Patent Art Unit Classification Using Text Based Features

Joseph Baillargeon, William Ermlick, Nathan Roberson

Engineering Science and Mechanics, Virginia Tech
Email: jmb86@vt.edu, ermlickw@vt.edu, nathir2@vt.edu

A series of unclear decisions from the Federal Circuit Court of Appeals over the last thirty years has made the patentability of software patents difficult to determine. As a result, the United States Patent and Trade Mark Office's (USPTO) patent application rejection rate for software-related technologies has substantially increased in recent times. To avoid this higher rejection rate, it is beneficial if a patent applicant can file their application under a different technology center within the USPTO. This project developed a classification model to predict the assignment of patents to particular technology centers. Several classifiers were used in the test including a Perceptron model, a Passive Aggressive model, a Multinomial Bayesian model, and a Stochastic Gradient Descent model. Result from the classification indicates a 90% correct classification rate from the final model utilizing a Stochastic Gradient Decent method.

Keywords: Patent Classification, Technology Center, Stochastic Gradient Decent, Native Bayes Classifier, Term Frequency-Inverse Document Frequency

Received 03 May 2016

1. INTRODUCTION

Starting in the 1970's, the Federal Circuit began issuing case law decisions that promoted the patenting of software technologies [1].[1] These decisions quickly transformed software technology into a patent driven industry and caused software patents to proliferate throughout society. However recently, with cases like *Mayo Collaborative Servs. v. Prometheus Labs Inc.* and *Alice Corp. v. CLS Bank International*, new Federal Circuit decisions have restricted the free reign software patents once enjoyed [2]. The result is an unclear ruling on the patentability of software, which has led to issues during patent examination.

The United States Patent and Trademark Office (USPTO) is the governmental organization in charge of examining and allowing patents for new technologies. They have responded to these rulings by increasing the rejection rate for patents dealing with software related technology. In 2014, patents which the USPTO classified as software-related jumped from 24% of applications being rejected or abandon in January 2014 to 78% in June 2014 [3].[2] Thus to avoid rejection, it is advantageous for a drafting patent attorney to write a patent application so that it is classified outside of the software technology category.

When a patent arrives at the USPTO, an employee is in charge of classifying a patent to a particular Technology Center (TC) for examination. Typically, the patent's title, abstract, and the first independent claim text are exclusively used to place the patent application in the proper Technology Center. Software and business method technologies usually are placed in TC 3600, and other types inventions, such as computer or networking inventions, in centers such as TC 2100 or TC 2400, respectively.

This study focused on developing a model that predicts which of the three most related Technology Centers (2100, 2400, and 3600) with the largest rejection rate gap (3600 having the highest) a newly drafted patent application will be placed into for examination. The model will utilize text from a patent application's title, abstract, and first independent claim in order to classify it to a particular art unit. In this way, attorneys and others that draft patent applications can utilize the model to tailor the title, abstract, and first claim of their patents to control the Technology Center where the application is examined. Using this model will increase the chances of a patent application being granted and improve the effectiveness of the drafting attorney.

2. APPROACH

2.1. Data Mining Patent Information

Patent data for all granted and non-granted patent applications were recently made public by the USPTO

[1] See *Diamon v. Deihr*

[2] Alice was decided in March 2014, notably causing the large rejection rate change from January to June.

in October, 2015. The patent office made all this data available via batch downloaded zip files. These files had a non-standard format, where thousands of complete XML files, each containing all the information for a single patent, were combined into a large document file and then zipped to allow for public download. Figure 1 shows an example of one of these large documents. The first step in this project was separating the individual XML files from the larger documents and then parsing each XML file for relevant patent information.

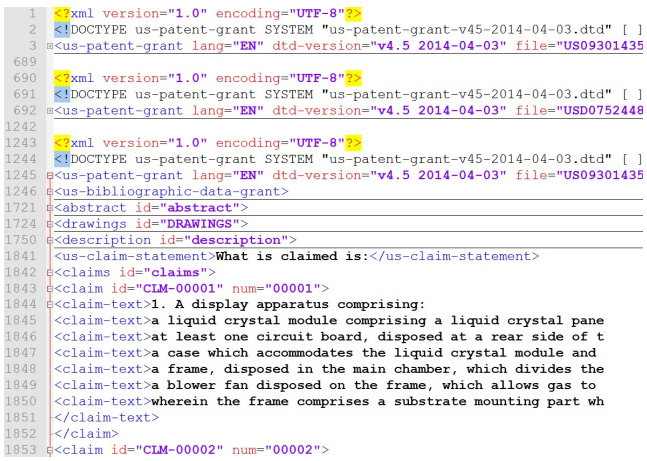

FIGURE 1: Zip file provided by USPTO. Multiple XML documents are enclosed in a single document. Claim text is shown in improperly formatted XML.

