

Supervised Approaches to Assign Cooperative Patent Classification (CPC) Codes to Patents

Tung Tran¹ and Ramakanth Kavuluru^{1,2}(✉)

¹ Department of Computer Science, University of Kentucky, Lexington, KY, USA
{tung.tran,ramakanth.kavuluru}@uky.edu

² Division of Biomedical Informatics, Department of Internal Medicine,
University of Kentucky, Lexington, KY, USA

Abstract. This paper re-introduces the problem of patent classification with respect to the new Cooperative Patent Classification (CPC) system. CPC has replaced the U.S. Patent Classification (USPC) coding system as the official patent classification system in 2013. We frame patent classification as a multi-label text classification problem in which the prediction for a test document is a set of labels and success is measured based on the micro-F1 measure. We propose a supervised classification system that exploits the hierarchical taxonomy of CPC as well as the citation records of a test patent; we also propose various label ranking and cut-off (calibration) methods as part of the system pipeline. To evaluate the system, we conducted experiments on U.S. patents released in 2010 and 2011 for over 600 labels that correspond to the “subclasses” at the third level in the CPC hierarchy. The best variant of our model achieves $\approx 70\%$ in micro-F1 score and the results are statistically significant. To the best of our knowledge, this is the first effort to reinitiate the automated patent classification task under the new CPC coding scheme.

1 Introduction

A patent can be briefly summarized as a contract between an inventor and a government entity that prevents others from using or profiting from an invention for a fixed period of time; in return, the inventor discloses the invention to the public for the common good. Patents are complex technical and legal documents that exhaustively detail the ideas and scopes of corresponding inventions. The United States Patent and Trademark Office (USPTO) is the government agency responsible for issuing and maintaining patents in the United States. The first U.S. patent was issued in 1790 [26] and since then the USPTO has issued over 9 million patents¹. Hundreds of thousands of patent applications are submitted to the USPTO each year and data suggests that this number will continue to rise heavily in the future. In 2015, there were 589,410 utility patent applications compared to 490,226 in 2010 and 390,733 in 2005 [22].

¹ See Patent No. 9,226,437, granted Jan. 5, 2016. Available at: <http://pdfpiw.uspto.gov/.piw?Docid=09226437>.

A patent application undergoing the review process must meet the novelty criteria in order for it to be published; i.e., it must be original with respect to past inventions. Hence, a *patentability search* is usually performed by inventors, lawyers, and patent examiners to determine whether the patent constitutes a novel invention. The manual process can be a strain on time and resource because of the scientific expertise needed to verify patentability. A patent classification system (PCS) is vital for organizing and maintaining the vast collection of patents for later lookup and retrieval. We first provide a brief dissection of a typical patent document before describing the various PCSs and formalizing the problem of assigning PCS codes to a patent. Since we are primarily concerned with patents in a U.S. context, mentions of patents in the remainder of this paper implicitly refer to U.S. patents unless otherwise stated.

A patent consists of several sections including title, abstract, description, and claims. From our observation, the title and abstract are consistent in length with those of a typical research paper, while description is much larger in detail and scope. The claims section is unique in that it describes the invention in units of “innovation”; each claim corresponds to a novel aspect of the invention and is conveyed in nuanced legal terminology. According to Tong et al. [20], the claims of a patent can actually be considered a collection of separate inventions; together they can be used to determine the true measure of a patent. A patent document additionally contains structured bibliographical information such as inventors, lawyers, publication date, application date, application number, and technology class assignments. Technology class assignments are available for each of the three PCSs: the U.S. Patent Classification (USPC) system, the Cooperative Patent Classification (CPC) system, and the International Patent Classification (IPC) system. The USPC has been the official PCS used and maintained by the USPTO since the first patent was issued. In January 1, 2013, however, it was replaced by the CPC system as a joint effort by the USPTO and European Patent Office (EPO) to promote patent document compatibility at the international level [24]. The CPC is intended to be a more detailed extension of the International Patent Classification (IPC) system². As part of the transition, all US patent documents as of 1836 have been retroactively annotated with CPC codes using “an electronic concordance system” [18].

