**August 19, 2019**

**From**: Michael Bryan, Partner, Open Environments LLC

**To**: Andrew Toole, Chief Economist, United States Patent & Trademark Office

**Field:** U.S. Federal Administration of Intellectual Property

**Title:** Multilabel Classification of Patent Text

Thank you for your engagement and support in studying U.S. adoption of the IPC. The proceeding report summarizes three major issues and applies statistical methods to their resolution. This document is complemented by programming files used to collect the data and reproduce the study’s analytical results.

# Project

In 2015, the U.S. Patent & Trademark Office (USPTO) adopted an international standard, tagging each new patent application with one or several codes based on the ideas involved (IPC). Data on the first full year of patents under this standard, 2018, is now available, and begs an assessment of the new coding scheme. This study applies machine learning methods to the year of patent text to assess opportunities for automation and data quality.

## Background

Economies that grow beyond a local scale will critically protect intellectual property to promote innovation. The framers of the U.S. Constitution recognized this in the document’s first provisions (Article I § 8 Clause 8). As world economies integrate, shared patent protection becomes a matter of global security. The Paris Convention in 1883 gave roots to the World Intellectual Property Organization (WIPO) which defines the terms of patents as well as treaties for process coordination.

More recently, the Strasbourg Agreement of 1971 called for an International Patent Classification (IPC) scheme to categorize patent subjects across languages. After decades of development, the United States converted to the IPC standard in 2015, applying its codes to all new patent applications. In 2018, U.S. Patent & Trademark Office (USPTO) published the year’s 309,360 patent grants tagged with 987,494 IPC codes.

While it serves the federal patent application process, the IPC is also critical to the global intellectual property industry. As inventors claim new ideas, they may coin language and challenge conventional thinking. Naturally, they need the abilities to search for competing ideas, to judge distance and to anticipate change. The IPC defines the space of intellectual property for these purposes.

The impact of IPC adoption poses several problems for government, however. First, the IPC codes are relatively young. In practice, codes may prove overly specific or general to be meaningful. Second, economic and scientific innovation will cause the IPC scheme to actively change. If the IPC scheme is maintained at the pace of international relations, it may fail to keep up with innovation and undermine WIPO’s mission. Lastly, the IPC is granular with approximately 7,400 codes at its main group level. Relating its detail to every new patent application is initially a manual process raising cost and quality concerns.

## Research Goals

In more specific terms, this study addresses the following questions:

**1.2.1. Are IPC codes assigned evenly?**

Any given time period will be subject to innovation at various paces, so code use frequency should have high variation. However, the IPC scheme should evenly separate ideas. Any overuse of a code would suggest areas of innovation that may warrant code subdivision. The study will apply objective tests for monitoring the IPC scheme needs for change.

**1.2.2. Can machine learning be applied to automate IPC code assignment?**

The USPTO uses multiple third party vendors to perform and audit this labor intensive work. Automation would address these costs as well as the mistakes and bias from human intervention.

**1.2.3. Can anomalous IPC assignments be detected with analytical methods?**

Patents by nature seek to define new processes and materials. As a result, a patent’s text will avoid language that makes it comparable to other patents. An automated solution that identifies probable classes should be able to also identify improbable existing assignments.

## Statistical Questions

The research questions above introduce questions for the supporting statistical methods:

**1.3.1. What distribution and parameters are appropriate to code frequency?**

Evaluating code frequency will involve testing with a probability distribution. Each probability distribution is defined distinct parameters. Any identified distribution should include methods for maintaining its parameter values.

**1.3.2 What methods are appropriate for learning against patent text?**

An analytical model will need numeric methods for interpreting patent text. These word embedding methods will vary in their fitness for patent language, computational performance and predictive accuracy.

## Variables of Interest

While the dataset is large, the variables in this study are few. The data includes an identifier for each patent, the corresponding patent text and the several IPC codes assigned by the patent processors. **Table 1** below describes each variable used in the study and provides a reference to the WIPO standard source name (ST36). All variables are textual.