Zipped files were downloaded from the USPTO's website and stored locally for separation.[3] Individual XML documents were then identified from the full document contained in the zip file using a Regular Expression string parsing technique. The start of each XML file is signaled by a "<?" symbol. Text between these symbols were temporarily saved as an XML file. This working file could be used as a typical XML file which can be parsed with a variety of techniques.

A Minidom[4] package was used to obtain particular patent elements from the working XML file (e.g., grant date, title text, abstract text, first independent claim text, Technology Center, etc.). Nodes were identified within the working file's XML tree and their appropriate children were extracted. Conveniently, the information needed for extract is in the same location, with the same dependency names, for each XML file. This helpful tree structure aided in streamlining the data collection process.

Data collected from each file were saved in a dictionary format. This process was repeated for each XML file in the zipped file and then subsequently for all of the zip files for the 2015-2016 granted utility patent data. Extracted elements stored in the dictionary were manipulated using SQL commands made available via the Python package Dataset.[5] Finally, the dictionary was converted to a CSV file and saved for feature extraction.

TABLE 1: Patent Distribution by TC

Technology Center	Number of Patents
3600	40738
2400	37877
2100	30276
1600	15426
1700	34143
2600	48243
2800	84658
3700	46896

In total, over 200,000 granted utility patents were downloaded and used for training and testing. This study did not utilize non-granted published patent applications, since the data set does not include non-published applications. The complete granted data set introduces less variance and possible confounding effects, which made it a better candidate for study. Additional patent data may be collected to improve the results obtained here and perform similar types of analysis on patents based on this extraction process. A table of the extracted patent data for each technology center can be seen below in Table 1.

2.2. Feature Engineering

2.2.1. Feature Generation

Data from the CSV file went through multiple manipulations to clean the data before feature generation. Strings containing the patent title, abstract, and first independent claim were tokenized and stemmed to remove suffixes and prefixes and obtain solely root words. Additionally, punctuation and stop words that provided no semantic content (e.g., "a," "and," "the," etc.) were also removed from the strings.

The data frame was then converted into a feature matrix utilizing a Term Frequency-Inverse Document Frequency Technique ($tf - idf$). Every word in each abstract, title, and claim was weighted numerically according to the frequency of the term's appearance divided by the log of the number of patents that the word appeared in. A mathematical description of the process can be seen below in (Eq. 1), (Eq. 2), and (Eq. 3).

$$tf(t,d) = f_{t,d} \tag{1}$$

$$idf(t,D) = log\left(\frac{N}{|d \in D : r \in d|}\right) \tag{2}$$

$$tfidf(t,d,D) = tf(t,d) \cdot idf(t,D) \tag{3}$$

[3]Download available via: https://bulkdata.uspto.gov/

[4]See: https://docs.python.org/2/library/xml.dom.minidom.html

[5]See: https://dataset.readthedocs.io/en/latest/

Where $f_{t,d}$ is the number of times the word t appears in document d, N is the number documents in the corpus and $|d \in D : t \in d|$ is the number of documents where the word t appears. Additionally, the number of words considered could be changed in our model to include a few words or a phrase (denoted still as t) instead of only singular words.

As a result of this conversion, a sparse matrix was created with the numeric $tf - idf$ results for each word placed in each matrix element. Based on these values, a feature matrix could then be created and used to train different classifiers. In terms of dimensionality, the large number of features (words or phrases) created from the $tf - idf$ transformation were not a large concern. The sparsity of the matrix and the large data set allowed us to use phrases of up to three words for each of our features. This enabled the model to capture a wide variety of descriptions used in the patents, and ultimately better predict art unit classification.

2.2.2. Feature Selection

Feature Selection was used to reduce dimensionality of the feature matrix. The $tf - idf$ results created a large feature matrix which was primarily reduced by setting a max number of features in the model itself. With this, the number of features could be limited to less than the number of training samples to ensure the classification models would not run into mathematical errors. Using the vectorizer, the most significant features were selected, furthermore, any zero variance features were removed, and the K best features were down selected according to a chi squared scoring function. K was set by 90% of the number of original features.