In CPC, the classification terms/labels (CPC codes) are organized in a taxonomy – a tree with each child label being a more specific classification of its parent label; that is, there is an *IS-A* relation between a label and its parent. A single patent can be manually assigned one or more labels corresponding to the leaf nodes (of the taxonomy) by a patent examiner. There are five levels of classification: *section*, *class*, *subclass*, *group*, and *subgroup*. As of January 2015, there are 9 sections, 127 classes, 654 subclasses, 10633 groups, 254794 subgroups in the CPC schema [23]. An example of a leaf label and its parent labels can be

² According to the Guide to the CPC [24], “unless stated otherwise, the rules and principles are identical” to that of IPC, which can be found in Guide to the IPC [27].

Table 1. Overview and example of CPC hierarchical taxonomy. The label count at each level is computed from the January 2015 version of the CPC scheme [23].

Level	Count	Example Label	Example Label Description
Section	9	H	Electricity
Class	127	H01	BASIC ELECTRIC ELEMENTS
Subclass	654	H01C	RESISTORS
Group	10633	H01C 3	Non-adjustable metal resistors made of wire or ribbon, e.g. coiled, woven or formed as grids
Subgroup	254794	H01C 3/08	Dimension or characteristic of resistive element changing gradually or in discrete steps from one terminal to another

observed in Table 1. Since CPC is an extension of IPC, many of the characteristics of CPC as described can be similarly observed in IPC.

In this paper, we propose a supervised machine learning system for the classification of patents according to the newly implemented CPC system. To our knowledge, ours is the first automatic patent classification effort for CPC based on literature review. Our system exploits the hierarchical nature of the CPC taxonomy as well as the citation records³. CPC codes appear in order of how adequately a code represents the invention [27]; here, we neglect the ordering and treat the problem as a multi-label classification problem. That is, our prediction is a set of labels, a task that is more challenging and comprehensive compared to past work that focuses on predicting a single “main” IPC label for each test patent. This is because many inventions span across multiple technological domains and this level of nuance cannot be captured in a single CPC code. As in past work that deal with the older IPC system, we collapse all CPC labels of a patent to their subclass representations and make predictions at the *subclass* level (row 3 of Table 1). We use real-world patent documents published by the USPTO in the years 2010 and 2011 to train, tune, and evaluate our proposed models. Moreover, we publicly release the dataset⁴ used in training and evaluating our system to stimulate further research in the area.

2 Related Work and Background

Fall et al. [4] explored the task of patent classification for IPC using various supervised algorithms such as support vector machines (SVM), naïve Bayes, and

³ These are other older patents that appear in the *References Cited* section and are sometimes mentioned within the description of a new patent being considered for automated coding.

⁴ <http://patents.ttran.net>.

k -nearest neighbors (k -NN). Their experiments used the title, first 300 words, and claims as the predictive scope for feature extraction. The system proposed does not attempt to make predictions on the correct set of labels but rather produces an exhaustive label ranking for each patent on which custom “precision” metrics are used to evaluate performance. For instance, the prediction for a test document is deemed correct if the top-ranked predicted label matches the first label of the patent. The authors also conducted experiments with variants of this metric such as whether any of the top-3 prediction labels matches the first label of the patent or whether the top-ranked prediction appears anywhere in the ground truth list of IPC labels. Their study concluded that SVM was superior to other methods under similar conditions.

Liu and Shih [11] proposed a hybrid system for USPC classification using patent network analysis in addition to traditional content-based features. Their approach is based on first constructing a network graph with patents, technology classes, inventors, and assignees as nodes. Edges indicate connectivity and edge weights are computed using a custom relation metric. A prediction for a test patent is made by looking at its nearest neighbors based on a relevance measure as determined by the constructed patent network. Li et al. [10] exploit patent citation records by proposing a graph kernel that captures informative features of the network. Richter and MacFarlane [14] showed that in some cases using features based on bibliographical meta-data can improve classification performance. Other studies [2, 8] found that exploiting the semantic structure of a patent such as its individual claims can result in similar gains. Automatic patent classification in the literature has primarily focused on IPC [2–4] or USPC [10, 11], and we observe k -NN to be a popular approach for many proposed systems [8, 11, 14]. When targeting IPC, classification is typically performed at the class or subclass level. This choice is motivated by the fact that labels at the subclass level are fairly static moving forward while group and subgroup labels are more likely to undergo revision with each update [4] of PCS system.

