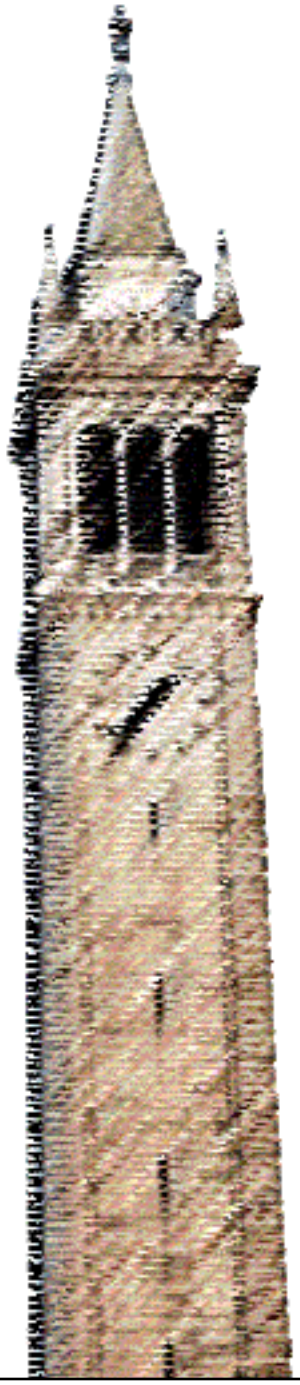
Large Scale Text Analysis



Kevin Tee

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2015-145
http://www.eecs.berkeley.edu/Pubs/TechRpts/2015/EECS-2015-145.html

May 15, 2015

University of California, Berkeley College of Engineering

MASTER OF ENGINEERING - SPRING 2015

Electrical Engineering and Computer Science

Data Science and Systems

Large Scale Text Analysis

KEVIN TEE

This **Masters Project Paper** fulfills the Master of Engineering degree requirement.

Approved by:

1. Capstone Project Advisor:

Signature: _____ Date _____

Print Name/Department: LAURENT EL GHAOUI/EECS

2. Faculty Committee Member #2:

Signature: _____ Date _____

Print Name/Department: LEE FLEMING/IEOR

Abstract

We take an algorithmic and computational approach to the problem of providing patent recommendations, developing a web interface that allows users to upload their draft patent and returns a list of ranked relevant patents in real time. We develop scalable, distributed algorithms based on optimization techniques and sparse machine learning, with a focus on both accuracy and speed.

Table of Contents

*Co-written with Johanna Ye and Weijia Jin

Problem Statement

Small businesses often face many challenges when filing for a patent and searching for relevant patents, from the threat of potential litigation from large corporations to costly lawyer fees. Currently, most patent-related tasks are often done by lawyers, an option that is both expensive and time-consuming. Lawyer fees can often cost up to thousands of dollars, although their services take on a broader range: from understanding the patent filing process to domain-specific knowledge to patent recommendations. We aim to automate the process and expedite the search process by helping researchers and inventors determine if their technology development or unique idea has been previously patented. We take an algorithmic and computational approach to this problem, developing a web interface that allows users to upload their draft patent and returns a list of ranked relevant patents in real time. With the growing amount of data available, as well as computing power and fast storage and retrieval, taking a data-driven approach can utilize this data to solve difficult problems, especially those related to text. Text mining technology can be directly applied to completely automate patent matching within the patent database. This technology has a wide range of applications in many different industries, from aiding fellow researchers in finding the information they need, to providing data analytic tools for working professionals from any field or domain (Smith, 2014). Because we lack substantial computing power, we are forced to think about algorithms that can scale without the the use of large machines or GPUs. We develop scalable, distributed algorithms based on optimization techniques and sparse machine learning, with a focus on both accuracy and speed. Our methods rely on both state-of-the-art research in academia mixed

with our own novel approaches. Currently, we develop an unsupervised approach to this problem, although recent we have investigated semi-supervised approaches with much promise. Our method is targeted at small businesses who are searching for a cheaper alternative to lawyer services. Not only do we offer a low-cost alternative for small businesses, but our algorithms are real-time, and return results in a few seconds. As more users use and provide feedback based on the relevance and ranking of our results, the algorithm will improve (again, in this semi-supervised setting), allowing us to iterate on our current prototype integrating a user feedback loop.

Strategy

This section will address our strategic positioning from a market perspective, touching on our relationship with customers and suppliers, our "go-to-market" strategy, and industry trends. Additionally, it connects these ideas to those of our competitive landscape, discussing industry potential and our competitive advantage. The first part of this section will investigate the market and our competitors as well as the barriers to entry. The second portion of this section will do a more in-depth analysis of the industry, examining the patent industry and our differentiating factors from the lense of several of Porter's five forces. At the end, we will look at the various trends in both the patent industry, and how it affects our short and long term strategy. Those components of the industry analysis will present a unifying view of our market strategy and the factors which accompany it, integrating the Five Forces model throughout the paper (Porter, 2008). We start by analyzing the competitors and potential substitutes for our product.

