus

	Training data	Test data
Number of patents	1,950,247	49,900
Number of different IPC-R subclasses	637	606
Average number IPC labels per patent	1.32	1.93

Table 2 The essential information of CLEF-IP 2011 corpus

	EPO	WIPO	Test data
Number of patents	580,546	161,551	1350
Number of different IPC-R subclasses	622	613	341
Average number of IPC per patent	1.96	1.99	2.09

model as five, meaning five words ahead and five words behind, and then split the whole text into dependent pairs (word and context) for the word vector training stage. Then, we use the dependent pairs to generate 200-dimensional word vectors. The word embedding model is fixed after training.

To evaluate the pre-trained word embedding approach, we conduct several experiments on the word analogy and a variety of word similarity tasks. We setup a collection of questions, such as "a is to b as c is to _?". The questions set contains two categories, a syntactic category, and a semantic category. The syntactic questions are mainly analogy about the plural of nouns or verb tenses, such as "image is to images as unit is to _?". The semantic questions are typically analogy about synonyms and antonyms, such as "inside is to outside as liquid is to _?". To find the correct answer to the question, the model should find the unique term. We give the answer to the question "*a* is to *b* as *c* is to _?" by finding the word *d* whose representation is closest to according the cosine similarity. We employ the t-SNE (Laurens 2014) to project the learned vectors down to 2 dimensions.

The subplots in Fig. 6 are created by projecting the 200-dimensional vectors of syntactic questions and semantic questions to two-dimension space. Subplot A and B indicate that the skip-gram model has the ability to capture the antonym of each word. The subplot C shows that the model can capture the semantic changes of singular and plural. The subplot D indicates that it can capture nominalization.

Figure 6 illustrates the capability of the skip-gram model to automatically organize concepts and learn implicitly the relationships among these concepts. The subplots showed that the skip-gram model can capture word antonyms and semantic changes between singular and plural as well as capture nominalization. It should be noted that during the training process, we did not provide any supervised information about what a specific word means. It shows that our trained word embedding can find semantically similar or related words.

Performance of DeepPatent on patent classification

We conducted a series of patent classification experiments on the English patent documents at the subclass level (637 classes) of the International Patent Classification (IPC) from the USPTO-2M and the CLEF-IP datasets. The evaluation measures used by the CLEF-IP patent classification campaign are applied here for evaluation and comparison of DeepPatent with existing algorithms. We also studied how different parts of a patent description such as the title, the abstract, the full text and their combination affect the classification performance. This allows us to choose the right sections from the patent description to better represent a patent document for classification. Furthermore, we use a varying number of words from the title and abstract sections as the model input to determine the number of words that can sufficiently represent the whole document. Finally,