3 Datasets

The dataset used in our experiments consist of utility patent documents published by the USPTO in 2010 and 2011, not including pending patent applications. For the 2010 and 2011 datasets, there are 215,787 and 221,206 patent documents respectively. Specifics about training and test set splits for supervised experiments are outlined in Sect. 5. The patents documents are freely available in HTML format at the USPTO website, although not in a readily machine-processable format. As outlined earlier, each patent contains the text fields *title*, *abstract*, *description*, and *claims*. Furthermore, each document is annotated with a set of one or more CPC labels. From the dataset, we counted 613 unique CPC subclasses which correspond to the range of candidate labels for this predictive task.

The distribution of CPC subclasses is skewed with some codes such as *G06F* (Electrical Data Processing), *H01L* (Semiconductor Devices), and *H04L* (Transmission of Digital Information) dominating the patent space at 5.9%, 4.6%, and

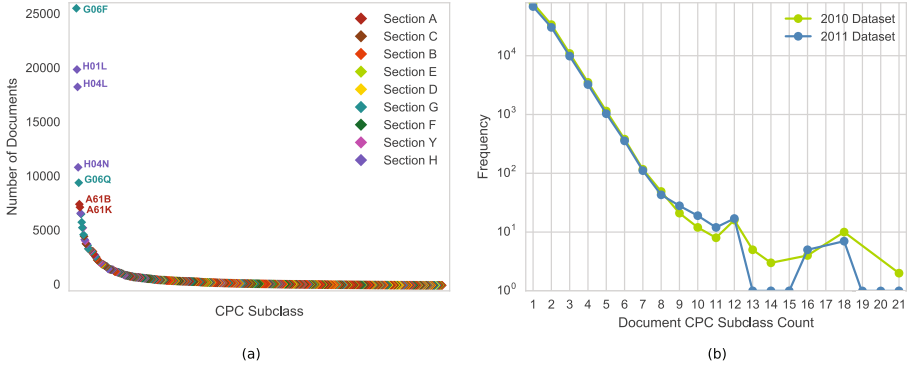


Fig. 1. Overview of CPC subclass level statistics within the 2010 and 2011 datasets for (a) CPC subclasses along the x -axis in order of label frequency—only those above the 99th percentile are labeled—and (b) frequency of the number of CPC subclasses that appear in a document.

4.1% document assignments respectively as seen in Fig. 1(a). The distribution of the number of CPC subclass assignments per document is likewise skewed such that the average number of labels per patent is only 1.76. Indeed, approximately 60% of documents contain only a single subclass while some outlier documents may have as many as 21 subclasses as shown in Fig. 1(b).

4 Methods: Label Scoring, Reranking, and Thresholding

As indicated earlier, the CPC taxonomy is hierarchical and takes upon a tree structure so that each label exists as a non-root node in the tree; henceforth, we refer to nodes and labels interchangeably. Since we are concerned with classification at the *subclass* (or third) level in the hierarchy, nodes at the *subclass* level are considered leaf-nodes in our experiments⁵. However, in practice it is possible to choose any target depth d in the hierarchy as the level at which labels are trained on (and predictions are made) essentially assuming d is the leaf-node level. The following notation is used in the formulation of our methodology. Let L^i be the set of all labels at level i in the CPC tree. For a leaf node $c \in L^d$, we define $[c]_i$ as the ancestor node of c at level i such that $[c]_i \in L^i$ and $i \leq d$. Given the tree structure, a node has unique ancestors at each level above it. As a special case, we have $c = [c]_i$ if $i = d$.

