Predicting Bad Patents

Joong Hwa Lee

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2017-57
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2017/EECS-2017-57.html

May 11, 2017

Predicting Bad Patents

Master of Engineering Final Report

In collaboration with William Ho, Dany Srage, Tzuo Shuin Yew, and David Winer

University of California, Berkeley

Electrical Engineering and Computer Science

Written by Joong Hwa Lee

Faculty Advisor: Lee Fleming

Faculty Committee Member: Vladamir Stojanovic

May 2017

Executive Summary

The recent rapid development of technology has led to more patent submissions for a variety of inventions. This leads to more similar patents that are likely to be invalidated. In fact, more than 20 billion dollars have been spent in the last 2 years in the patent litigation cases. A tool that could check if there is a similar patent before submitting a potential patent would prevent many of these resources being wasted. Our team has developed an algorithm that can predict how likely it is for a potential patent to be accepted by the Patent Trials and Appeals Board and if it is accepted, how likely it is to be invalidated by the United States Patent and Trademark Office.

The basic steps in developing our project is as follows: data collection from the USPTO, building and tuning the machine learning algorithm, and hosting the algorithm on a graphical user interface. Chapter 1 of this paper is the technical contribution that briefly summarizes how each member of the team contributes to the project and discuss in-depth of my own contribution to the project. Specifically, I discuss the web scraper that uses Beautifulsoup to extract data from the USPTO website with an additional features for error prevention, and different algorithms like the simplistic bag of words algorithm that vectorizes each patent by the word frequency to determine the similarity and the n-gram vectorizer that vectorizes each word of the patent by observing 2 words that precede and follows every word to better understand the context of the paper. I also discuss the graphical user interface that is built with the patent prediction algorithm and designed with trending design principles. In Chapter 2 of the paper, we explore the industry context and competitors in our business landscape. The current machine learning trends and the ethical considerations regarding the project are explored.

Chapter 1: Technical Contribution

Introduction

With more than 20 billion dollars spent in the patent litigation in the past 2 years, a tool that can predict how likely that it would be invalidated in the future would serve to be a valuable tool in the patent industry. Our project aims to use machine learning techniques to train a model that can predict bad patents to avoid unnecessary time and resources spent in patent invalidation and also provide descriptive statistics about patents that could be useful in patent analytics. Such tool would be used by patent authors and aspiring inventors to check if there is already a patent that is similar to their patents, so that their effort is not spent on inventing and patenting an idea that has already been patented. This not only saves a ton of resources of the United States Patent and Trademark Office, but also the time and resources of many patent authors. Many litigations and legal battles would be avoided with a simple check with our tool. We have modularized the process of building such tool to the following steps. First, we must download and scrape data from the United States Patent and Trademark Office website which holds millions of metadata about the patents and the patent file PDFs. We also gather data from the Patent Trials and Appeals Board which holds patent trial information including patent invalidation texts. Then, we collect the gathered from different sources and insert it to our database. This database is used to train our machine learning model using different techniques. We must determine which machine learning model works best for patent invalidation prediction and further tune the model for performance optimization. After we complete the development of the machine learning model, we install this into our Django framework where we have our website hosted and develop a good front end design to accentuate the features of the website.

Tzuo Shuin Yew will discuss the setup and insertion of data into our database in detail as well as the graphical user interface in Django environment that we are using. William Ho will explain the discovery of data sources, the implementation and optimization of data acquisition, parsing results from PDF texts, and high-level date-based descriptive statistics. David Winer will explain data wrangling and featurization of denials algorithm, linear SVC and random forest approaches to predicting denials. He will also explain how we use cross-validation and tune the model and the method of evaluation. Dany will discuss how he parses US Code from the files, generates descriptive stats, downloads patent related data, and the invalidation algorithm. In this paper, I will discuss downloading and extract data from the USPTO, vectorizing and running a bag of words algorithm, running n-gram algorithm with gensim model, and the front end design of the website. The entire process of this project is further shown in the work breakdown structure below in Table 1. The PTAB case process is described in Figure 1.

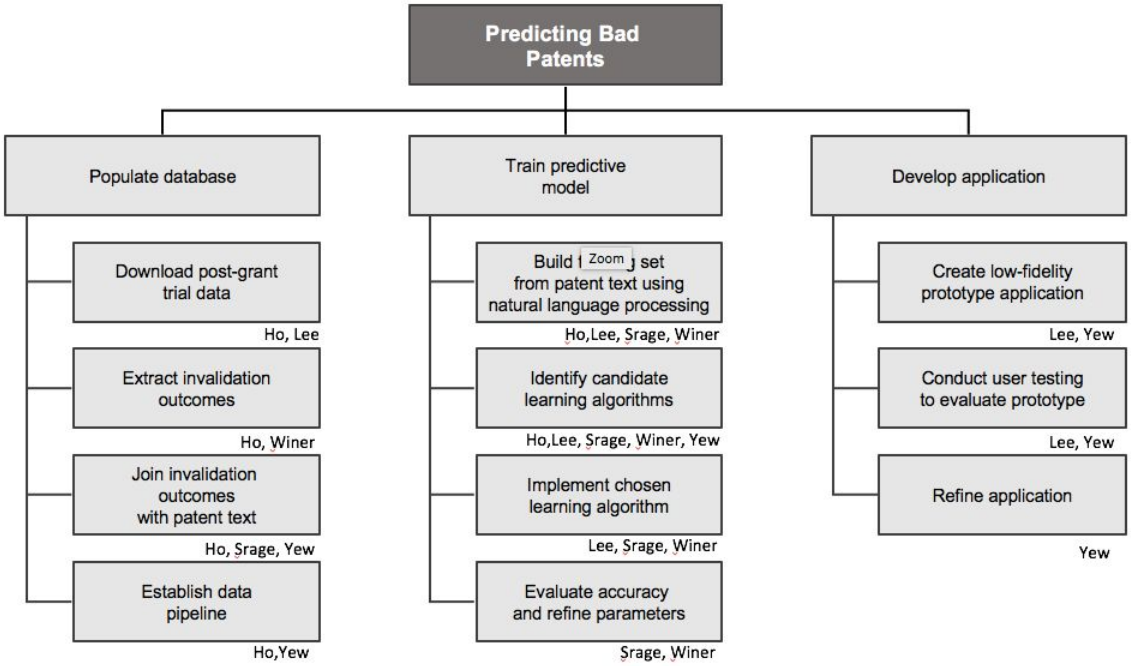

Table 1: Work breakdown structure that explains the process of this year-long project