The variables are taken from the USPTO source files and staged in a relational database. The staging database has a table for the patent text and another for the assigned IPC codes. Both objects are indexed by the patent publication number. The IPC code assignment table is additionally indexed by IPC code number. No further reference elements are needed for the study. Additional technical materials are available in the supplemental files listed in section 7.1 of the appendix.

Table : Variables of Interest

|  |  |  |
| --- | --- | --- |
| **Variable** | **Source Reference** | **Description** |
| pub | *us-patent-grant.us-bibliographic-data-grant.publication-reference.document-id.doc-number* | Publication Number that uniquely identifies each patent. |
| ipc | Concatenation of *section, class, subclass* and *main-group* within *us-bibliographic-data-grant.classifications-ipcr* | The IPC code including section, class, subclass and main group level. One to several for each patent publication. |
| txt | Concatenation of *us-patent-grant.description* and *us-patent-grant.abstract* | The combined text of each patent’s Abstract and Description. |

## Study Diagram

The study uses the USPTO’s patent publications for the 2018 calendar year. The source does not reflect an a priori design, but rather observed patents as granted. In addition, the research questions call for predictive neural network applications rather than analysis of variance (ANOVA) or regression methods. A study diagram is not meaningful in this context.

# Exploratory Data Analysis

## The Red Book

The USPTO publishes all new patent grants weekly in electronic form for dissemination across the intellectual property industry. Specifically, the USPTO’s online Bulk Data Storage System (BDSS) provides public access to the data product entitled “Patent Grants Full Text Data”. This publication is familiarly called the “Red Book” and includes all elements captured by the examination process excluding the digital images of patent drawings. BDSS stores each week’s publication in compressed XML formatted files.

From this data source, the study takes IPC code assignments as well as the narrative text for each patent grant as described in ***Variables of Interest*** above. From each of 2018’s weekly files, the study extracts each patent publication number and descriptive text, removing any formatting characters. In total, the source contains 6,699 distinct IPC codes assigned to 309,360 patents collectively 987,494 times. The data are stored in a small database 10.9GB in size.

The patent text includes 3,390,338 terms used 1,629,204,688 times. By comparison, the Oxford English Dictionary (OED) only has 171,476 words. The many specialized and low frequency terms present a problem for modelling. Words that quickly identify one or few patents will cause overfitting. An upper limit of the 100,000 most frequent words is appropriate.

Any predictive model will depend critically on the amount of text available on a candidate patent. The complexity and compute resources involved in natural language processing will need sufficient text. ***Figure 1*** below suggests that there are some grants with very little text. A floor limit of 5,000 characters text size is appropriate. Patents with text size below this limit will be removed from the study. This effect is summarized in **Table 2**: Data Available below.

Figure : Length of Patent Text

## International Patent Classification

The International Patent Classification (IPC) system is a scheme of codes used for categorizing patent content into subject areas. Operationally, IPC codes assist applicants and examiners in finding similar patent materials. Analytically, the codes provide a basis for economic analysis and public policy. As an international standard, the codes are designed for use across languages. They allow inventors to interchange patents among the 194 national patent offices around the world.

The European Patent Office (EPO) and the USPTO maintain the addition and retirement of codes in an independent public organization called Cooperative Patent Classification (CPC). More information can be found on the websites of both the WIPO and CPC.

The IPC is hierarchical with each progressive level subdividing its parent level. An individual code reflects a concatenated value of each of these levels. ***Figure 2*** provides an example of an IPC code with its components.

|  |  |  |  |
| --- | --- | --- | --- |
| The IPC code **H01F001/053** refers to: | | | |
|  | Section | H | Electricity |
|  | Class | H01 | Basic electric elements |
|  | Subclass | H01F | Magnets |
|  | Group | H01F001 | Magnetic materials |
|  | Subgroup | H01F001/053 | Magnetic rare earth metals |

Figure : IPC Code Example

The 2018 annual version of the IPC defines 7,434 codes at the main group level. The study observed 6,699 applied in by the USPTO in the study year. However, many were used fewer than 30 times limiting a study’s ability to predict. **Section 3.1** analyzes this issue in detail. The sparsely used codes may be removed from the analysis.