2.3. Classifiers

At this stage, several classifiers were tested on the data in order to provide the best predictive model. Each classifier was employed based on it's suitability for text based feature analysis. A discussion is held here on each of the classifier's approaches and their different strengths. Namely, a MultiNomial Naive Bayes approach, and a Stochastic Gradient Decent approach, a Perceptron model and a Passive-Aggressive model were tested as classifiers [4].

2.3.1. Multinomial Naive Bayes

Multinomial Bayes method utilizes Bayes theory, seem below in (Eq. 4),

$$P(y|x_i, ..., x_n) = \frac{P(y)P(x_1, ...x_n|y)}{P(x_1, ..., x_n)} \tag{4}$$

and expands it to a multinomial distributed data where $\hat{\Theta}_{yi}$, the probability of feature i appearing in class y, is given by (Eq. 5).

$$\hat{\Theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n} \tag{5}$$

Where N_{yi} is the number of times feature i appears in the sample of the class y in the training set and N_{yi} is the total count of all the features for class y. α is denoted as the smoothing parameter which accounts for features not present in the training model. Naive Bayes classifier is very efficient due to it's low computational intensity in both CPU and memory usage and it requiring a small amount of training data. Moreover, the training time with Naive Bayes is significantly smaller as opposed to alternative methods that follow.

2.3.2. Gradient Decent

The fastest and most effective method we implemented utilized a Stochastic Gradient Descent (SGD) learning routine. The advantages of this method are its ease of implementation and the speed at which it trains on a large sparse data set. A disadvantage however, is that SGD is an iterative method and care must be taken to ensure that the parameters sufficiently converge. Additionally, this model is relatively sensitive to extreme values, however this was mitigated by the use of $tf-idf$ to scale the data between 0 and 1. The model parameters of Stochastic Gradient Descent method are found by minimizing the regularized training error seen in (Eq. 6).

$$E(w, b) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x)) + \alpha R(w) \tag{6}$$

Where R is the regularization term that penalizes complexity and L is the loss function. We optimized for a ratio of the L2 norm, $\frac{1}{2}\sum_{i=1}^{n}$ and the L1 norm as our regularization term since it yielded the best results. The loss function we used was soft margin Support Vector Machines (SVM). Both the $R(w)$ and the α parameter, which weights the complexity penalization factor, were optimized for an added test and training accuracy.

SGD is an iterative method that iterates over the training data updating the model parameter weights according to the following rule of (Eq. 7).

$$w_{i+1} = w_i - \eta\left(\alpha\frac{\partial R(w)}{\partial w} + \frac{\partial L(w^T x_i + b, y_i)}{\partial w}\right) \tag{7}$$

Where b is the intercept and is updated similarly and η is the "learning rate" which we used the default value of the Classifier method given by (Eq. 8).

$$\eta^t = \frac{1}{\alpha(t_0 + t)} \tag{8}$$

2.3.3. Perceptron

The Perceptron classifier has similar implementation as SGD, with a Perceptron loss function and a constant learning rate.

$$\hat{y} = argmax_y f(x, y) \cdot w \tag{9}$$

The Perceptron model iterates over the training data set and attempts to predict the output, altering the weighting of the predictor if a larger error is observed. The major disadvantage of the Perception model is that it is unable to converge if the data is not linearly separable by a hyper−plane; otherwise it is guaranteed to converge. It has an advantages in its ability to process large data sets as well as its interpretability, since it is a linear classifier. The Perceptron model is also slightly faster than SGD method due to only updating the weighting function,(Eq.11), when there are mistakes.

$$w_{t+1} = w_t + f(x, y) - f(x, \hat{y}) \tag{10}$$

2.3.4. Passive Aggressive

The Passive Aggressive method similar to the the Perceptron model, since it uses a constant learning rate, predicts the class, and compares it to the true class while suffering a loss according to its associated loss function. However, it does use a regularization parameter for scaling. Similar to the Perceptron model, when the algorithm suffers a loss the discriminate function is updated with new weights via the optimization of the following equation:

$$w_{t+1} = argmin_w = \frac{1}{2} \sum_{u=1}^{K} ||w_u - w_u^{(t)}||^2 \tag{11}$$

Where w_{t+1} is the nearest point in space based on the weight vectors that would suffer zero loss against the current training data. The advantages and disadvantages of the Passive Aggressive model are very similar to those of the Perceptron model.