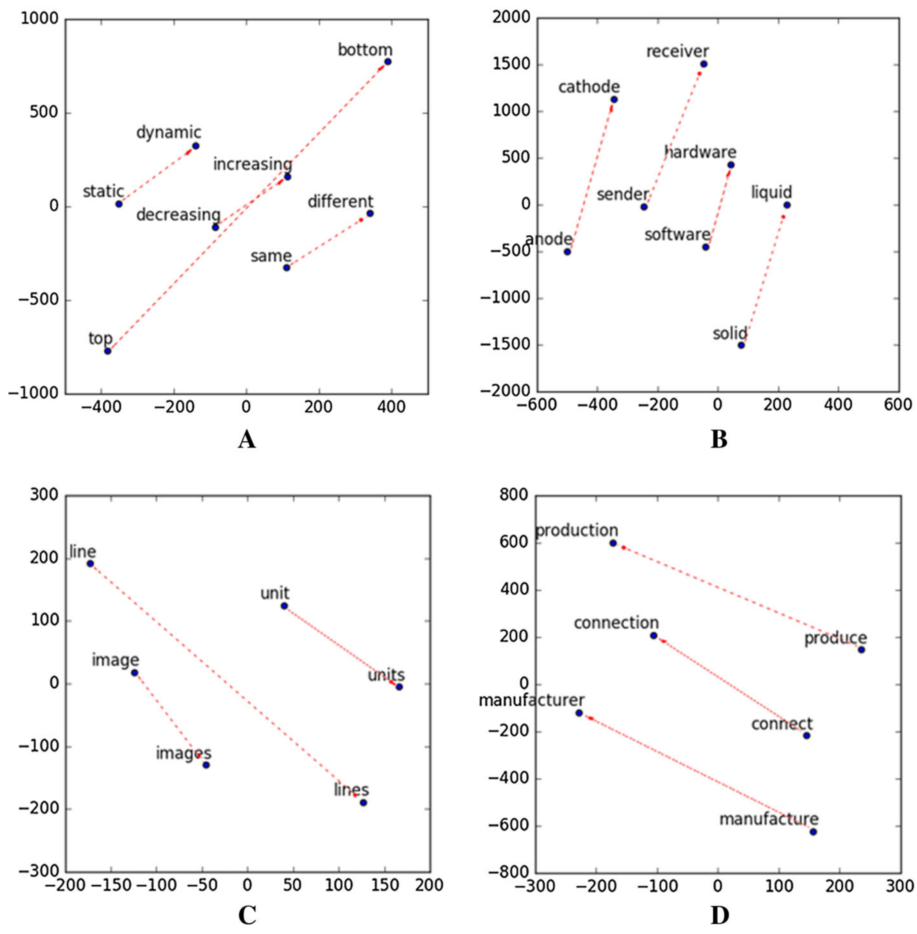


Fig. 6 Successful learning of semantic relationships among words by the skip-gram model

we compared the performance of DeepPatent with other classification algorithms over the standard benchmark dataset CLEF-IP.

Firstly, we conducted a series of experiments on the USPTO-2M dataset to evaluate how different sections of the patent description can lead to best classification performance. We compared the prediction performances of DeepPatent and other four algorithms using different individual or combinations of the patent description sections including titles, abstracts, and their combination. The experimental results are shown in Fig. 7. As shown in Fig. 7a, when only the title section is used as input, the classical BP networks achieved better performance than the others in terms of top5 precision, recall, and F1 measures. Our DeepPatent algorithm instead only achieved the best performance for top1 precision. This shows that using title information only is insufficient for DeepPatent to exploit the power of feature extraction by the convolutional neural networks. Figure 7b shows the performance of all 5 algorithms when only the abstract text is used as input. First, we found that our DeepPatent algorithm outperformed all other 4 algorithms in terms of all four evaluation criteria including top1 precision, top5 precision, top5 recall, and top5 F1. The BP