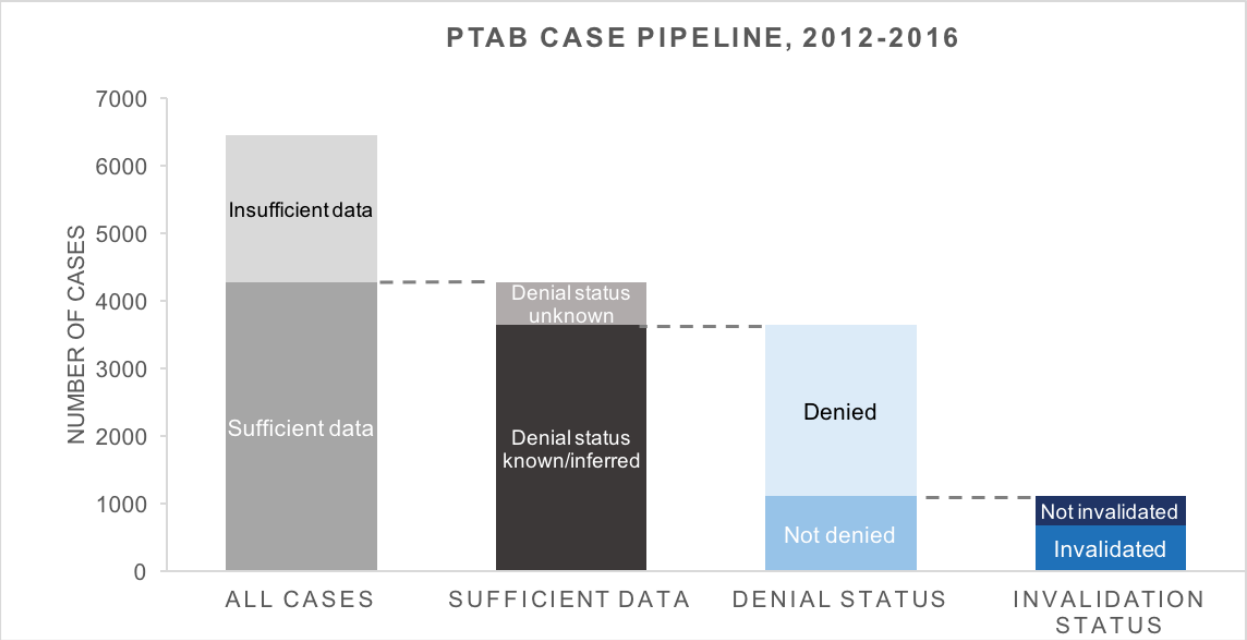

Figure 1. The graph shows the number of cases are in each steps of PTAB case process. Only a small set of cases are given invalidated/not invalidated results.

1. Data collection - Web scraping

In order to build a good machine learning model, we must have numerous data that represents the entire universe of patents. We collect our data from a various of sources, one of which is from United States Patent and Trademark Office. The information regarding final decisions of patent trials were collected from July of 1997 to current date, March of 2017. There are 107,734 items in total, which are organized into a tabular format on the United States Patent and Trademark Office website. The table contains the following information:

- Application number
- Appeal number

- Interference number
- Publication number
- Publication date
- Patent number
- Issue date
- Decision date
- Inventor name
- Case number
- Application document
- Status of the application which can be one of:
    - Decision
    - Judgement
    - Decision on motions
    - Order
    - Rehearing on decision
    - Subsequent decision.

The data is organized in a tabular format. We use a Python library called BeautifulSoup that utilizes Python parsers and creates a parsing tree to facilitate data extraction from html pages (Richardson, 1996). We programmatically locate the table in the html page and download the entire table per page for all of the downloaded html files as csv(comma separated values) files.

There is a number of issues that we must consider with the web scraping process. First, the web server in Patent Appeals and Trials Board can only handle so many requests. This server

was never designed to handle web scraping, so it returns an error page when we attempt to download a lot of pdfs in a short amount of time. To resolve this, we put a 2 second delay between downloading the pdfs which significantly lengthens the web scraping time. We had initially planned to write code to download the tabular data directly from the website one by one. However, with the delay, it takes more than 2 minutes per page to download all of the PDF's and the rest of the table with more than 1700 pages. This mounts up to about 3400 minutes which is about 57 hours. It takes a significant time to download the entire database and even if we dedicate one of our machines for more than 2 days, we cannot assume that the internet connection would be stable for 57 hours on our device. We modified our program to be able to run multiple times in case it crashes by checking in the local folder which item it downloaded last as a mechanism to recover easily from unexpected errors. Also, the Patent Appeal and Trials Board updates all of the data on the same table, so entries were being updated daily. Duplicates would cause certain patent invalidation cases more weight than others and joining the patent invalidation outcomes and the patent claims text to create a database would be problematic. To avoid duplicates, our program first downloads the html pages from 1726 pages so that we have a hard copy of all tables without it being updated daily (ScrapeHero, 2014).

This process is iterated until we reach the end of the table. the downloads are complete, our program converts the pdfs to text files using optimal character recognition, known as OCR. For the scope of our project, we are only interested in post-grant patents after 2012. For files that do not fall under this category, the program skips the conversion process. The decisions are

parsed out from these documents and inserted into our database. This process is further elaborated by William Ho in his paper.

With our scraper that has error-prevention and error-recovery features, we can reuse this tool to scrape the dat from the United States Patent and Trademark Office data consistently without having to

2. Patent vectorization and word frequency algorithm

There are millions of patents that are of various lengths and types and we need a way to vectorize the patents in order to be fed into our machine learning algorithm. We explored different machine learning algorithms and patent vectorization methods. Our team decided to diverge into different paths of machine learning algorithm and use the most effective algorithm. One of the first efforts at machine learning algorithm is bag of words algorithm. The bag of words algorithm vectorizes patents' claims text by observing term frequency of each word in its own dimension. Every patent is projected to a n-dimensional space and our intuition was that invalidated patents will have a closer average distance to other patents in the dimension. To do this, we first parse out the claims text from the Fung database that holds millions of patents. We have 777 patents that invalidated and 877 patents that are not invalidated. Of these 1654 patents we sampled about 34 patents, 16 invalidated patents and 18 not invalidated patents, to gauge if this algorithm is suitable for patent prediction. From the test set, we parsed out the claims text and the patent numbers. Then, we preprocessed the data to remove words that have very little lexical content like "and", "the", "itself", and etc. as these words appear frequently in sentences. These words were removed using a natural language toolkit that is installed in python.