3. RESULTS AND DISCUSSION

3.1. Cross Validation & Parameter Selection

Once a handful of of classifier methods were selected that were most likely to yield the best results in predicting patent classification, each method was trained using the training data. A cross validation of 10 fold was used to minimize our training error and, therefore, the test error. Alpha and L1 parameters were varied for the SGD in order to obtain the optimal test results. Cross validation was used for this process. Grid searches were also coupled with the cross validation to determine the optimal values. In addition to the tuning parameters, the number of words per element of the feature matrix was also varied. It was found that a maximum of three words per element provided the best test results.

3.2. Classifier Comparison

Each of the different classifiers discussed above were trained on a 10 fold of the full data set of 200,000 granted utility patents from 2015. The classifiers made use of the title, abstract, and first independent claim of each patent and it's associated technology center where it was examined. A prediction model was created for each classifier that encompassed all of this data. As a benchmark, and due to the demand, a model was only created for comparing between Technology Centers 2100, 2400, and 3600. Further work may be completed on a model for all eight TCs, which will be discussed later.

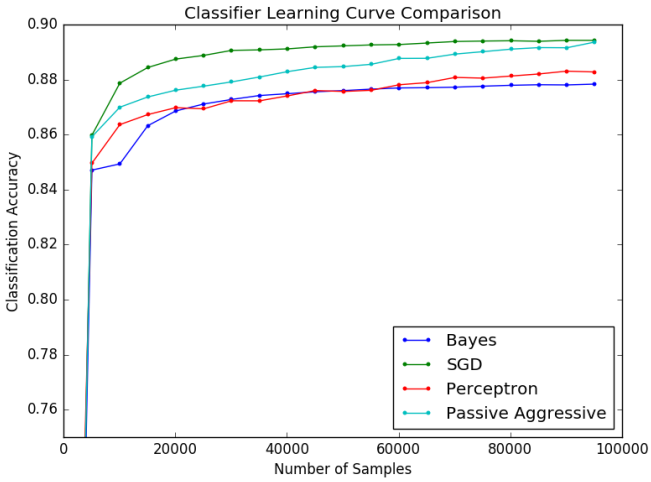

FIGURE 2: Learning curve asymptotic convergence of the classifier accuracy as the number of samples increases.

TABLE 2: Patent Distribution by TC

Classifier	Validation Error
Stochastic Gradient Descent	89.4%
Passive Aggressive	89.3%
Multinomial Naiive Bayes	87.8%
Perceptron	88.3%

A plot of the overall learning curves for each classifier can be seen in Figure 2. As shown, Stochastic Gradient Decent demonstrated the most effective ability to predict the correct technology center classification with an overall prediction percentage of around 90 percent. The Passive Aggressive model was the next most effective classifier, with the Perceptron and Mulitinomial Bayes as the two least adept models, in order respectively. A table of the associated errors for one of the cross validation tests is shown in Table 2.

Table 3 shows the confusion matrix for the SGD method with the best tuning parameters for alpha and L1. It can be seen that around 10% of the data is misclassified, with the majority of the misclassifications placing a patent application in 2100 when it is truly in 2400. This helps avoid placing any patent application in 2100 or 2400 when it should be classified as 3600, Since 3600 has the largest patent application rejection rate.

TABLE 3: Confusion Matrix

	TC 2100	TC 2400	TC 3600
TC 2100	29448	704	124
TC 2400	561	37200	116
TC 3600	172	362	40204

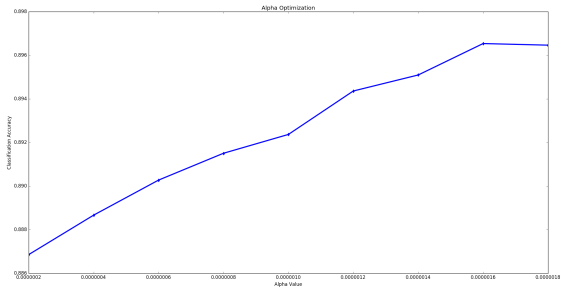

FIGURE 3: Grid search of the alpha complexity weighting parameter to find optimal classifier accuracy.