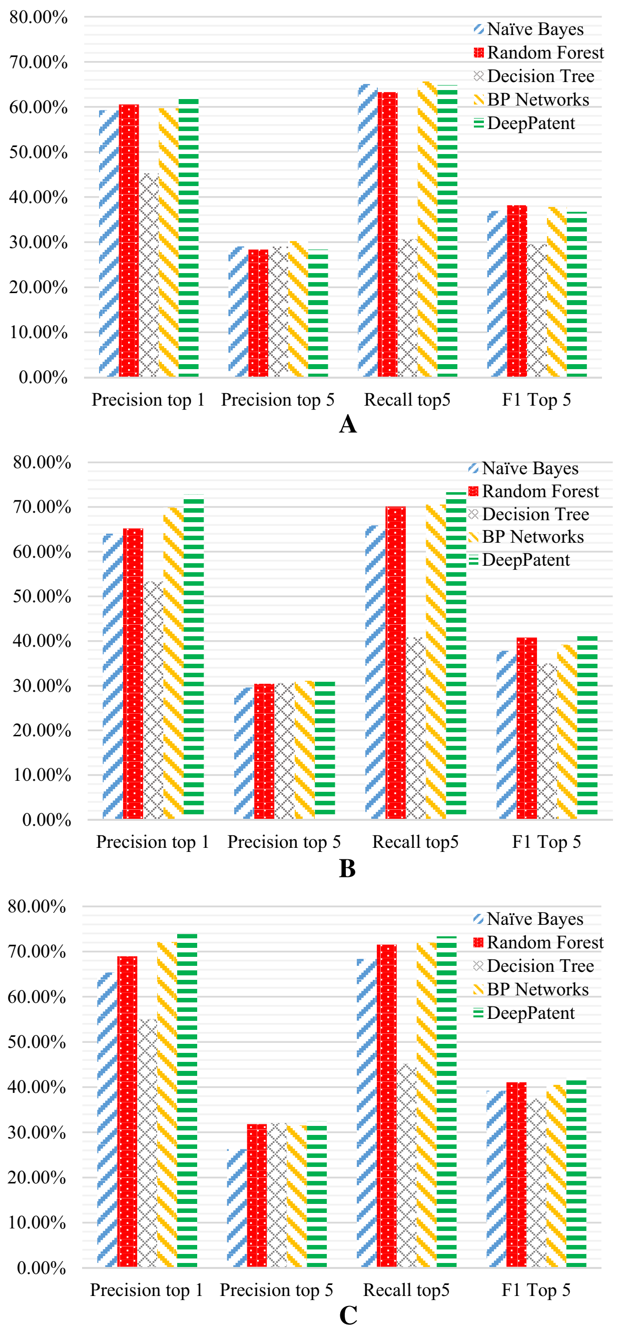


Fig. 7 Performance comparison of different algorithms on datasets with different text fields (title and abstract) evaluated on the USPTO-2M dataset. **a** Performance comparison on datasets with title section. **b** Performance comparison on datasets with abstract section. **c** Performance comparison on datasets with title and abstract section

networks achieved the second best performance overall. In addition, it is found that the prediction performance has been improved compared to those in Fig. 7a, which only uses the title as the sample information. Figure 7c shows the experimental results achieved by all five algorithms using both title and abstract sections from the USPTO-2 M dataset. Again, first, the performance of all algorithms have improved compared to Fig. 7a, b. Also, the DeepPatent algorithm shows the best results in terms of all 4 criteria.

Our experimental results showed that the abstract section has the key information for patent classification and reasonable results can be achieved using only the abstract section, which can be further improved by combining abstract with title section. We also found that the DeepPatent model cannot show its advantage when only the title section is used as the input even though it has achieved competitive results compared with other benchmark algorithms. However, when both the title and abstract are used as the input content, DeepPatent showed its superior performance in terms of all criteria.

Another important parameter in our CNN model is the number of words used to represent a patent, which determines the input dimension of our input data. We tested the performance of our CNN models using a different number of words from the title and abstract sections as the input to determine how many words can well represent the patent information for effective patent classification. We trained and evaluated the DeepPatent models using the top words in the patent text as input samples ranging from 10 words to 514 words. We used the DeepPatent models to predict top 1, 5 and 10 classification labels for each patent document respectively. The model performances in terms of precision with a different number of words are reported in Fig. 8. As we can see from Fig. 8, the number of input words a huge effect on the classification performance of our CNN models. The precision score @top1 continuously increases from 61.43 to 73.88% as the number of words goes up towards 100. Similar trends have been observed also for the precision scores of top 5 and top 10 results. We found that the significant increase of the precision scores happens when the word number increases from 10 to 70. When the number of input words reaches 100, the performance tends to stagnate.

Similarly, we checked how the recall scores of DeepPatent changes with different numbers of input words used as the model input. The recall scores @ top 5 and 10 are shown in Fig. 9. The best recall score @ top 5 is 73.66% using 80 words while the best recall score @ top 10 is 82.74% using 70 words. Similar to the effect of word number on the prediction precision, it has the similar effect on the recall scores of the CNN models as shown in Fig. 9: the top5 recall scores first increase as the number of words goes up from 10 to 70 words, which stagnates and then decreases as the word number increases over 100 words. Top 10 recall scores showed a similar trend. These results illustrate that appropriately setting the number of input words (which is 100 for our final models) for patents helps to ensure good performance of our CNN models.

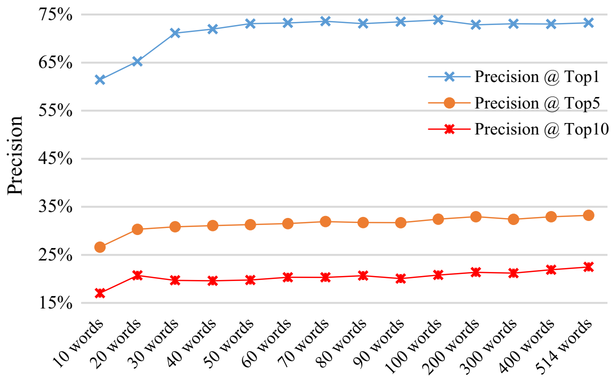


Fig. 8 The precision scores achieved by DeepPatent versus the number of words using USPTO-2M dataset