Projecting every patent to a n-dimensional space would amount to be a very expensive operation. With millions of patents the words that these patents cover would amount to a very large number. Assuming that the entire universe of patents would cover just 50 percent of English language, this means that millions of patents would be projected to a 80,000 dimensional space with most dimensions having a value of 0. To avoid this extraordinary amount of computation, we only compare distances between two patents by projecting to a space that only has the words that are covered by two patents. Even though this method would require computing n-dimensional space for every pair of patents, it is much more efficient and faster than the original implementation. Also, this doesn't require us to rebuild the model whenever new patents are added because we reconstruct the n-dimensional space for each pair of patents.

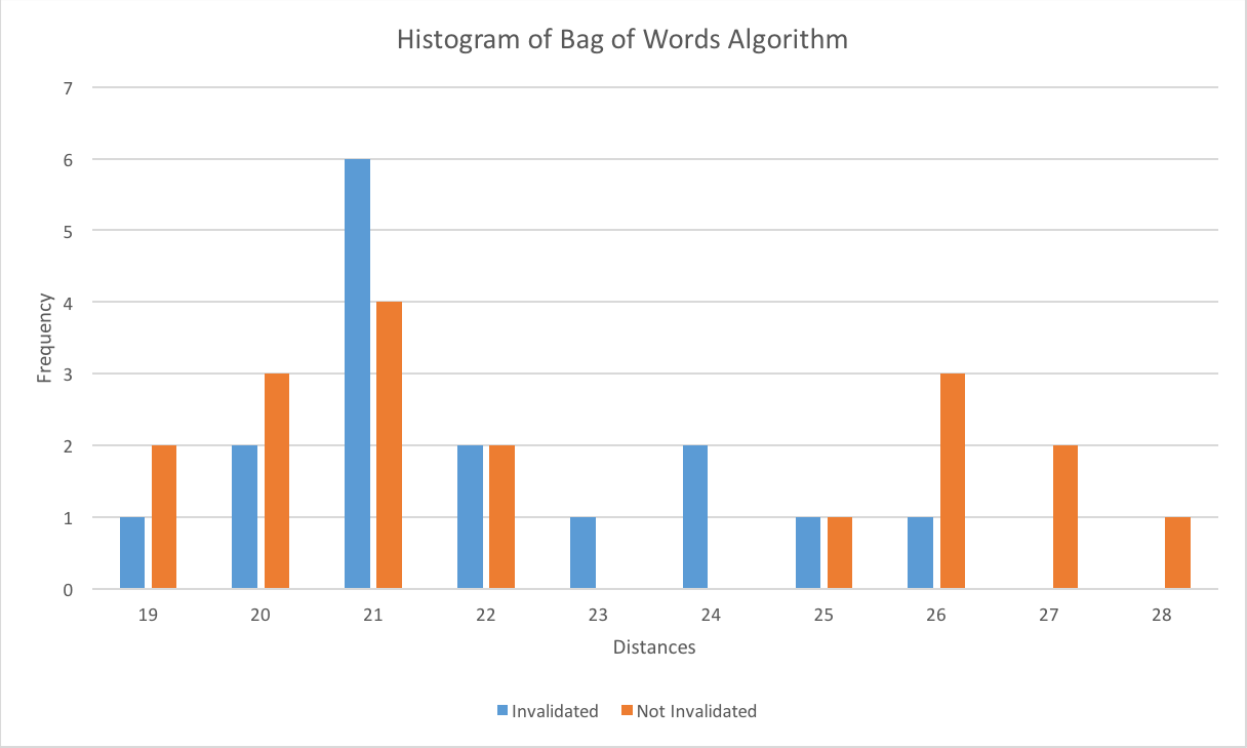

Figure 2. The bag of words distances between 34 patent documents with a decision and 77,000 patents is shown above. This shows that the invalidated patents have a slightly closer

distance than the not invalidated patents, but the gap is not large enough for us to make any conclusions.

	Average	Standard Deviation
Invalidated Patents	22.47	9.21
Not Invalidated Patents	23.28	10.89

Table 2. The average and standard deviation of distances between invalidated and not invalidated patents.

With 32 patents with a decision from the invalidation process, we sample 70,000 patents from the entire patent database to compare the distance. The result is as follows: for invalidated patents, the average distance between the patent and every other patent in the sample is 22.47 with a standard deviation of 9.21 while for not invalidated patents, the average distance is 23.28 with a standard deviation of 10.89 as shown in Table 1. We hoped to see a big enough gap between the average distances between invalidated and not invalidated, but the mean distances are too close and the standard deviations are far greater than the difference between the two means. We tested a variation of the algorithm which only calculates distance between a given patent and 10 nearest patents around it. The logic behind this implementation is because not all patents are similar or relevant to that patent and the most similar patents are the ones that are most likely to invalidate that patent. We ran this variation of the algorithm and the result is as follows: the average distance of invalidated patents is 16.03 with a standard deviation of 5.2 and the average distance of not invalidated patents is 15.28 with a standard deviation of 5.9. Even with the variation, we cannot draw any conclusions as the gap between the mean distances are

too small. We determine that the bag of words algorithm might be too simplistic for predicting bad patents.