## Assignments

The USPTO assigns one or several IPC codes to each patent during the application process. In the study set, IPC codes were assigned to granted patents in aggregate 987,494 times. The number of codes assigned to each patent varies with a skewed distribution bounded at one. **Figure 3** below provides a visual of this distribution.

Figure : Number of Assigned IPCs

The reduction of grants for limited text combined with the reduction of codes for limited use jointly reduces the number of code assignments available for analysis. **Table 2** below details the effects on the study data set.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Patent  Grants | IPC  Codes | Code  Assignments |
| Observed in 2018 | 309,360 | 6,699 | 987,494 |
| Removed | 4,869 | 3,601 | 42,674 |
| Surviving for Analysis | 304,491 | 3,098 | 944,810 |

Table : Data Available

# Statistical Analysis

Each of the study’s research questions involves a different field of applied analytics. And, each field has its own particular methods and semantics. So, the analytical response will be organized by research question and introduced by the terminology of the relevant analytical field.

## Evenness of IPC Code Use

This question involves the field of statistical inference. Addressing the question will apply the following concepts:

**Hypothesis Test.** To make objective statements, the study will start with the view that codes are evenly used. Then, the study will calculate test measures, with a proposed sensitivity, to determine whether that original view can be rejected.

**Chi Square.** If IPC codes are used evenly, they should occur in a pattern comparable to a uniform probability distribution. Pearson’s Chi Square statistic will be used for the hypothesis test on this condition.

**Multiple Comparisons.** Code frequencies can be evaluated both collectively and within each level of the IPC code hierarchy. The study needs to test codes simultaneously, for a collective result as well as within each class and each subclass.

Simple counts of IPC Code assignment suggest their distribution is far from uniform. Of the 6,699 codes used in 2018’s patent grants, the top 67, or 1%, represent 38.2% of all code assignments. The most popular of the codes, G06F003, is involved in 2.1% of all patents alone. Among the least popular, 4,099 codes have 30 or fewer uses in the studied year. So, the bottom 55.14% of codes account for 0.02% of all assignments. These may be problematic for predictive modelling with weaker supporting data. **Figure 4** below illustrates this observation.

Figure 4: Uniformity of IPC Code Assignments

Applying inference, the study will first perform a family wide test. The first test will evaluate whether all observed usages are simultaneously uniform. Consider the use of each IPC having an individual probability function with mean vector observed in the study’s source data. Further, let represent the vector of means resulting from the same number of random variables with uniform probability distributions. The study can use the test for uniformity. In statistical notation:

|  |  |
| --- | --- |
|  | By default, the vector of IPC means reflects a vector from uniform distributions. Alternatively, the observed vector is not equal to one from a uniform distribution. |
|  | statistic on the study data compares observed frequency vector against a uniform distribution. |
|  | The distribution with , or 6698, degrees of freedom and even the most conservative significance level |
|  | The test statistic is greater than the critical value so the study can reject the null hypothesis in favor of the alternative, that the family of IPC distributions are not uniform. |

This is consistent with the initial observations on the study data, but it should not be surprising. The 7,434 potential IPC code values are not independent. They are expressly related in the design of the IPC code hierarchy. Further, the default view expects 7,434 individual functions to be uniform at the same time. The multiplicative chance that any one function is not uniform is high. Formally, a conservative Bonferroni adjustment would be applicable, but the result above is far from marginal.