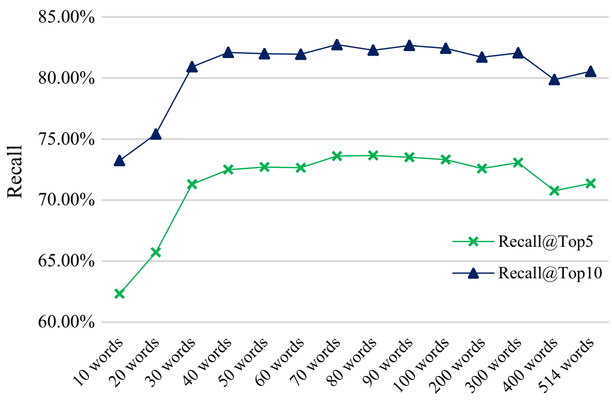


Fig. 9 The recall scores achieved by DeepPatent with different numbers of words over the USPTO-2M dataset

Combining the precision and recall scores, we calculated the F1 scores for the CNN models with different numbers of word number to comprehensively understand the effects of word number on classification performance. As shown in Fig. 10, we can find that the best performance was achieved when the number of input words is around 70 to 100. The F1 scores tend to slowly decrease when the number of input words is more than 100. This general trend can be explained as follows: when the number of words used to represent the patent is too small to capture the patent information, the prediction performance is low. At this stage, increasing the number of words to represent the patent will increase the classification performance. When the number of words reaches more than 100, the benefit of including more information is then balanced by the increased input dimension. Potentially, for patent classification, the most important words might be the first words, so more words do not give a gain since most of the relevant information is already captured. Besides, when the number of samples is the same, CNN models with a dramatic increase of model complexity may lead to more severe overfitting or sparsity issues, leading to lower prediction performance on the test dataset.

Based on above experiment results, we use the title and abstract sections as the information source and select top 100 words as the input information for our DeepPatent model.

Finally, in order to evaluate how our DeepPatent model performs in patent classification, we performed a series of patent classification experiments on the CLEF-IP dataset to compare DeepPatent against 9 other patent classification schemes. We found that it is challenging to conduct completely fair completion as neither the softwares or the web

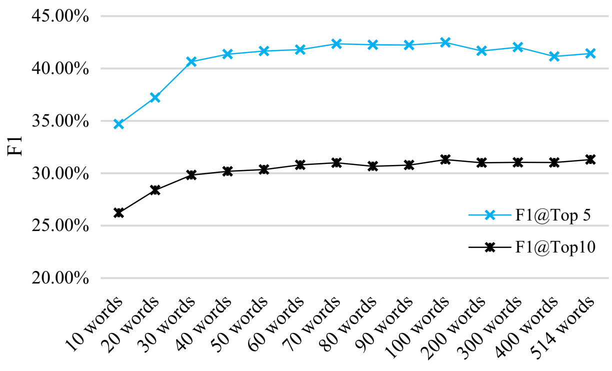


Fig. 10 The F1 scores achieved by DeepPatent with a various number of words using USPTO-2M dataset

services of other patent classification algorithms are available and different patent information (such as the title, abstract, description, or full content) is used as the classifier input. It is thus necessary to consider all these factors in comparing and interpreting the performance differences. We used 580,546 EPO patents and 161,551 WIPO patents from 1985 to 2011 out of the CLEF-IP dataset as the training samples and 1350 patents from 2012 to 2015 as test samples to train these classifiers and evaluated their classification performances in terms of top1 precision, top4 precision/recall/F1 as well as top 50 recall score.

Table 3 shows the performance comparisons of DeepPatent with other algorithms. It is found that our DeepPatent algorithm significantly outperforms most of the other methods using only the title and abstract information of the patents, achieving a Top1 precision of 83.98%, better than 69.95% and 71.85% of SVM^{Light} (which use the abstract information) and 74.43% by LCS (which uses abstract and description). DeepPatent also performs better than the SVM-based algorithm reported in (Derieux et al. 2010) that uses two types of human-designed features including similarity and semantic and statistics along with full content information as input. The performance of DeepPatent with 83.98% top1 precision is better than the state-of-the-art result of 82.1% top1 precision achieved by the SVM with full content information of the patent as sample input and complicated human-designed features. Out of the 11 methods compared in Table 3, DeepPatent, SVM$^{\text{Lights}}$, and LCS all use the abstract with or without titles, which makes their performance to be more comparable. From the first 3 rows of Table 3, it can be found that DeepPatent apparently achieved the best performance with Top 4 F1 score of 55.09% compared to 47.42% and 48.56% of SVM$^{\text{Light}}$. The top 50 recall scores of DeepPatent are also much higher than of those of SVM$^{\text{Light}}$ (97.35% compared to 87.61% and 89.56%).