3. N-gram vectorizer

When humans read sentences of any language, they do not just read word individually without thinking about context. The previous effort of considering the term frequency of each word in its own dimension does not capture the contextual information of each word. For example, The word 'mine' has two different meanings in the sentences "The pen is mine" and "The mine exploded". Humans have no problem determining that the word 'mine' is used to mean that the pen belongs to the speaker in the first sentence and that it is used to describe an explosive device that is buried beneath some surface in the second sentence. This is because humans understand the contextual information in the sentence. To do help machine understand some context to each word, we use n-gram vector model. N-gram model observes n previous and following words of each word in the document. N-gram vector model is similar to the bag of words algorithm in that each document is vectorized to a n-dimensional space; the number of dimension is the number of n-grams in the document. $w_k = (ln(f_k) + 1)/NORM$ where NORM is the normalization factor to make the length of each vector equal to one and f_k is the occurrences of k-th N-gram. Each document was preprocessed to remove any words without lexical content like the bag of words algorithm. We used a similarity function that computes the cosine similarity between two vectors that return +1 for most similar and -1 for least similar. This model requires the all the documents to have the same dimension and in the same space. We were able to reduce the computation space and time by doing a pairwise comparison with tens of

thousands of patents; however, we need to feed in the vectors into the doc2vec model that vectorizes and maps the documents which takes a significant amount of time and computation. We've vectorized about 42,000 patents which takes about 40 hours. We compared the average similarity with every other document in the 42,000 patents to observe if there is a big gap between not invalidated patents and invalidated patents.

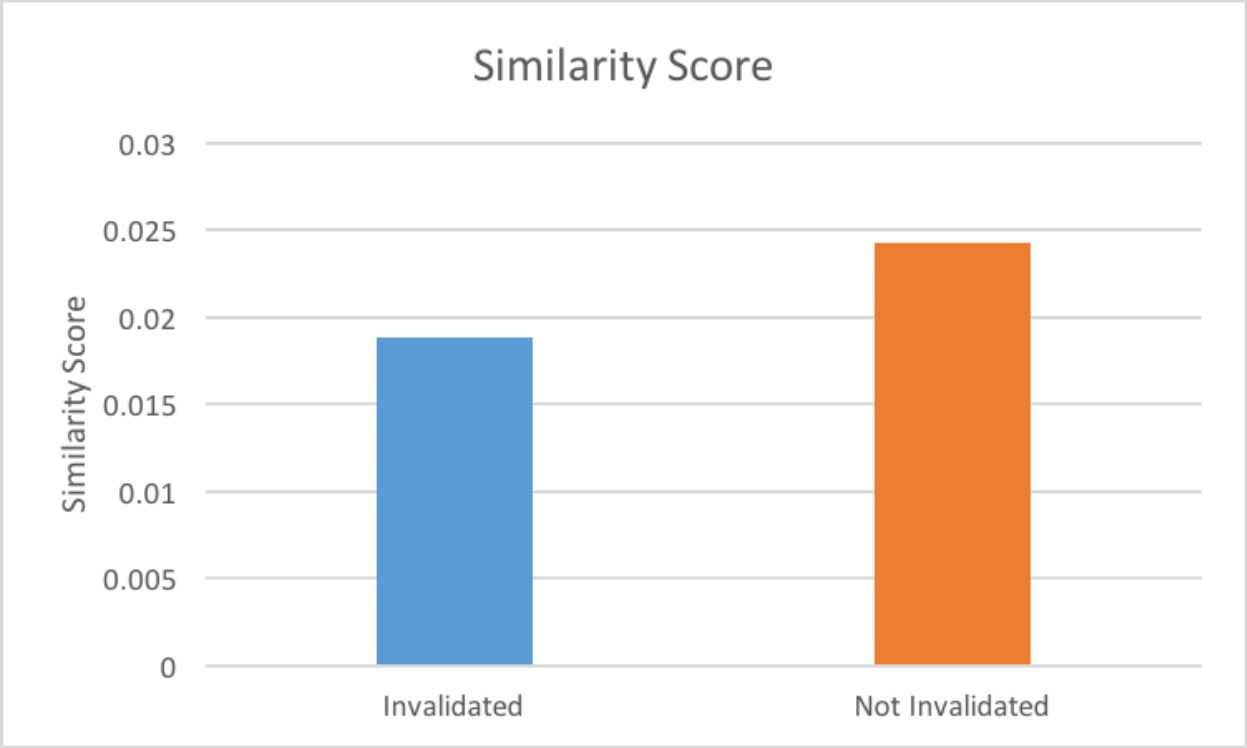

Figure 3. N-gram vectorizer result shows that the similarity score of the not invalidated patents is higher than that of the invalidated patents which is the opposite of what we expected.

The result is shown in Figure 3.The average similarity score of not invalidated patents is 0.018787 while not invalidated patents have an average similarity of 0.02427. We expected invalidated patents to have a higher similarity than the not invalidated patents, but the result

actually shows that not invalidated patents are similar. This might be due to the lack of enough patents with invalidation decisions.

A drawback of this algorithm is that there are 1,600 patents with invalidation results which are too few to get any substantial result from the doc2vec model. This is one of the major problems that we have had. In order to have a precise, accurate machine learning algorithm with a good predictive power, there has to be a large enough training and test data. However, only having 1,600 samples is not enough for us to get any predictive power. We have considered using an unsupervised learning which doesn't require patents to be labeled. This would allow us to expand our training set to the order of millions instead of thousands. Some of the popular unsupervised learning is k-means, clustering, but we concluded that finding patent documents' similarity might be too sophisticated to use unsupervised learning algorithms that are available today. Another disadvantage of this method is that it takes a very long time to build the model. For every new patent sample that is added to the model, we would have to rebuild the model which takes tens of hours. We could re-train the model for every 100 or so patents that are added, but this would decrease the validity of the model as newly added patents would not be taken into consideration when outputting the probability of invalidation. We determine that the n-gram vectorizer is not appropriate and perhaps too simple for a complex task like predicting bad patents.

4. Website Design

Although our solution will consist mostly of back-end programming with machine learning, front-end development of the project is crucial as well. According to Myers and

Rosson, much of software development efforts and time are spent in front end design: “In today’s applications, an average of 48% of the code is devoted to the user interface portion”(1992). The current software industry’s key factors in competition are “usability, ubiquity/compatibility, and external competition”(Blau, 2016a, Competitive Landscape), and this trend is increasing.

Though popular design principles relevant today have been published for years, a lot of software applications from reputable companies have obvious design flaws that violate crucial design principles. Even when these flaws are small errors that only occur 5 percent of the time, usability delays add up quickly; in a program with 8000 daily users, if 400 people experience 4 two-minute delays a day, more than 53 hours are wasted a day. Most usability issues stem from designers not understanding users well enough. Much of our time will be dedicated to analyze our users’ needs and goals to optimize usability for patent applicants.