The next test logically would evaluate uniform distributions within levels of the IPC hierarchy for any local uniformity. The method above would be applied iteratively to each of the IPCs classes then each of the IPCs subclasses. Each test has three potential outcomes. When only a single observation occurs uniformity is not meaningful. Otherwise the local test may pass or fail the method above. **Table 3** below summarizes the results of these tests. None of the class level tests passed the test of uniformity, but 48 of 631 subclasses may be uniform.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Single | Pass | Fail | Total |
| Classes | 0 | 0 | 120 | 120 |
| Subclasses | 24 | 48 | 559 | 631 |

Table : Tests of Local Uniformity

Section 7.1 of the Appendix provides a complete list of the tests within each subclass with the programming to reproduce them.

## Automating IPC Code Assignment

The task of assigning IPC codes has important traits that direct the types of algorithms that should be applied to its automation.

**Embeddings**. The field of Natural Language Processing (NLP) algorithms need text to be represented in some numeric form. In the study’s case, machine learning needs to have each patent text to be converted to some numeric data structure in a way that supports many documents with a variety of semantics. Associating text to a numeric vector is called word embedding.

**Dimension Reduction**. Patent language is legally specialized in general and further specialized by each of many subjects. If there are more than 100,000 words used, a square embedding matrix will scale to tens of billions of values. Dimension reduction methods simplify these large, sparse matrices into major theoretical dimensions. These smaller latent dimensions are not easily interpretable, but they serve to mathematically represent the meaning within words.

**Hierarchical Multilabel Classification (HMC)**. Classification algorithms assign objects to categories. Multilabel classification assigns objects to more than one category. For this study, a patent’s text is the object and IPC codes are the categories. And, the IPC code scheme has progressive layers so HMC describes the problem (Wehrmann).

**Artificial Neural Networks (ANNs)**. In the field of machine learning, ANNs refer to algorithms that are modeled off the human brain. They process long vectors of numbers through layers of functional nodes to output a new derived vector. For this study, a neural network will take a vector generated by text embedding to calculate a vector of probabilities that each IPC code would be assigned.

Building a model to automate IPC code assignment will follow three general steps. First, the study selects from available embedding methods appropriate to patent text as explored in section 2.1 above. A reference of words and their embeddings needs to be developed as the model’s input. Next, a loss function is designed. A neural network will teach its node functions to minimize some definition of incorrect output. Finally, the neural network can be trained, passing the embedded vectors of patent texts iteratively over its layers. Some best number of iterations will be found that minimizes training and test definitions of loss.

### Word Embeddings

The table below identifies four alternative word embedding methods commonly used in natural language processing solutions. A citation is provided relating each to detailed reference in section 7.4 of the Appendix. The study will use the GloVe algorithm, developed by Stanford University’s NLP Group in 2014 (Pennington). Specifically, it will apply pre-trained embeddings with 300 dimensions on 400,000 terms collected from the University of Pennsylvania’s LDC Gigaword dictionary (LDC). This embedding is available under public domain licensing terms from Stanford’s website (Stanford).

Table : Word Embedding Algorithms

|  |  |
| --- | --- |
| **Algorithm** | **Description** |
| **fastText**  (Joulin) | This method uses a neural network to build numeric vectors for each word based on letter series within the word. Facebook authors this method and provides pre-trained vector dictionaries in 157 languages.  By design, fastText lends itself to simpler, low dimensional, interpretation of colloquial text generalizing many, changing local semantics. Lightweight, it can be used on small devices. |
| **GloVe** (Pennington) | This method captures a word’s meaning by measuring its co-occurrence with other words. The word-word matrix is sparse and can be reduced to form a lower dimensional latent word vector space. Any new text, then, is interpreted by accumulating the text’s word vectors.  Calculating the GloVe vector space is computationally expensive, but only necessary once or occasionally as semantics evolve. Pre-calculated, generic GloVe vector dictionaries are freely available.  The language in patent descriptions includes specialized legal terms and may span industrial subjects. A single, integrated dictionary may not be realistic for the USPTO. |
| **word2vec**  (Mikolov) | This method attempts to predict words given their position in a continuous text. It includes two alternative architectures and the optional parameters of each. Like GloVe, word2vec generates word-based vectors. It results however in somewhat higher dimensional vectors covering larger subjects spaces. |
| **tf-idf**  (Jones) | This method treats each text as a vector of term frequencies. Each frequency is then divided by that term’s general frequency in all texts in a group. Variations on this method are used in document search and retrieval solutions.  tf-idf has two limitations. It depends upon a stable set of shared terms and, naturally it needs a bounded set of comparable documents. When the number of terms grows very large or documents grow diverse, the resulting document vectors generalize. |