To gain more insight over the failure cases of our algorithm, 214 wrongly classified samples are identified in the scenario of predicting 1 label for each document. Misclassifications of these samples can be attributed to three types of errors: non-relevant error, section-correct error, and main-class-correct error. The non-relevant error means that a prediction label does not match either the label at the section level or at the main class level. The section-correct error means that a prediction label only matches the label at the section level. While the main-class-correct error means that a prediction label matches the label at the main class level. There are 90 (42%), 64 (30%) and 60 (28%) mis-classified samples that belong to these three error types, respectively. The non-relevant error maybe partially caused by the dynamical nature of the IPC due to the creation and deprecation (or merge) of categories over time in the IPC system. This definitely affects the performance of our model, since the definitions of some categories could be modified in different versions of the IPC, and the documents in the training data may contain various IPC version labels. As for the section-correct and main-class-correct errors, our DeepPatent could correctly predict the superset of the subclass level labels. For example, for test samples No. 68, 81, 354, 487, 494, and 531, the true labels are C07C, C08F, H04L, B41J, F04B and H04Q, while the prediction labels are C07G, C08L, H04N, B41N, F04C and H04J. Actually, the true and prediction labels are siblings, which means that the true labels and prediction labels have very similar meaning. At the same time, the main text in those patent documents may be written ambiguously. Those are the chief reasons contributing to the section-correct-errors and main-class-correct-errors.

Table 3 also shows that for the same classifier, the prediction performance varies when different features are used. For example, the SVM-based model using combined handcrafted similarity and semantic features achieved the best performance (82.1% top1 precision) over the EPO corpus when the full content is used. Instead, our DeepPatent algorithm achieved competitive performance despite its use of less information with only

Table 3 Classification performance of DeepPatent and other methods over patent datasets

Methods	Corpus	Sections	Text feature extraction	Precision top 1 (%)	Precision top 4 (%)	Recall top 4 (%)	F1 top 4 (%)	Recall top 50 (%)
DeepPatent	EPO + WIPO	Title + abstract	Word vector	83.98	45.79	75.46	55.09	97.35
SVM^{light} (Verberne et al. 2010)	EPO	Abstract	Words-only	69.95	37.46	66.60	47.42	87.61
SVM^{light} (Verberne et al. 2010)	EPO	Abstract	Words + triples	71.85	38.36	66.16	48.56	89.56
SVM-based (Derieux et al. 2010)	EPO	Full content	Similarity	77.5	–	–	–	–
SVM-based (Derieux et al. 2010)	EPO	Full content	Semantic and statistic	75.15	–	–	–	–
SVM-based (Derieux et al. 22010)	EPO	Full content	Semantic and statistic	82.1	–	–	–	–
LCS (Verberne and D'hondt 2011)	EPO + WIPO	Abstract + description	Words-only	74.43	–	–	–	–
LCS Winnow (Koster et al. 2011)	EPO	Abstract	Words + triples	78.25	–	–	–	–
LCS SVM (Koster et al. 2011)	EPO	Abstract	Words + triples	82.29	–	–	–	–
Balanced Winnow (D'hondt et al. 2013)	EPO	Abstract	Unigrams + Bigrams + Stanford + AEGIR triples	79.51	–	–	–	–
MyClass (Guyot et al. 2010) (the best performance in the competition)	EPO	Description (4000 characters)	5-gram	83.55	–	–	–	–
LSTM (Grawe et al. 2017)	USPTO	Description	Word vector	63	–	–	–	–

100 words from title and abstract as input. It has additional benefits of avoiding the tedious manual feature engineering process. Our CNN model thus provides a promising way to do patent classification without handcrafted features.

Conclusions

Patent pre-classification and classification is typically a task currently undertaken almost exclusively by patent experts and patent examiners. In this paper, we proposed an effective patent classification algorithm DeepPatent, based on word vector and convolutional neural networks to solve the large-scale, unbalanced, multilabel, multiclass patent classification problem.