⁵ Although code assignment in reality is only done at the subgroup level, we can collapse such deeper codes to the subclass level by simply removing code components specific to the deeper nodes. In Table 1, code for resistors (row 3) can be obtained from the more specific subgroup code (row 5) by removing the “3/08” part. Typically, due to extremely high sparsity, code assignments are carried out at a higher level in preliminary studies including our current attempt and other automated PCS efforts.

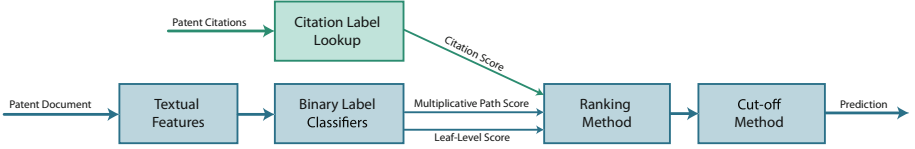


Fig. 2. The pipeline of the proposed framework. The core framework produces scores for each patent while the ranking and cut-off components jointly make a label set prediction based on these scores.

Basic multi-label pipeline. Figure 2 represents the high level skeleton of our approach, a pipeline setup that is quite common in multi-label classification scenarios [28] where binary classifiers, one per label, are used first based on n -gram features. Besides scores from these classifiers, additional domain-specific features that are not directly related to the lexical n -gram features of the text are also used to score the labels. Subsequently, the scores are used to rank all labels for a given input instance. Given the sparsity concerns, the default threshold of the binary classifiers might not be suitable for infrequent labels. So instead a so called cut-off or calibration method [5, Sect. 4.4] is used to make a partition of all labels where the top few labels that form one of the partitions are considered as the final predictions. This approach has been used in obtaining medical subject headings for biomedical research articles [12, 15] and assigning diagnosis codes to electronic medical records [7]. We will elaborate on each of the components in Fig. 2 in the rest of this section with a focus on novel CPC and patent specific variations we introduce in this paper. But first, we outline the configuration for the binary label specific classifiers.

Lexical features for binary classifiers. For the textual features, we extract unigram and bigram features from the title, abstract, and description fields. We found that including textual features from the claims field tend to have a neutral or negative impact on performance while adding drastically to the feature space. We suspect that unigram and bigram features are unable to adequately capture the nuance in legal language and style as presented in the itemized claims section. To further reduce the feature space, we apply a lowercase transformation prior to tokenization, remove 320 popular English stop words, and ignore terms that occur in fewer than ten instances in the entire corpus. We apply the well-known *tf-idf* transformation [16] to word counts to produce the final document-feature matrix. In total, there are nearly 6.8 million n -gram features in the final document-feature matrix.

Training label-specific classifiers. Our system uses a top-down approach [17] such that a local binary classifier is trained for each label in the taxonomy (including the non-leaf nodes). Let f_c be the classifier associated with the label $c \in L^1 \cup \dots \cup L^d$. The classifier f_c is trained on a set of positive and negative examples, such that the set of positive example set $[M_c]^+$ includes only documents that are labeled with c and the set of negative examples $[M_c]^-$ is the set of patents that are not

labeled with c . Since there typically many more negative examples than positive examples for a given label, we use only a random subsample of negative examples such that $|[M_c]^+| = |[M_c]^-|$. This under-sampling of the majority class is a well-known idea that seems to fare well in imbalanced datasets [25]. A simple logistic regression (LR) model is used for each classifier which expresses the output in the form of a probability estimate in the range of $[0, 1]$.

4.1 Label Scoring

Since the classification task is multi-label, the prediction for some patent x is necessarily a subset of all possible labels at level d in the CPC hierarchy. A natural approach is to rank the labels based on some *scoring* function with respect to x , then truncate the ranked list after some cut-off k such that only the top k labels appear in the predicted set. We define three such scoring functions to be used as the basis for our label ranking methods: leaf-level score, hierarchical multiplicative-path score, and citation score. Let $L^i(x)$ be the set of labels at level i assigned to patent x . Each scoring function takes two parameters: input patent x and a leaf-node label $c \in L^d$.