When initially tested, Stochastic Gradient Descent had the best performance in cross validation tests. From here, it was important to optimize this model to achieve an even higher classification accuracy. Due to long computational times, it was not possible to optimize every model. Only the SGD model was given a large parameter space to optimize over. The most important inputs to this model were the alpha, and L1 values. Alpha was varied from 10^-7 to 10^-5 while L1 factor was varied from 0.1 to 0.9. A plot of the convergence of the alpha parameter is shown in Figure3.

A grid search was conducted to find the optimal L1 ratio in addition to the cross validation. A plot of the L1 ratio search can be seen in Figure 4. The optimized SGD classifier found had an alpha value of $1.3 * 10^{-5}$ and an L1−ratio of 0.1.

A plot of a single learning curve with only the SGD model is shown in Figure 5. This shows how the training error increases as the optimal cross validation model error decreases. The optimal value for the testing data set results in a lower training set result, which is expected for most learning models.

3.3. Heuristic Determinations

An attempt was made to train a tree model, however it ultimately failed due to the complexity of the data set. Determining the set of words that would help classify a patent to a particular technology center would vastly aid the drafting attorney. This will be discussed in more detail in the future work section.

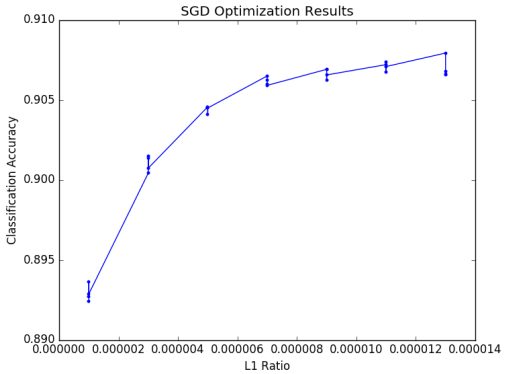

FIGURE 4: Grid search of the l1 ratio normalization function to find optimal classifier accuracy.

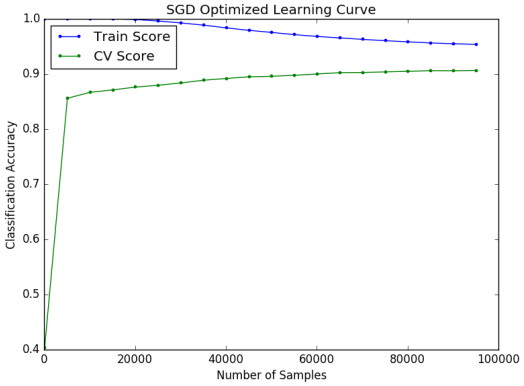

FIGURE 5: Shows an asymptotic convergence of the optimized SGD classifier accuracy as the number of samples increases.

3.4. Web Application

A web application was developed to enable users to take advantage of the predictive model. The user, which may be a patent examiner, an inventor, an attorney, or the like, is able to input the title, abstract, and first claim of a patent application and receive prediction information. The application parses each individual component within the respective vector model, combines the feature vectors, and uses the optimized classifier to predict the art unit. This application enables drafters, examiners, and inventors to determine where a patent application will be placed for examination and puts our predictive model to direct, applicable use. Figure 6 shows the web application prompt currently published locally in our code repository on git hub.[6]

3.5. Future Improvements

More data may be used to improve our model for better prediction. Patent applications, which have yet to be granted, may also be used to train the classifiers. This would yield more current information for how workers

[6] https://github.com/ermlickw/PatentAnalysis.git

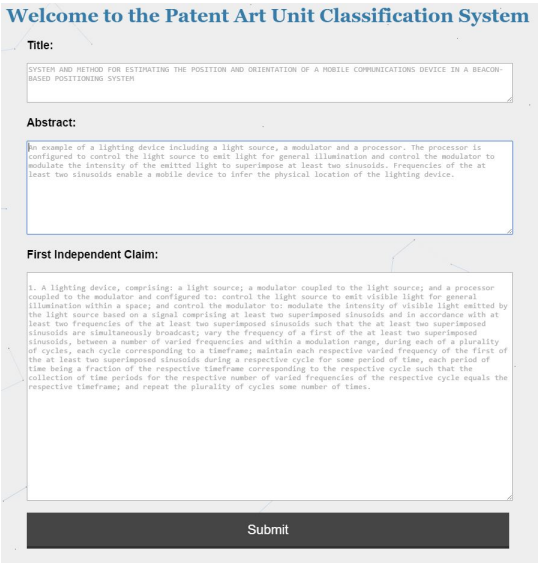

FIGURE 6: App created to allow someone to run the classifier algorithm on.