To design a good user interface, it is important to focus on the target users that are going to utilize our project. Understanding the needs of users to have a large user base is crucial in entering the software industry(Blau, 2016b,Industry Performance). Current front end industry utilizes task analysis and contextual inquiry to analyze and understand their users. Task analysis is a method of interviewing potential users a series of questions to learn about how they perform tasks, what tools they use, and what challenges they face. The goal is to understand users, the tasks they perform, and potentially predict new tasks the users might perform. For our project, the potential users are patent lawyers, patent applicants, potential inventors, corporate lawyers, etc. These users have at least some understanding of the patent system and legal jargon related to the patent litigation process.

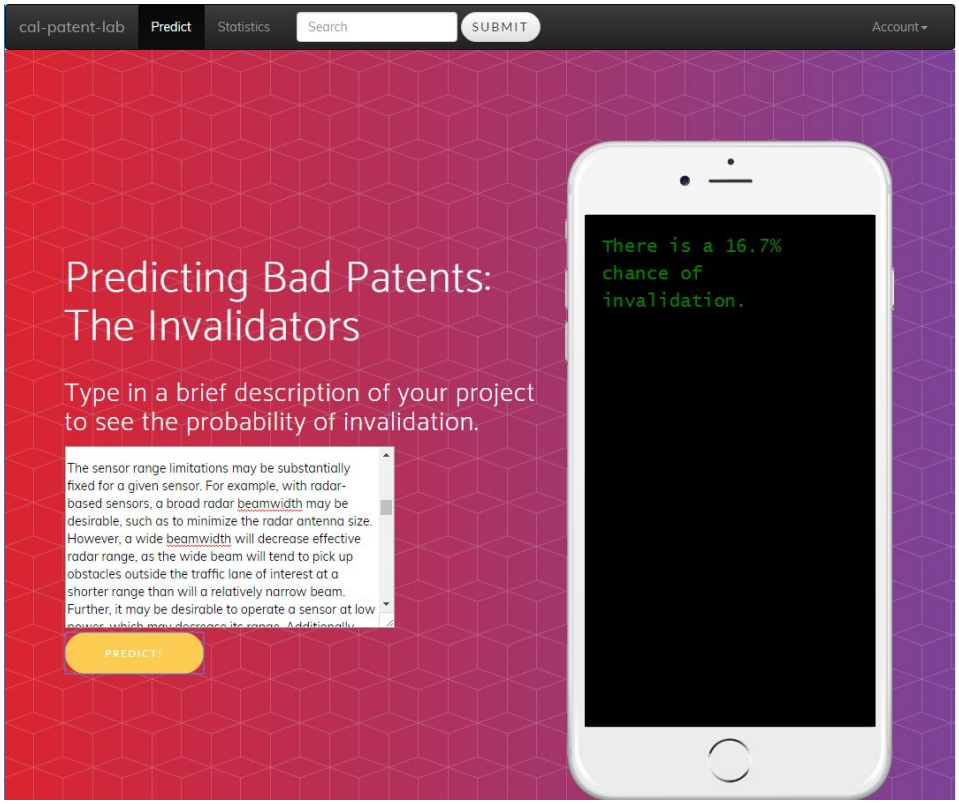

Figure 4. Django website with patent prediction program installed

Our website is designed behind a django framework. The structure and the framework of the website is elaborated more on Tsuo Shin Yew's technical contribution paper. . For the initial stages of the website launch, we wanted to target the users to be potential investors as our algorithm is not yet as reliable for it to be used by patent lawyers and professionals. We wanted our website to follow the minimalistic design which is the trend of the front end design today. The website has an input window where people can paste their patent abstract. Then, below the text box, when a user clicks on 'predict' button, the code runs the text through the algorithm to output the prediction. We will have the machine learning model installed in the backend of the django framework. The website will display the probability of invalidation with ranges between 0~ 20 percent displayed in green, 20~ 60 percent in yellow, and 60~ 100 percent in red. Even

though the website may seem simplistic in design, we believe that this rather simplistic design helps highlight the main objective of the project which is to predict bad patents. Without a ton of buttons and texts everywhere, it makes it clear to users what this website does and further makes it visually aesthetic. Simplistic designs also contributes to the speed because it loads faster and is less complex to manage. Since our main target users for the beta stage of the website are the potential inventors, we want to get rid of any inconvenience and make the experience as smooth and user-friendly as possible.

Conclusion

In today's technological trend, the number of potential inventions and patents are only expected to increase every year. The descriptive statistics regarding patents provides insight to the patent application process that would help patent lawyers to foresee how a patent would do throughout the legal process.We hope that our machine learning algorithm will be used as a tool by professionals in the patent industry and aspiring investors to save billions of dollars that are spent on patent invalidation and a tremendous amount of time spent on the legal proceedings.

Chapter 2: Engineering Leadership

1 Introduction and Overview of Technology

As technological innovation has accelerated over the last two decades, the number of patents has also grown substantially. 300,000 patents were filed with the US Patent and Trademark Office (USPTO) in 2014, which shattered the previous record for the number of patents in a year (U.S. Patent Statistics Chart, 2016). This growth in applications, however, also led to more resources being spent on patent litigation; for example, more than $20 billion was spent on patent litigation in the last two years. We aim to build a machine learning tool for patent applicants and examiners to check the likelihood that their patents will be invalidated before even filing them. By providing this information before patents are approved, we hope to limit the cost that bad patents impose on the legal system and the economy—estimated to be up to $26 billion annually (Beard, p. 268). In this paper, we will examine the industry context and business considerations associated with building this machine learning tool. First, we will discuss the current patent landscape and how it informs the marketing strategy for potential customers. Second, we will analyze the different competitors in the legal services space and define how our tool differs from existing offerings. Third, we will discuss the current state and trends in the machine learning field today and how they can be applied to our tool. Fourth and finally, we will close with an ethics section that will examine the ethical issues that we must consider in building the product.