The leaf-level score

$$S_L(c, x) = f_c(x) \quad (1)$$

is simply the probability $f_c(x)$ output by the binary classifier for c at level d in the hierarchy. Since the problem is *mandatory leaf-node prediction*, using this score alone in label ranking is referred to as *flat classification* in the literature [17]. The multiplicative-path score is the geometric mean⁶ of probability outputs for each classifier along the path from the leaf-node label to the root, or more formally,

$$S_M(c, x) = \left(\prod_{i=1}^d f_{[c]_i}(x) \right)^{1/d}. \quad (2)$$

Both $S_M(c, x)$ and $S_L(c, x)$ produce a real number in $[0, 1]$. Next, we define the citation score that is specific to the patent domain. Let $R(x)$ be the set of patents cited by x . For a given patent x and a candidate label c , the raw citation score

$$S_R(c, x) = |\{t : t \in R(x) \wedge c \in L^d(t)\}|, \quad (3)$$

which intuitively counts the number of cited patents that also are assigned our candidate code c . This is not unlike the k -NN approach where similar instances' code sets are exploited to score candidate codes for an input instance. However, in this work, we simply used the cited patents as neighbors drastically reducing the notorious test time inefficiency issues for nearest neighbor models. The citation

⁶ The product of d probabilities is significantly reduced in magnitude especially for large values of d . Here, use of the geometric mean is intended to restore the magnitude of the resulting product while ensuring that it remains in $[0, 1]$. This eases the interpretability of intermediate probability outputs while guarding against practical concerns such as floating point underflow.

score above is simply a count but we linearly rescale it across all leaf-level labels such we obtain a value in $[0, 1]$ range via

$$S'_R(c, x) = \frac{S_R(c, x) - \min Z(x)}{\max Z(x) - \min Z(x)} \quad (4)$$

where $Z(x) = \{S_R(c, x) : c \in L^d\}$.

4.2 Label (re)ranking

The three different scoring functions in Eqs. (1), (2), and (4) offer evidences of the relevance of a label c for a test instance x . At this point, we do not know how much each of these contributes to an effective ranking model. Thus to leverage them, we combine them into a final score – for the purpose of label ranking – using two approaches

1. The first approach involves a *grid search* [6] over weights $p_1 \geq 0$ and $p_2 \geq 0$, where $0 \leq p_1 + p_2 \leq 1$, for the combined scoring function

$$S(c, x) = p_1 \cdot S_L(c, x) + p_2 \cdot S_M(c, x) + (1 - p_1 - p_2) \cdot S'_R(c, x), \quad (5)$$

that maximizes the micro F1-score over a validation dataset. Note that the coefficients of the three constituent functions in Eq. (5) add up to 1. Hence, given each score is also in $[0, 1]$, this ensures $S(c, x) \in [0, 1]$.

2. The second approach uses what is known in the literature as *stacking* [9], a popular meta-learning trick. Here, we train a new binary LR classifier for each label at level d using scores S_L , S_M , and S'_R computed from each patent example in the validation set as features.

4.3 Label Cut-Off/Thresholding

As mentioned earlier, once we have a final ranking among leaf labels for some patent x , we need to choose the optimal cut-off k tailored specifically to x . Thus only the top k labels are included in the final prediction. One simple approach is to train a linear regression model to predict the number of labels based on core n -gram features [12] and a few select domain-specific features. We found that adding length-based features such as character-count and word-count of text fields resulted in poor performance. Instead, the following features were selected given a patent x : the number of claims made by x , the number of inventors associated with x , and descriptive statistics based on the distribution of known label counts associated with the set of citations $R(x)$. We refer to this approach as the *linear* cut-off method.