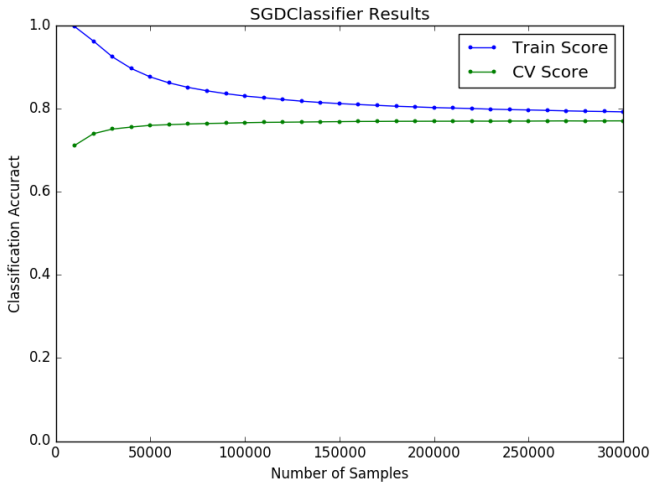

FIGURE 7: Learning curve for all 8 technology center classifier.

at the USPTO are classifying patents since these applications are more recent. Tuning of parameters may also be improved through various methods.

Another feature that may be interesting for classifying is determining the type of independent claim that is listed first in the patent application. For example, a patent application may list first a device claim, a method claim, a computer readable medium claim, or the like. Which of these claim types are listed first may play an important role in determining technology center classification and may be an interesting aspect to consider for future models.

Currently, the best predictive model only compares three technology centers (the three that are most similar and have the highest rejection rate differences). However, creating a model that would predict for all eight technology centers would help increase the robust predictive power of our model. From the most recent results, the model is only able to classify to the correct TC around %80 of the time when all eight TCs are incorporated. A plot of the learning curve is plotted in Figure 7. Future work may be done to improve the modeling for predicting over all the technology centers.

The addition of heuristic tools to help a drafting attorney predict what phrases and language to exclude from their patent would also be beneficial to include in our model. These features may be extracted based on a weighting function used in several classifiers or counting the appearances of certain words using the $tf - idf$. A tree model may also help to find the most significant words to avoid while drafting.

4. CONCLUSIONS

This paper considered the prediction patent application assignment to particular technology centers within the United State Patent and Trademark Office utilizing classification learning techniques. Several classification techniques were used in order to create the final predictive model. Results from the testing indicate a maximum of 90% correct classification rate with the best predictive model for three commonly confused technology centers, namely 2100, 2400, and 3600. This error is also supported by the fact that a human is used to classify the patents at the USPTO. This may help account for some of the error seen.

In addition to the model, an application was created for the prediction of new application assignment. This application allows attorneys to enter in the title, abstract, and first independent claim for their application and receive the predicted technology center assignment. Utilizing this feature, attorneys may edit the language in their patent applications in order to avoid the assignment of their application to a particular technology center. In this way, an drafting attorney reduce the potential rejection rate for their application by avoiding art units such as 3600, which implicitly have a higher patent rejection rate due to the current court legislature.

In the future, work may be done to improve the accuracy of the model. Additional data may be collected to improve the training set size. Features such as all of the claim text, and the specification text may be used to further improve predictive power. Also, optimization of various parameters of each of the classifiers may be conducted to heighten the predictive capabilities of the model. Studies on other areas of patent law utilizing predictive power may be interesting for attorneys, inventors, and those affiliated with patent law. Techniques from this study may be applied to solve these types of problems.

ACKNOWLEDGEMENTS

This research was conducted in collaboration with

Harrity & Harrity LLP. Conception of the project aim as well as interpretation and analysis of the results were made possible as a result of this collaboration.[5]

REFERENCES

[1] D. Stein, "The main event: Alice v. diehr," *USPTO Talks*, 2014.

[2] M. Goetz, "Why alice v. cls bank is a victory for software patents," 2015.

[3] M. Masnick, "Good news: Us patent office now rejecting a lot more software patents," 2014.

[4] F. Pedregosa, "Feature classification using sklearn," 2010.

[5] P. H. J. O. Rocky Berndsen, John Harrity *Harrity & Harrity LLP*, 2016.