### Loss Function

The definition of error in multilabel classification is not straight forward. The study identifies three issues specifically that guide the design of a customized loss function. These will be addressed in terms of the tensorflow (Google) package’s use in the proposed neural network. Specifically, the binary cross entropy function will be applied.

First, IPC code assignment reflects a rare event problem. On any given patent, only a few of the 3,098 available codes will be used. A model that predicts no code assignment at all would be 99.87% correct. The neural network can address this by using an autoencoder method to redefines the dimensions of the patent text (Liou, Cheng and Liou).

Next, the IPC code scheme is hierarchical. A prediction of an incorrect assignment in the same subclass as a correct assignment should be treated as less inaccurate than that of an outside alternative. This can be crudely addressed by adding a loss distance function that costs local code misassignment at less that of nonlocal misassignment.

Lastly, the study has shown that code assignments are far from uniform. An incorrect prediction of a popular codes should be treated as less inaccurate than that of a rare one. This is addressed by applying weights to each code’s predictions which vary inversely to each code’s popularity (King). The frequency distribution observed in section 3.1 serves this purpose.

### Model Training & Performance

With the word embeddings and loss function, a neural network model may be trained on observed assignments. Building the word embeddings and a large neural network is beyond the ability of a standard personal computer. As a result, the study used a memory intensive server whose definition can be found in section 7.2 of the Appendix. The model’s technical performance will depend heavily on the resources of the computing environment.

The study’s neural network is designed in layers to address the issues explained in the section **Loss Function** above. The input layer applies the word embedding. With 100,000 words each having a vector reflecting 300 latent dimensions, the input layer has 30 million nodes. The second and third layers apply autoencoding (Liou) reducing the many inputs to a latent space of 64 by 64 dimensions. The final layer outputs a vector of probabilities for each of the 3,098 IPC codes in the study. A summary of the model’s layer definition, with the number of nodes in each, is available in section **7.3** of the Appendix.

A neural network learns incrementally, passing the input numeric vector over layers of functions, incrementally increasing its accuracy defined by the loss function. In the field of neural networks, these iterations are called epochs. At the end of each training epoch, the model’s training accuracy is measured in terms of the loss function.

With each epoch, however, the model will also gain some bias particular to the training data. This overfitting is measured by testing the model against a portion of the study’s data withheld for validation testing. So, the model needs a limit of epochs that yields jointly the best training accuracy and best validation testing rate. **Figure 5** below provides the results of the model’s performance on these two measures. The large size of the dataset drives loss down aggressively. This model performs best at 4 training epochs.

Figure : Choosing an Epoch

Evaluating the performance of a multilabel classification model is complex. The model’s performance can be measured by two types of error and these can occur on the same prediction vector. For a given patent text, the model may suggest correct IPCs, incorrect IPCs and it can fail to suggest an IPC. Any prediction then can be correct, incorrect and partially correct. **Table 5** below provides examples of these two types of mistakes. In addition, the study may summarize these two types of errors by patent or by IPC code.