The linear cut-off approach will serve as a reasonable baseline. However, the skewed distribution of label counts presents a source of difficulty. Approximately 60.95% and 26.39% of patents in the 2010 dataset have one and two labels respectively. The remaining 12.66% of patents are exceptions in that they have more than two labels with some patents having as many as 21 labels. A simple

linear regression approach has the tendency to overestimate the label count, as such we consider a more sophisticated version of this method using a two-level top-down learning model. In the first level, we train a binary classifier to predict whether, for some test patent, the case is common (one or two labels) or an exception (three or more labels). If it is predicted common, we train another binary classifier at the second layer to predict whether the number of labels is one or two. If it is predicted to be the exception case, we instead use a linear regression model to predict a real number value, which is rounded to the nearest natural number and used as the outcome for the label-count predictor. We refer to this approach as the *tree* cut-off method and is inspired by the so called “hurdle” process in regression methods for count data [1].

Finally, we propose a more involved cut-off method that does not rely on supervised learning. In this alternative method, referred to as *selective cut-off*, there are two steps. (1) Recall our final scoring function $S(c, x)$ from Eq. (5) used to rank labels before applying cut-off. Suppose the label ranking is $c_{i_1}, \dots, c_{i_{|L^d|}}$, $i_j \in \{1, \dots, |L^d|\}$, we choose the cut-off k based on

$$\operatorname{argmax}_{k \in \{1, \dots, |L^d| - 1\}} S(c_{i_k}, x) - S(c_{i_{k+1}}, x).$$

This has the effect of choosing the cut-off at the greatest “drop-off point” in the ranking with respect to the score. (2) Performing the first step alone is prone to overestimating the actual k , so additional pruning is required. Suppose the top k labels are c_{i_1}, \dots, c_{i_k} . As a second step, we further remove a label from this ranking if its rank is higher than the average label count over all patents (in training data) to which it is assigned. Let A_j be the mean label count over all patents in the training set with label c_{i_j} for some $1 \leq j \leq k$. We remove c_{i_j} from the final list if $j > A_j$. Intuitively, for example, a label that typically appears in label sets of size 3 on average is not likely to rank 4th or higher when generalizing to unseen examples. In a way, this leverages the thematic aspects of patents and how certain themes typically lend themselves to a narrow or broad set of patent codes.

5 Experiment and Results

For our experiments, we optimized hyperparameters (from Eq. (5)) on the 2010 dataset using a split of 70% and 30% for training and development, respectively. We use this 2010 dataset exclusively for tuning hyperparameters. We evaluate the variants of our system by training and testing on the 2011 dataset (with the 70%–30% split) using the hyperparameters optimized on the 2010 dataset to simulate the prediction process – learn on existing data to predict for future patents. Since we are framing it as a multi-label problem with class imbalance concerns, we measure classification performance based on the popular micro-averaged F1 [21] measure. We evaluate all combinations of the proposed ranking and cut-off methods on top of the core framework (as described in Sect. 4). For each variant of our system, we perform 30 experiments each with a

random train-test split of the 2011 dataset and record the micro-averaged F1/Precision/Recall as shown in Table 2. The macro-averaged F1 measure that gives equal importance to all labels regardless of their frequency is additionally recorded in Table 3.

Table 2. Results comparing variants of our method in micro-averaged F1

	Micro-P (%)	Micro-R (%)	Micro-F (%)
Grid Ranking + Linear Cut-off	69.951 ± 0.072	68.392 ± 0.051	69.146 ± 0.034
Grid Ranking + Tree Cut-off	69.926 ± 0.059	68.400 ± 0.046	69.154 ± 0.037
Grid Ranking + Selective Cut-off	73.749 ± 0.057	66.418 ± 0.040	69.892 ± 0.033
Stacked Ranking + Linear Cut-off	63.391 ± 0.236	47.804 ± 0.175	54.505 ± 0.200
Stacked Ranking + Tree Cut-off	62.517 ± 0.230	48.799 ± 0.190	54.812 ± 0.206
Stacked Ranking + Selective Cut-off	63.423 ± 0.238	47.52 ± 0.177	54.505 ± 0.200