Two substitutes, Google Patent Search and the USPTO website, do not pose a large threat to our product because they lack a quality algorithm and a robust, informative UI giving our product a unique competitive advantage. Since the market and industry are intertwined and dependent on each other, a brief introduction of the market can shed light on the industry we are in and the substitutes or solutions in this space. Our primary target market is small businesses that have very limited resources but want to protect their technology or avoid using patented technology by providing effective patent similarity searches (we discuss this more in subsequent paragraphs). Our industry is within the software industry that helps businesses protect their IP and avoid litigation incurred by using patented technology. In this context, three major

substitutes/alternative solutions deserve our attention and analysis: United States Patent and Trademark Office (USPTO), law firms, and online patent searching services (e.g. Google). For each of these substitutes, we describe what about our product offers that the other does not, or our competitive advantage.

The USPTO presents an obvious substitute for any patent search product, providing a free web search interface that is alluring to small businesses due to the cost and the brand name; however, the USPTO's search functionality is limited in a few key ways. The USPTO reviews trademark applications for federal registration and determines whether an applicant meets the requirements for federal registration. They have the most up-to-date and comprehensive database regarding all of the patent data, but the website does not answer questions regarding whether a particular invention is patentable and that is also difficult to use and navigate. They only offer keyword search (e.g. "Machine Learning") but not document search with similarity ranking, and often when small businesses are looking to do a patent search, they want to compare it against their draft patent. For the purposes of determining patent similarity, the USPTO serves more as a supplier than competitor. Our technology is meeting an unmet need that the USPTO doesn't satisfy, and in the process making it more convenient for small businesses. We provide much better functionality (as well as a more user-friendly web interface), rendering USPTO a weak substitute/competitor in the dimension that our product will compete.

The other substitute/competitor is the law firm which facilitates the patent filing process, but our product is significantly cheaper while providing a similar, if not better, quality of service. The law firm can carry out multiple functions: it can help determine if an invention qualifies for a patent or potentially infringes on an existing patent; it can guide the customer through the patent

application process, and the law firm can work with the customer to enforce the patent against potential infringers. However, our research shows that a professional patent search may cost approximately $500 to $1,500, while obtaining a patentability opinion from a lawyer costs approximately $1,000 to $2,000 (Costowl.com, 2014). Additionally, setting up an appointment with a lawyer is a slow process and often requires multiple visits. Our product addresses both of these issues by providing a cheaper alternative for small businesses and giving near instantaneous results. We are able to achieve these near instantaneous results through efficient indexing and parallel algorithms on the backend of our website. If there was an online service that provided a patent similarity match and gave instantaneous results, the response speed and the reduction on cost would provide immense efficiency and reduce the burden on budgets for small *and* large businesses alike. However, our service can be somewhat subpar to law firms as we don't provide the personalized "touch" or face-to-face interaction with a small business. There is an inherent tradeoff that a small business owner must make, and we've decided to segment the market such that we enter at a cost-effective end. Our product may not outperform every aspect of a law firm, but for our target market, small businesses, it is the most sensible option.

Our third competitor is the online websites for patent searches, and while these services are most close to the service we aim to provide, our product holds a unique competitive advantage. The most publicly-known search is Google patent search. According to Google, their patent search ranks results according to their relevance for a given search query without any human oversight. However, Google Patent doesn't support document search and the algorithm is not transparent. Our project differentiates itself from Google Patent in that our functionality supports document similarity ranking and provides strong interpretability that helps the user qualitatively understand

the results. Our goal is to help users understand the results that are given back to them: make sense of which parts of each document are most similar, as well as ask the user to give some feedback on the quality of the results. This gives us to the ability to improve our algorithms by integrating user feedback and improving our results with time. In a sense, what we are building is not just a search engine, but a interactive website to help small businesses that is adaptive and gets better overtime.

The threat of new entrants is of a moderate concern, although it requires time to establish oneself in the market. Given that the particular industry doesn't have a strong force from the supplier and the ease of getting the data to start a new online service with patent ranking functionality, we note that the barrier to entry is low, as with most open source software projects. However, the quality of the algorithm and performance under large amounts of data is one of the most important differentiation factors from one online service to another. It takes time to aggregate data, discover the optimal solution for data storage and iterate on the algorithm with more user data. Additionally, to protect our leading position and our state-of-the-art technology as the incumbent in the space, we will consider filing patents for the database configuration and algorithms that we are using in an effort to raise the barrier of entry for new entrants.

The interpretability of software products impacts the user's decision for choosing the product. Early entry allows us time to fine tune the algorithm and develop customer relationships. With early customer relationships, we can develop demand-side benefits of scale. As more customers use our technology and advocate it, this in turn will generate more new customers. The more customers we have, the better testing data we can obtain. With the aggregation of data, we develop better products and the positive iterative circle continues. Over time, as more customers use our product and we apply interactive feedback, the greater success we will achieve.