We collected and published the USPTO-2M dataset, the largest patent classification data set so far for benchmark studies of patent classification algorithms. It contains 2,000,147 records after data cleaning of 2,679,443 raw utility patent documents in 637 categories at the subclass level from US patent office. We have identified the best strategy to select patent sections and number of words for patent representation to achieve better classification performance. Evaluated on the USPTO-2M dataset, we found that using the top 100 words from the title and abstract sections achieved the best result. With this setting, our DeepPatent algorithm achieved a classification precision of 73.88% over the USPTO-2M dataset. We further evaluated the performance of DeepPatent along with 6 other patent classification algorithms over the CLEF-IP benchmark dataset. Our DeepPatent algorithm with automatic feature extraction has achieved a precision of 81.11% for top 1 label predictions, which is better than all traditional machine learning algorithms except for one SVM method with precision score 82.1% that was derived with a large amount of feature engineering effort and using the full content as input.

To the best of our knowledge, our work is the first study that developed and applied a deep learning model and algorithm to large scale real-world patent classification. Compared with traditional machine learning algorithms, the DeepPatent model has several advantages in large-scale patent classification including free of hand crafted features, straightforward models, and easy to implement without tedious feature engineering. Our CNN models are flexible to accommodate variable-size word embedding and can easily be applied to classify different levels of the IPC hierarchy (e.g., group and sub-group). This method can also be easily applied to other text processing tasks.

There are several ways to further improve the DeepPatent algorithm. Firstly, the corpus for training word embedding is not large enough currently. We can collect more patent documents to generate a bigger patent corpus for training patent domain vectors for word embedding. Secondly, due to the fact that patents have a large number of category labels, it is hard to predict the exact number of labels for each patent. In the future, we can improve the capability of the model to determine the threshold to predict the categories and the number of labels.

Acknowledgements This work is supported by the National Natural Science Foundation of China under Grant Nos. 51475097, 91746116 and 51741101, the China Scholarship Council, and Science and Technology Foundation of Guizhou Province under Grant Nos. JZ[2014], Talents[2015]4011, and [2016]5013, and Collaborative Innovation[2015]02. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research (Grant No. 61640209).

References

Altuntas, S., Dereli, T., & Kusiak, A. (2015). Forecasting technology success based on patent data. *Technological Forecasting and Social Change, 96,* 202–214.

Azam, N., & Yao, J. (2012). Comparison of term frequency and document frequency based feature selection metrics in text categorization. *Expert Systems with Applications, 39*(5), 4760–4768.

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research, 3*(6), 1137–1155.

Benzineb, K., & Guyot, J. (2011). Automated patent classification. *Current Challenges in Patent Information Retrieval, 29,* 239–261.

Blum, A. L., & Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence, 97*(1), 245–271.

Brants, T., & Franz, A. (2006). Web 1T 5-gram Version 1.

Chen, Y.-L., & Chang, Y.-C. (2012). A three-phase method for patent classification. *Information Processing and Management, 48*(6), 1017–1030.

D'hondt, E., Verberne, S., Weber, N., Koster, K., & Boves, L. (2012). Using skipgrams and pos-based feature selection for patent classification. *Computational Linguistics in the Netherlands Journal, 2,* 52–70.

Derieux, F., Bobeica, M., Pois, D., & Raysz, J. P. (2010). Combining semantics and statistics for patent classification. In *CLEF 2010 LABs and workshops, notebook papers, 22–23 September 2010, Padua, Italy*.

D'hondt, E., & Verberne, S. (2010). CLEF-IP 2010: Prior art retrieval using the different sections in patent documents. In *CLEF (notebook papers/LABs/workshops)*.

D'hondt, E., Verberne, S., Koster, C., & Boves, L. (2013). Text representations for patent classification. *Computational Linguistics, 39*(3), 755–775.

Fall, C. J., Törcsvári, A., Benzineb, K., & Karetka, G. (2003). Automated categorization in the international patent classification. ACM SIGIR forum.

Grawe, M. F., Martins, C. A., & Bonfante, A. G. (2017). Automated patent classification using word embedding. In *16th IEEE international conference on machine learning and applications (ICMLA)*.

Guyot, J., Benzineb, K., Falquet, G., & Shift, S. (2010). myClass: A mature tool for patent classification. In *CLEF (notebook papers/LABs/workshops)*.

Hinton, G. E., McClelland, J. L., & Rumelhart, D. E. (1984). *Distributed representations* [M]. Pittsburgh, PA: Carnegie-Mellon University.

Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

Kim, Y. G., Suh, J. H., & Park, S. C. (2008). Visualization of patent analysis for emerging technology. *Expert Systems with Applications, 34*(3), 1804–1812.

Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Korde, V., & Mahender, C. N. (2012). Text classification and classifiers: A survey. *International Journal of Artificial Intelligence & Applications, 3*(2), 85.