Table 3. Results comparing variants of our method in macro-averaged F1

	Macro-F (%)
Grid Ranking + Linear Cut-off	74.281 ± 0.031
Grid Ranking + Tree Cut-off	74.274 ± 0.031
Grid Ranking + Selective Cut-off	74.262 ± 0.032
Stacked Ranking + Linear Cut-off	57.452 ± 0.214
Stacked Ranking + Tree Cut-off	57.652 ± 0.214
Stacked Ranking + Selective Cut-off	57.403 ± 0.214

Among the proposed approaches, the *grid* based ranking and *selective* cut-off combination achieved a micro-F1 of almost 70%, which is best performance (row 3 of Table 2). Based on non-overlapping confidence intervals, we also see the differences in performance with respect to other variants is statistically significant for this particular combination. From Table 2, it is clear that the *stacked* ranking approach performs poorly compared to the *grid* ranking approach and is observed to be approximately 15 points worse in micro-F1 due to the low recall. Among variants that use the *grid* ranking method, the *selective* cut-off approach is able to achieve superior precision with only a minor dip in recall. It can also be observed that, in terms of macro-F1, the *grid* ranking approach is still superior by far; with *grid* ranking, *linear* cut-off has a higher average macro-F1 over

other cut-off methods but the gains are not statistically significant. Moreover, we note that the macro-F1 score is greater than the micro-F1 score for each respective method combination, which may suggest that the system is generally performing better on the low-frequency labels and worse on the popular ones [19]. This is a counterintuitive outcome and needs further examination as part of our future work. However, it could also be due to the case that some infrequent labels are very specific to certain esoteric domains for which the language used in the patents is highly specific.

6 Conclusion

In this paper, we proposed an automated framework for classification of patents under the newly implemented CPC system. Our system exploits the CPC taxonomy and citation records in addition to textual content. We have evaluated and compared variants of the proposed system on patents published in years 2010 and 2011. As a take away, we demonstrated that using the proposed framework with *grid* ranking (based on three different scoring functions) and the *selective* cut-off method outperform other variants of the system. In this work, we used logistic regression as the base classifier since it is fast and uses relatively fewer parameters. In future work, we propose to combine the information in the CPC hierarchy as a component of recurrent and convolutional neural networks for multi-label text classification using the cross-entropy loss [13].

Acknowledgements. We thank anonymous reviewers for their honest and constructive comments that helped improve our paper’s presentation. Our work is primarily supported by the National Library of Medicine through grant R21LM012274. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

References

1. Cameron, A.C., Trivedi, P.K.: Regression Analysis of Count Data, vol. 53. Cambridge University Press, Cambridge (2013)
2. Don, S., Min, D.: Feature selection for automatic categorization of patent documents. *Indian J. Sci. Technol.* **9**(37), 1–17 (2016). Kindly check and confirm the edit made in Ref. [2]
3. Eisinger, D., Tsatsaronis, G., Bundschuh, M., Wieneke, U., Schroeder, M.: Automated patent categorization and guided patent search using IPC as inspired by MeSH and PubMed. *J. Biomed. Semant.* **4**(S1), 1–23 (2013)
4. Fall, C.J., Töröcsvári, A., Benzineb, K., Karetka, G.: Automated categorization in the international patent classification. *ACM SIGIR Forum* **37**(1), 10–25 (2003)
5. Gibaja, E., Ventura, S.: A tutorial on multilabel learning. *ACM Comput. Surv. (CSUR)* **47**(3), 52 (2015)
6. Hsu, C.-W., Chang, C.-C., Lin, C.-J., et al.: A practical guide to support vector classification (2003). <https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>