Table : Example Predictions

|  |  |  |  |
| --- | --- | --- | --- |
| Publication  Number | Observed  IPC Assignments | Predicted  IPC Assignments | Error |
| 9862061 | B23K35, B23K9 | B23K35, B23K9 | 2 Correct |
| 10076241 | G02C7, A61F2, A61B3 | G02C9, A61F3, A61B5 | 3 Incorrect, 3 Missing |
| 9954791 | H04L29, H04L12 | H04L29 | 1 Correct, 1 Missing |
| 9973561 | G06F15, H04L29 |  | 2 Missing |

The study’s model was trained on 90% of the study set, randomly selected, and validated on the remaining ten percent. A threshold prediction probability for each IPC was determined using the corresponding frequency of that IPC. **Table 6** below provides the categories of error and relevant proportions.

Table : Model Performance Summary

With this simple result, the study can continue to assess whether certain IPC codes are more predictable that others.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Count |  | Proportion |
| Assignments to Predict | 944,810 |  |  |
| Correct Predictions | 673,107 |  | 68.1% |
| Incorrect Predictions | 108,249 |  | 26.7% |
| Missed Assignments | 316,302 |  | 31.9% |

`

## Detecting Anomalous Assignments

The subject of anomaly detection involves concepts in addition to those discussed in sections **3.1** and **3.2** above:

**Unsupervised**. The previous neural network learns from the observed IPC assignments. An anomaly detection model would need a similar volume of observed exceptions. Collecting these observations would be infeasible, so a detection model would need to apply unsupervised algorithms.

**Distance**. The method proposed in this section compares the probability predictions from section **3.2** with binary vectors of actual assignments. The comparison of two probability distributions involves concepts of statistical distance.

**Continuous and Discrete**. In the proposed case, the expected probabilities will vary as real numbers between 0 and 1. The observed indicators however will be integers, either 0 or 1. The distance between two distributions, one continuous and the other discrete, can be addressed with methods of signal processing. These methods transform the discrete vector to a continuous one.

The model developed in section 3.2 above, given a patent text, will generates a vector of probabilities with a value for each of the candidate IPC codes. Just as this vector may predict or propose new IPC code assignments, it may be applied to evaluate observed assignments.

Practically, two anomalous conditions may exist. An observed IPC code assignment may be incorrect, or a proper assignment may be missing. Consider a vector representing predicted IPC code probabilities for a given patent’s text. Consider an alternative vector as a binary map of the actual assignments with value in positions where the corresponding IPC code is assigned and elsewhere. The value of will near zero when the model’s prediction agrees with the observed assignment. It will approach when a predicted assignment is missing, and it will approach for an incorrect assignment.

Figure : Anomaly Conditions

This approach depends critically on the performance of the model, however. The study’s model incorrect assignments above, for example, may include anomalous assignments. The accuracy of the model should be well established before it is consequently applied for this purpose. At that point, the errors may be prioritized using the absolute value of described above.

# Recommendations

The question presented in sections 1.1 and 1.2 above can now be addressed directly.

**1.2.1. Are IPC codes assigned evenly?**

Section 3.1 demonstrates that IPC code assignments are far from uniform. This is true collectively, within classes and for 559 of 631 subclasses.

**1.2.2. Can machine learning be applied to automate IPC code assignments?**

A basic neural network model was developed in section 3.2 with cross validation accuracy of 61.8%. This model is not comprehensive. However, it demonstrates that analytic methods can contribute to the classification process.

**1.2.3. Can anomalous IPC assignments be detected with analytical methods?**

Section 3.3 provides a method for using the predictive model from 3.2 to anomaly detection. The model’s vector of probabilities can be used to identify incorrect assignments as well as missing assignments. This should not be pursued until the model’s accuracy is well established.

**1.3.1. What distribution and parameters are appropriate to code frequency?**

A Pearson’s Chi Square test of uniform probability distribution is appropriate and may be applied family wide. It may also be applied with varying intervals within the levels of the IPC code hierarchy. When code assignments become more uniform, a Bonferroni adjustment should be applied.