2 Marketing Strategy

It is becoming both increasingly challenging for research-oriented firms and their attorneys to navigate the intellectual property landscape in the United States. In addition to the increase in the sheer number of patents, recent changes in US law have made it significantly easier for members of the public to challenge existing patents. In 2012, the US federal government enacted the Leahy-Smith America Invents Act (AIA). This legislation substantially expanded the role of the US Patent and Trademark Office in post-grant patent reexamination (Love, 2014, Background). The AIA opened the gates of post-grant patent opposition to members of the public by providing a much less costly and more streamlined avenue for post-grant opposition through the Patent Office's Patent Trial and Appeals Board (PTAB). Any member of the public could challenge an existing patent for only a few thousand dollars—relatively inexpensive compared to litigation (Marco, 2016).

Accordingly, the patent application process is under two types of strain: it is resource constrained—since there are more and more patents being filed every year—and it is coming under more scrutiny due to the America Invents Act. There are two main sets of stakeholders that have an interest in improving the current application process: (1) the USPTO and (2) patents filers and their lawyers.

First, because "IP-intensive industries accounted for about […] 34.8 percent of U.S. gross domestic product […] in 2010" (Economics and Statistics Administration and United States Patent and Trademark Office, March 2012, p. vii), reducing the time it takes to effectively examine a patent—perhaps through assistance from a computerized algorithm—is a critical priority for the USPTO (appendix A). Indeed, helping patent examiners reduce the time they spend on each patent (while still maintaining the quality of examinations) would mean reducing

the cost and time associated with filing patents and both prove economically accretive and reflect well on the US Patent and Trademark Office. In fact, the USPTO has expressed interest in a predictive service in the past and has conducted its own research into predicting invalidation (US Patent and Trademark Office, 2015, p. 38).

Secondly, when applying for patents, patent filers and their attorneys have a strong interest in preempting potential litigation through effective framing and wording of their patents. Patent litigation is becoming more and more common as evidenced by IBISWorld: "Demand for litigation services has boomed over the past five years" (Carter, 2015, p. 4). Therefore, a tool that could help patent filers prevent litigation would be valuable during the application process. One industry that may be especially interested in this sort of tool is Business Analytics and Enterprise Software. In the past several years, the costs associated with protecting "a portfolio of patents" have disproportionately increased in this industry (Blau, 2016, p. 22).

3 Competition

Patent validity is a major concern for the $40.5 billion intellectual property licensing industry, whose players often must decide whether to license a patent or challenge its validity (Rivera, 2016, p. 7). These decisions are currently made through manual analysis conducted by highly-paid lawyers (Morea, 2016, p. 7). Because of the cost of these searches, data analytics firms such as Juristat, Lex Machina, and Innography have created services to help lawyers perform analyses more effectively.

One common service is semantic search for prior art and similar patents, where queries take the form of natural language instead of mere keywords. Other services include statistics about law firms, judges, and the patent-granting process. These services build their own

high-quality databases by crawling court records and other public data sources, correcting or removing incomplete records, and adding custom attributes to enable such search patterns and reports. Their prevalence reflects the trend towards data analysis as a service, since law firms are not in the data-analysis business (Diment, 2016).

The above services lie outside the scope of predicting invalidation from patent text and metadata, but become relevant when discussing commercialization because high-quality data improves model accuracy and enables techniques like concept analysis that are difficult or impossible with raw unlabeled datasets. As such, partnering with existing firms that provide clean datasets or otherwise cross-licensing our technologies may be advantageous.

While these services help lawyers make manual decisions with historical statistics, we have found no service that attempts to predict invalidation for individual patents. Juristat is the only major service we found that performs predictions on user-provided patent applications. Specifically, Juristat predicts how the patent office will classify a given application and highlights key words contributing to that classification, with the goal of helping inventors avoid technology centers in the patent office with high rejection rates (Juristat, n.d.).

Our project, if successful, can become a Juristat-like service for predicting post-grant invalidation. While we cannot speculate on existing firms' development efforts, the lack of similar services on the market suggests a business opportunity. Whereas existing services target law firms and in-house IP lawyers, our project aims to help the USPTO evaluate post-grant petitions, which are brought forth by parties attempting to invalidate a patent.

4 Machine Learning Technology Trends and Strategy

This work has been enabled by many recent advances in the application of machine learning to large data problems. Even though machine learning has been around for several decades, it took off within the past decade as a popular way of handling computer vision, speech recognition, robotic control, and other applications. By mining large datasets using machine learning, one can "improve services and productivity" (Jordan et al, 2015 p. 255-256), for example by using historical traffic data in the design of congestion control systems, or by using historical medical data to predict the future of one's health (Jordan et al, 2015 p. 255-256). For this project, we had access to a large dataset of historical patent filings since 1976, for which recently developed machine learning techniques proved especially useful.

Machine learning algorithms generally fall into one of two categories: supervised and unsupervised (Jordan et al, 2015 p. 256). Supervised learning algorithms need to be run on training data sets where the correct output is already known. Once the algorithm is able to generate the correct output, it can then be used for regression or clustering. In contrast, unsupervised learning algorithms use data sets without any advance knowledge of the output, and perform clustering to try to find relationships between variables which a human eye might not notice. Recent trends indicate that supervised learning algorithms are far more widely used (Jordan et al, 2015 p. 257-260). Our historical data set indicated whether or not patents were invalidated during past disputes, which made a supervised learning algorithm the appropriate choice.

5 Ethical Challenges

As with all engineering projects, we anticipated the possibility of running into potential ethical conflicts. We used the Code of Ethics, written by the NSPE (National Society of

Professional Engineers) (NSPE, 2017), as a guideline for our planning. We identified two components of the Code of Ethics, which our project could potentially violate if left unchecked.

The first is Rule of Practice 2: “Engineers shall perform services only in the areas of their competence” (NSPE, 2017). As mentioned in Section 2, one of our target customers is the United States Patent and Trademark Office, who would ideally use our project to aid with their patent approval decisions. If our project were seen to be an automated replacement, rather than a complement, for trained patent examiners or attorneys, that may be considered an attempt to perform services outside of our “areas of competence.” While we cannot dictate how our customers ultimately utilize our product, we can mitigate the issue through thorough written recommendations in our documentation to hopefully encourage responsible usage.