7. Kavuluru, R., Rios, A., Lu, Y.: An empirical evaluation of supervised learning approaches in assigning diagnosis codes to electronic medical records. *Artif. Intell. Med.* **65**(2), 155–166 (2015)
8. Kim, J.-H., Choi, K.-S.: Patent document categorization based on semantic structural information. *Inf. Process. Manag.* **43**(5), 1200–1215 (2007)
9. Kotsiantis, S.B., Zaharakis, I., Pintelas, P.: Supervised machine learning: a review of classification techniques (2007)
10. Li, X., Chen, H., Zhang, Z., Li, J.: Automatic patent classification using citation network information: an experimental study in nanotechnology. In: *Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries*, pp. 419–427. ACM (2007)
11. Liu, D.-R., Shih, M.-J.: Hybrid-patent classification based on patent-network analysis. *J. Assoc. Inf. Sci. Technol.* **62**(2), 246–256 (2011)
12. Liu, K., Peng, S., Wu, J., Zhai, C., Mamitsuka, H., Zhu, S.: MeSHLabeler: improving the accuracy of large-scale MeSH indexing by integrating diverse evidence. *Bioinformatics* **31**(12), i339–i347 (2015)
13. Nam, J., Kim, J., Loza Mencía, E., Gurevych, I., Fürnkranz, J.: Large-scale multi-label text classification — revisiting neural networks. In: *Calders, T., Esposito, F., Hüllermeier, E., Meo, R. (eds.) ECML PKDD 2014. LNCS (LNAI)*, vol. 8725, pp. 437–452. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44851-9_28
14. Richter, G., MacFarlane, A.: The impact of metadata on the accuracy of automated patent classification. *World Pat. Inf.* **27**(1), 13–26 (2005)
15. Rios, A., Kavuluru, R.: Analyzing the moving parts of a large-scale multi-label text classification pipeline: experiences in indexing biomedical articles. In: *2015 International Conference on Healthcare Informatics (ICHI)*, pp. 1–7. IEEE (2015)
16. Salton, G., McGill, M.J.: *Introduction to Modern Information Retrieval*. McGraw-Hill, New York (1986)
17. Silla, C.N., Freitas, A.A.: A survey of hierarchical classification across different application domains. *Data Min. Knowl. Discov.* **22**(1–2), 31–72 (2011)
18. Simmons, H.J.: Categorizing the useful arts: part, present, and future development of patent classification in the United States. *Law Libr. J.* **106**, 563 (2014)
19. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. *Inf. Process. Manag.* **45**(4), 427–437 (2009)
20. Tong, X., Frame, J.D.: Measuring national technological performance with patent claims data. *Res. Policy* **23**(2), 133–141 (1994)
21. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. In: *Maimon, O., Rokach, L. (eds.) Data Mining and Knowledge Discovery Handbook*, pp. 667–685. Springer, Boston (2010). https://doi.org/10.1007/978-0-387-09823-4_34
22. U.S. Patent and Trademark Office: U.S. Patent Statistics Chart. https://www.uspto.gov/web/offices/ac/ido/oeip/taf/us_stat.htm (2016). Accessed 30 Nov 2016
23. U.S. Patent and Trademark Office and European Patent Office: Cooperative Patent Classification Scheme in Bulk. <http://www.cooperativepatentclassification.org/cpcSchemeAndDefinitions/Bulk.html> (2015). Accessed 01 Feb 2015
24. U.S. Patent and Trademark Office and European Patent Office: Guide to the CPC. <http://www.cooperativepatentclassification.org/publications/GuideToTheCPC.pdf> (2015). Accessed 30 Nov 2016
25. Wallace, B.C., Small, K., Brodley, C.E., Trikalinos, T.A.: Class imbalance, redux. In: *2011 IEEE 11th International Conference on Data Mining (ICDM)*, pp. 754–763. IEEE (2011)

26. Wolter, B.: It takes all kinds to make a world-some thoughts on the use of classification in patent searching. *World Pat. Inf.* **34**(1), 8–18 (2012)
27. World Intellectual Property Organization: Guide to the IPC. http://www.wipo.int/export/sites/www/classifications/ipc/en/guide/guide_ipc.pdf (2016). Accessed 30 Nov 2016
28. Zhang, M.-L., Zhou, Z.-H.: A review on multi-label learning algorithms. *IEEE Trans. Knowl. Data Eng.* **26**(8), 1819–1837 (2014)