**1.3.2 What methods are appropriate for learning against patent text?**

Section 3.2 includes discussion of four major word embedding methods: GloVe, word2vec, fastText and tf-idf. The model in section 3.3 uses pretrained word vectors of the GloVe method. In addition, issues and consequences related to model performance measurement are discussed in 3.2’s subsection entitled **Loss Function**.

# Resources

The preparation of this study relied on analytical, technical and functional sources in addition to those cited in the **References** section of this report, including:

* USPTO’s Bulk Data Storage System (USPTO)
* WIPO’s IPC Master files specification version 2.1 (WIPO)
* Stanford University’s GloVe algorithm (Pennington)
* Keras Deep Learning Library (Chollet)
* Natural Language Toolkit (Bird)
* TensorFlow (Google)
* Porter Stemming Algorithm (Porter)
* University of Pennsylvania’s English Gigaword (LDC)
* Princeton University’s WordNet Lexical Database for English (Fellbaum)

**Section 7.2** of the Appendix provides a detailed description of the computing environment including the server, programming languages, dependent packages and databases.

# Considerations

* The results in section 3.2 depend upon a threshold for IPC code prediction. In practice, this may be different for each IPC code. If the USPTO pursues automation with similar model, each IPC code should be associated with a different threshold acceptance probability.
* Addressing non uniform IPC code assignments will directly impact a predictive model and its use for anomaly detection. If the USPTO expects to see assignments improve in consistency, a subsequent model should be actively redesigned and retrained.
* Word embeddings demonstrated in this study have multiple natural language applications. The USPTO should consider developing an internal embedding data resource for reuse and continual improvement.
* The IPC code scheme is hierarchical. As a result, a misclassification within a level of the hierarchy could be considered less than a misclassification outside that level. This study addressed the issue with neural network design. The USPTO should assess alternatives including explicit distances among IPCs.
* This analysis was performed at the group level of IPC detail. The IPC scheme supports lower levels of assignment using the subgroup level. This lowest level supports nesting subgroups within subgroups and more than 70,000 details.
* The study’s model is exploratory and only intended to assess automation potential. A next step should include tuning algorithms and their associated optional parameters. In addition, a target operational state warrants planning the proper compute environments for training and real time inference.

# Appendix

## Supplemental Files

This report is complemented by files with the programming to support the production of its results. These files are available in a public version control repository for shared access as https://github.com/michaelbbryan/patentclass.

|  |  |
| --- | --- |
| **File** | **Description** |
| install.sh | A linux shell script, installing Python and requisite packages. |
| get\_red\_book.py | Downloads, uncompresses and converts the weekly publication files from XML to tabular form. |
| load\_red\_book.sql | Loads CSV files of Case Files dataset into a MySQL database for exploratory and relational analysis |
| subclass\_chisqr\_test.R | Programming and results of uniformity testing within each subclass |
| subclass.csv | Subclasses with assignment frequencies |
| classes.csv | Classes with assignment frequencies |
| multilabel\_with\_keras.py | Model with word embedding predictors and multilabel classification outputs |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## Computing Environment

|  |  |
| --- | --- |
| Programming | JetBrains PyCharm Community Edition 2019.1.2  Python programming language version 3.6.2  Python packages  glove 1.0.0  pymysql 0.9.3  pandas 0.24.2  keras 2.2.4  nltk 3.4.3  tensorflow 1.14.0  PyMySQL 0.9.3  RStudio Version 1.1.463  R version 3.5.3  git version 2.17.1 |
| Database | MariaDB relational database version 10.3 |
| Server | Amazon Web Services  Ubuntu Server 18.04 LTS (HVM)  r5a.4xlarge 16 CPUs 128GB memory |
|  |  |
|  |  |

## Model Summary

|  |
| --- |
|  |
|  |
| Layer (type) Output Shape Param # |
| ============================================================= |
| input\_1 (InputLayer) [(None, 200000)] 0 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| embedding (Embedding) (None, 200000, 100) 20000000 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| bidirectional (Bidirectional) (None, 200000, 256) 175872 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| conv1d (Conv1D) (None, 199998, 64) 49216 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| global\_average\_pooling1d (Globa (None, 64) 0 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| global\_max\_pooling1d (GlobalMax (None, 64) 0 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| concatenate (Concatenate) (None, 128) 0 |
|  |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| dense (Dense) (None, 3080) 397320 |
| ============================================================= |
| Total params: 20,622,408 |
| Trainable params: 622,408 |
| Non-trainable params: 20,000,000 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

## References

Bird, Steven Loper, Edward and Klein, Ewan. "Natural Language Toolkit." The NLTK Project, 2001.