The second ethical consideration is Professional Obligation 3: “Engineers shall avoid all conduct or practice that deceives the public”. While we fully intend our project to be used in service to the public, we recognize the possibility of bias in our supervised machine learning algorithm (Reese, 2016), with the resulting output capable of unfairly swinging the outcome of a patent decision. Unlike the prior ethical issue, we have more control in this situation, since we do not have a viable product without a sufficiently trained algorithm. By verifying our datasets to ensure equal representation and objective input, we can avoid inserting biases and thus maintain ethical integrity.

Conclusion

Collectively, recent economic and regulatory trends have made now an exciting but uncertain time for inventors, attorneys, and the US Patent and Trademark Office. Thoughtful applications of machine learning and statistics can make sense of these recent changes and assist

stakeholders in truly understanding what drives patent invalidation. As we pursue this technology, our understanding of the industry landscape of potential customers/competitors, leading trends in machine learning research, and the ethical considerations associated with our technology will drive our research. Ultimately, we hope that our technology contributes to the continued development of a patent ecosystem that enables inventors to do what they do best: developing novel and socially valuable inventions.

References

Beard, T.R., Ford, G.S., Koutsky, T.M., & Spiwak, L.J. (2010). Quantifying the cost of substandard patents: some preliminary evidence. *Yale Journal of Law and Technology, 12(1): 240-268*.

Blau, G. (2016a, June). *IBISWorld Industry Report 51121c. Business Analytics and Enterprise Software Publishing in the US.* Retrieved October 13, 2016 from IBISWorld database.

Blau, G. (2016b, September). *IBISWorld Industry Report 51913a. Search Engines in the US.* Retrieved October 13, 2016 from IBISWorld database.

Carter, B. (2015). *IBISWorld Industry Report OD4809 Trademark & Patent Lawyers & Attorneys in the US.* IBISWorld. Retrieved October 15, 2016

Diment, Dmitry (2016, March). IBISWorld Industry Report 51821. Data Processing & Hosting Services in the US. Retrieved October 11, 2016 from IBISWorld database.

Economics and Statistics Administration and United States Patent and Trademark Office. (March 2012). *Intellectual Property And The U.S. Economy: Industries in Focus.* U.S. Department of Commerce. Retrieved October 15, 2016

Jordan, M. I., & Mitchell, T. M. (2015, 07). Machine learning: Trends, perspectives, and prospects. *Science, 349*(6245), 255-260. doi:10.1126/science.aaa8415

Juristat - Patent Analytics. (n.d.). Retrieved October 13, 2016, from https://juristat.com/#etro-1

Love, B.J., & Ambwani, S. (2014). Inter partes review: an early look at the numbers. *University of Chicago Law Review, 81(93)*. Retrieved October 14, 2016 from: https://lawreview.uchicago.edu/page/inter-partes-review-early-look-numbers

Marco, A. (2016, October 5). Phone interview with USPTO Chief Economist.

Morea, S. (2016). *IBISWorld Industry Report 54111 Law Firms in the US.* IBISWorld. Retrieved October 16, 2016

Myers, B. A., & Rosson, M. B. (1992). Survey on user interface programming. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '92*. doi:10.1145/142750.142789

NSPE. (n.d.). Code of Ethics. Retrieved February 03, 2017, from https://www.nspe.org/resources/ethics/code-ethics

Patent Trial and Appeal Board Statistics [PDF Document]. Retrieved October 15, 2016 from United States Patent and Trademark Office web site: https://www.uspto.gov/sites/default/files/documents/2016-08-31%20PTAB.pdf

Reese, H. (2016, November 18). Bias in machine learning, and how to stop it. Retrieved February 04, 2017, from http://www.techrepublic.com/article/bias-in-machine-learning-and-how-to-stop-it/

Richardson, L. (1996). Beautiful soup: We called him tortoise because he taught us. Retrieved November 13, 2016, from https://www.crummy.com/software/BeautifulSoup/

ScrapeHero. (2014, 7). How to prevent getting blacklisted while scraping. Retrieved November 13, 2016, from https://www.scrapehero.com/how-to-prevent-getting-blacklisted-while-scraping/

U.S. Patent Statistics Chart [PDF Document]. Retrieved from United States Patent and Trademark Office web site: https://www.uspto.gov/web/offices/ac/ido/oeip/taf/us_stat.htm

US Patent and Trademark Office (2015, January). Patent Litigation and USPTO Trials: Implications for Patent Examination Quality. Retrieved October 9, 2016 from https://www.uspto.gov/sites/default/files/documents/Patent%20litigation%20and%20USPTO%20trials%2020150130.pdf

Walker, J. & Copeland, R. (2015, April 7). New hedge fund strategy: dispute the patent, short the stock. *The Wall Street Journal*. Retrieved October 15, 2016 from: http://www.wsj.com/articles/hedge-fund-manager-kyle-bass-challenges-jazz-pharmaceuticals-patent-1428417408

Appendix A: How will the invalidation prediction algorithm help the USPTO?

In 2015, the USPTO received 629,647 applications and this number is steadily growing (Patent Technology Monitoring Team, 2016). If our classifier can help them save only two hours on each application by predicting the outcome of an invalidation request and therefore help them make their decision more easily, it would save them about 430 working days (of 8 hours each).

Appendix B: Code

In [ ]:

```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from collections import Counter
from sklearn.feature_extraction.text import CountVectorizer
from nltk.corpus import stopwords
import pickle
import MySQLdb


f = open('sample_patent.txt', 'r')
g = open('sample_patent2.txt', 'r')
# word_vectorizer = CountVectorizer(ngram_range = (1,2), analyzer = 'word')
# sparse_matrix = word_vectorizer.fit_transform(f)

# frequencies = sum(sparse_matrix).toarray()[0]
# pd.DataFrame(frequencies, index=word_vectorizer.get_feature_names(), columns=['frequency'])

def calc_distance(file1,file2):
    f = open(file1, 'r')
    g = open(file2, 'r')
    input_string = f.read()
    input_string2 = g.read()
    word_freqs = Counter(input_string.split())
    word_freqs2 = Counter(input_string2.split())
    for key in set(word_freqs2):
        if key not in word_freqs:
            word_freqs[key] = 0
        word_freqs[key] -= word_freqs2[key]
    distance = 0
    for val in word_freqs.values():
        distance += val**2
    distance = distance ** (.5)
    return distance
```