Koster, C. H., Beney, J. G., Verberne, S., & Vogel, M. (2011). Phrase-based document categorization. In K. Mayer & A. J. Trippe (Eds.), *Current challenges in patent information retrieval* (pp. 263–286). Berlin: Springer.

Laurens, V. D. M. (2014). Accelerating t-SNE using tree-based algorithms. *Journal of Machine Learning Research, 15*(1), 3221–3245.

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278–2324.

Lee, S., Lee, H. J., & Yoon, B. (2012). Modeling and analyzing technology innovation in the energy sector: Patent-based HMM approach. *Computers & Industrial Engineering, 63*(3), 564–577.

Lemley, M. A., & Feldman, R. (2016). Patent licensing, technology transfer, and innovation. *The American Economic Review, 106*(5), 188–192.

Lewis, D. D. (1992). An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th annual international ACM SIGIR conference on research and development in information retrieval*.

Li, Q., Maggitti, P. G., Smith, K. G., Tesluk, P. E., & Katila, R. (2013). Top management attention to innovation: The role of search selection and intensity in new product introductions. *Academy of Management Journal, 56*(3), 893–916.

Li, Z., Tate, D., Lane, C., & Adams, C. (2012). A framework for automatic TRIZ level of invention estimation of patents using natural language processing, knowledge-transfer and patent citation metrics. *Computer-Aided Design, 44*(10), 987–1010.

Luong, T., Socher, R., & Manning, C. D. (2013). Better word representations with recursive neural networks for morphology. In *CoNLL.*

Meireles, M. R. G., Ferraro, G., & Geva, S. (2016). Classification and information management for patent collections: A literature review and some research questions. *Information Research* 21(1), paper 705.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). Efficient Estimation of Word Representations in Vector Space. arXiv:13013781v3.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems, MountainView.*

Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data, 2*(1), 1.

Park, H., Yoon, J., & Kim, K. (2013). Identification and evaluation of corporations for merger and acquisition strategies using patent information and text mining. *Scientometrics, 97*(3), 883–909.

Park, Y., Yoon, J., & Phillips, F. (2017). Application technology opportunity discovery from technology portfolios: Use of patent classification and collaborative filtering. *Technological Forecasting & Social Change, 118,* 170–183.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *EMNLP*.

Piroi, F., Lupu, M., Hanbury, A., & Zenz, V. (2011). CLEF-IP 2011: Retrieval in the intellectual property domain. In *CLEF 2011 labs and workshop, notebook papers, 19–22 September 2011, Amsterdam, The Netherlands*.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986) Learning representations by back-propagating errors [J]. *Nature*, *323*(6088), 533–536.

Sahlgren, M. (2008). The distributional hypothesis. *Italian Journal of Linguistics, 20*(1), 33–54.

Taddy, M. (2015). Document classification by inversion of distributed language representations. arXiv preprint arXiv:1504.07295.

Taylor, G. W., Fergus, R., LeCun, Y., & Bregler, C. (2010). Convolutional learning of spatio-temporal features. In *European conference on computer vision*.

Trappey, A. J., Hsu, F.-C., Trappey, C. V., & Lin, C.-I. (2006). Development of a patent document classification and search platform using a back-propagation network. *Expert Systems with Applications, 31*(4), 755–765.

Trappey, A. J., Trappey, C. V., Wu, C.-Y., & Lin, C.-W. (2012). A patent quality analysis for innovative technology and product development. *Advanced Engineering Informatics, 26*(1), 26–34.

Verberne, S., & D'hondt, E. (2011). Patent classification experiments with the linguistic classification system LCS in CLEF-IP 2011. In *CLEF (notebook papers/labs/workshop)*.

Verberne, S., Vogel, M., & D'hondt, E. (2010). Patent classification experiments with the linguistic classification system LCS. In *CLEF (notebook papers/LABs/workshops)*.

Wagner, S., & Wakeman, S. (2016). What do patent-based measures tell us about product commercialization? Evidence from the pharmaceutical industry. *Research Policy, 45*(5), 1091–1102.

WIPO. (2018). *World intellectual property indicators 2017*. Geneva: World Intellectual Property Organization.

Wu, C.-H., Ken, Yun, & Huang, Tao. (2010). Patent classification system using a new hybrid genetic algorithm support vector machine. *Applied Soft Computing, 10*(4), 1164–1177.

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Advances in neural information processing systems*.