Chollet, François. "Keras: The Python Deep Learning library." Massachusetts Institute of Technology, 27 March 2016.

Fellbaum, Christiane. "WordNet: An Electronic Lexical Database." Cambridge, MA: MIT Press, 1998.

Google. "TensorFlow." Google Brain Team, 9 November 2015. https://github.com/tensorflow/tensorflow.

IPC. *Nice Classification*. WIPO - World Intellectual Property Organization., 2017. http://www.wipo.int/classifications/nice/en/.

Jones, Karen Sparck. "IDF term weighting and IR research lessons." *Journal of Documentation* (2004): 521-523.

Joulin, Armand and Grave, Edouard and Bojanowski, Piotr and Mikolov, Tomas. *fastText*. Facebook's AI Research (FAIR). *Bag of Tricks for Efficient Text Classification*. 2016.

King, Zeng. "Logistic Regression in Rare Events Data." 2001. https://gking.harvard.edu/files/0s.pdf.

Kral, Ladislav Lenc and Pavel. "Word Embeddings for Multi-label Document Classification." *Proceedings of Recent Advances in Natural Language Processing*. Varna, Bulgaria, 2017. 431–437. https://www.acl-bg.org/proceedings/2017/RANLP%202017/pdf/RANLP057.pdf.

LDC, University of Pennsylvania. *English Gigaword Fifth Edition*. Philadelphia, 17 June 2011. https://catalog.ldc.upenn.edu/LDC2011T07.

Liou, Cheng-Yuan, et al. "Autoencoder for words." *Neurocomputing, 139* (2014): 84–96.

Mikolov, Chen, Corrado and Dean. *Efficient Estimation of Word Representations in Vector Space*. Cornell University. *arxiv.org/pdf/1301.3781.pdf*. 2013.

OED. "Oxford English Dictionary." Oxford University Press, 2019. www.oed.com/viewdictionaryentry/Entry/11125.

Pennington, Socher and Manning. "GloVe: Global Vectors for Word Representation." Stanford, CA: Stanford University, 2014.

Porter, vanRijsbergen and Robertson. "New models in probabilistic information retrieval." *British Library Research and Development Report, no. 5587*. Cambridge, 1980. Print.

ST36. "Handbook On Industrial Property Information And Documentation." *Standard ST.36*. World Intellectual Property Organization, November 2007. https://www.wipo.int/export/sites/www/standards/en/pdf/03-36-01.pdf.

Stanford, NLP Group. *Pre-trained Word Vectors*. n.d. https://nlp.stanford.edu/projects/glove/.

USPTO, Electronic Information Products Division. "U.S. Patent Grant Data." *XML v4.3 (a.k.a. Red Book)* . Alexandria, VA, 4 December 2012. PDF. <https://www.uspto.gov/sites/default/files/products/43AIACPCXML-Documentation-508version.pdf>.

Wehrmann, Cerri, Barros. "Hierarchical Multi-Label Classification Networks." *International Conference on Machine Learning*. 2018. 5225-5234. http://proceedings.mlr.press/v80/wehrmann18a/wehrmann18a.pdf.

WIPO, World Intellectual Property Organization. "Strasbourg Agreement Concerning the International Patent Classification." *International Patent Classification (IPC)*. 24 March 1971. <https://www.wipo.int/treaties/en/classification/strasbourg/>.