In [ ]:

```
db = MySQLdb.connect("cal-patent-lab.chhaitskv8dz.us-west-2.rds.amazonaws.com","teamrocket","teamrocket","
teamrocket" )
cursor = db.cursor()
cursor.execute("SELECT patent_id, invalidated from patents_decision")
data = cursor.fetchall()
count = 0
# contains invalidated/validated patent numbers
invalidated_patents = []
not_invalidated_patents = []
# contains average distances of invalidated/validated patents
invalidated_distances = []
not_invalidated_distances = []
invalidated_distances_k = []
not_invalidated_distances_k = []

for line in data:
    if line[1] == 0:
        not_invalidated_patents.append(line[0])
    elif line[1] == 1:
        invalidated_patents.append(line[0])
```

In [ ]:

```
DB_HOST = "patentnetwork.berkeley.edu"
DB_USER = "teamrocket"
DB_PASSWORD = "teamrocket"  # Get this field from the user
DB_TABLE = "patents_decision"
DB_NAME = "teamrocket"
def db(host, username, passwd, database_name):
    return MySQLdb.connect(host, username, passwd, database_name)

patent_db = db(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME)
cursor = patent_db.cursor()

# Get list of all patent IDs in DB
print("Getting all existing patent IDs in DB")
```

```
cursor.execute("select claims_text from patents_decision LIMIT 100000")
abstracts = cursor.fetchall()
print(abstracts[0])
```

In [ ]:

```

```

In [ ]:

```
stopWords = stopwords.words('english')

# count_vect = CountVectorizer(stop_words  = stopWords)
# content = ["How to format my hard disk hello", " Hard disk format problems "]
# X = count_vect.fit_transform(content)
# print count_vect.get_feature_names()
# print count_vect.fit_transform(content).toarray()
# print count_vect.vocabulary
```

In [ ]:

```
from os import listdir
from os.path import isfile, join
files = [f for f in listdir("patents") if isfile(join("patents", f))]
patent_numbers = [name[:len(name)-4] for name in files]
#patent_numbers
```

In [ ]:

```
test_files = files[1:]

input_file = 'patents/'+files[0]
abs_list = []
for name in test_files:
    f = open("patents/" + name, 'r')
    abs_list.append(f.read())
```

In [ ]:

```
print len(abs_list)
print len(test_files)
```

In [ ]:

```
count_vect = CountVectorizer(stop_words  = stopWords)
abs_lst = [i[0] for i in abstracts ][:5000]
abs_lst.extend(invalidated_patents)
abs_lst.extend(not_invalidated_patents)
X = count_vect.fit_transform(abs_lst)
print count_vect.fit_transform(abs_lst).toarray().shape
```

In [ ]:

```
import numpy as np
invalidated_distances_k = []
not_invalidated_distances_k = []
inval_num = len(invalidated_patents)
not_inval_num = len(not_invalidated_patents)
for i in range(inval_num):
    dist = (X.toarray()[:] - X[5000+i])
    dist = np.square(dist)
    dist = np.sum(dist, axis = 1)
    dist = np.sqrt(dist)
#     dist.remove(min(dist))
    dist = np.average(dist)
    invalidated_distances_k.append(dist)
    print(i)
for i in range(not_inval_num):
    dist = (X.toarray()[:] - X[5000+inval_num + i])
    dist = np.square(dist)
    dist = np.sum(dist, axis = 1)
    dist = np.sqrt(dist)
#     dist.remove(min(dist))
    dist = np.average(dist)
    not_invalidated_distances_k.append(dist)
    print(i)
# single = X.toarray()[0]
# dist = (X.toarray()[1:]-single) **2
```

```
# dist = np.sum(dist, axis = 1)
# dist = np.sqrt(dist)
```

In [ ]:

```
print(len(invalidated_distances), len(not_invalidated_distances))
print(len(invalidated_distances_k), len(not_invalidated_distances_k))
print(sum(invalidated_distances)/float(len(invalidated_distances)))
print(sum(not_invalidated_distances)/float(len(not_invalidated_distances)))
print(sum(invalidated_distances_k)/float(len(invalidated_distances_k)))
print(sum(not_invalidated_distances_k)/float(len(not_invalidated_distances_k)))
```

In [ ]:

```
import matplotlib.pyplot as plt
import numpy as np

# import plotly.plotly as py
# Learn about API authentication here: https://plot.ly/python/getting-started
# Find your api_key here: https://plot.ly/settings/api

plt.hist(invalidated_distances, bins=range(int(min(invalidated_distances)),1))
plt.title("'Invalidated'Distance Histogram. Average: 22.4")
plt.xlabel("Distance")
plt.ylabel("Frequency")

fig = plt.gcf()
axes = plt.gca()
axes.set_ylim([0,8])

plt.figure()
plt.hist(not_invalidated_distances, bins=range(int(min(not_invalidated_distances)) 1))
plt.title("'Not Invalidated' Distance Histogram. Average: 23.2")
plt.xlabel("Distance")
plt.ylabel("Frequency")

axes = plt.gca()
axes.set_ylim([0,8])
plt.show()
# plot_url = py.plot_mpl(fig, filename='mpl-basic-histogram')
```

In [ ]:

```
import pickle
with open('distances.pickle', 'w') as f:  # Python 3: open(..., 'wb')
    pickle.dump([distances,not_invalidated_distances,invalidated_distances,not_invalidated_distances_k,inv
alidated_distances_k], f)
```

In [ ]:

```
len(invalidated)
# len(not_invalidated)
```

In [ ]:

```
with open('distances.pickle', 'r') as f:  # Python 3: open(..., 'wb')

[distances,not_invalidated_distances,invalidated_distances,not_invalidated_distances_k,invalidated_distanc
s_k] = pickle.load(f)
```

In [ ]:

```
import matplotlib.pyplot as plt
import numpy as np
data = np.vstack([invalidated_distances, not_invalidated_distances]).T
plt.hist([invalidated_distances, not_invalidated_distances], bins=range(int(min(invalidated_distances)),30
, 1))
plt.title("'Invalidated'Distance Histogram. Average: 22.4")
plt.xlabel("Distance")
plt.ylabel("Frequency")
plt.show()
```

In [ ]:

```

```