Developing this feedback loop is crucial to our success and helps to prevent the new entrants who have not been in the industry long enough to aggregate user data.

As noted above, the major challenge and potential competition in our industry is the quality of algorithms and customer relationships. Threats of competition, both current and future, is relatively low; however, to maintain our dominance requires us using our position as incumbents to constantly iterate and improve our algorithm, and to us our own Intellectual Property (IP) to raise the bar of entry for new competitors. Having established our competitive advantage in our industry and differentiating ourselves from various competitors, we turn now to analyze our customers and suppliers, and our relative positioning among them.

While the demand for patent recommendations is somewhat diverse, our target customers will be small businesses. The needs for patent recommendation and patent services span across many markets, ranging from commodities such as coffee machines to huge operations such as pharmaceuticals (MINTEL, 2014). Interestingly, MINTEL does not have a section of their reports on the patent market, but is a common topic in many articles, suggesting that patents are crucial in many markets. Currently, patent lawyers tend to drive the market, with both their expertise and ability to customize to their customers with services include patent application and renewal services, litigation services, and patent novelty search (Carter, ; Hoovers, 2014). Our product aims to compete directly with patent novelty search, with our differentiator being novel algorithms and speed, both of which we will discuss in the following section. The number of small businesses in the United States is approximately 23 million, accounting for 54% of all the sales, and they provide 55% of the total jobs in the United States since 1970. Most importantly, the number of small businesses has increased by 49% since 1982, suggesting its enormous growth and potential (SBA).

Small businesses as customers exhibit a strong force in the context of Porter's five forces because small businesses have many options for patent novelty search, in both lawyer firms, other search engines, and online resources. Porter argues that the industry structure drives competition and profitability, and in our market, the differentiating factor is the quality of service (in this case patent recommendations, reliability, algorithms) suppliers can provide (Porter, 2008). We therefore turn to weakening the forces of customers by looking at our competitive advantage in the context of a marketing strategy.

Designing a market strategy using the 4Ps model will give us the ability to take our idea and technology and offer it as a robust service (McCarthy). As stated earlier, our competitive advantage lies in our ability to provide apply novel algorithms to the problem of patent recommendations, with excellent results in both speed (time it takes to return a query) and accuracy (relevance). Along with law firms, there are other search engines for the same purpose: Google Patent Search and subject-oriented databases (Google, Carpenter). Currently, our product is a minimum viable product (MVP): we have a minimal, simple web interface with an algorithm that has huge potential to be improved with customer feedback. Our pricing model will be per-document based: for each draft patent, a customer will be charged a flat rate. This rate will be significantly cheaper than going to a lawyer, which can cost thousands of dollars (Carter, 2015). Promotion strategy will be largely dependent on our positioning within the market and the market segment of interest. While the market for small businesses is large (23 million total small businesses), not all of these companies are in need of IP or patents (SBA). For example, a local restaurant would not need our services. Instead, we define our market segment to be small businesses in the United States less than two years old with unfiled IPs or patents. The service will

be promoted primarily through a single channel: advertisements on social media sites such as Facebook, Google, etc. Running a targeted ad campaign toward small businesses will help to spread the word about our service. Overtime, our goal is to establish a reputation and gain more sales and awareness from word of mouth (so future customers will be more drawn to our product). Distribution (place) is easy for us, as our service is a website (although we host our content on servers, more about that in suppliers) and is therefore easily accessible provided one has internet connection. With these 4Ps and our competitive advantage, we have positioned ourselves to weaken the force from the customers, allowing us to differentiate ourselves in a competitive market.

The two suppliers we will need are web servers and a patent database. The power, or force, of both of these suppliers is low, much to our advantage. The trend toward using web services (Amazon, cloud, etc.), along with cheaper storage and more competition, will drive down the price (Hart). Colloquially, web services is a service which allows website to run queries (such as a Google search, or patent algorithm). These web services tend to be run in large databases, or "data farms", which are dedicated to processing large amounts of information or queries. Moreover, because our development is still in the initial state, switching costs are low and moving from one web service to another web service is relatively easy. Additionally, obtaining patent data from the United States Patent Trademark Office (USPTO) or Google Patent Search is relatively simple. Our program is able to "scrape" all the relative information from these websites and deposit the information into a database. Because these patents are available to the public, they will remain available and easy for us to use. The force from the suppliers, as with many web startups, is low and puts us in a good position going forward with our market strategy.

Despite the strong force from customers in Porter's five forces models for our patent recommendation service, having carved out a particular subset of the enormous market helps to weaken this force. Along with the 'go-to-market' strategy using the 4Ps model, a weak supplier force, and analysis of industry and trends from the complementing papers, our service can immediately make a substantial impact in the market. In addition to understanding the forces from customers and suppliers and how to mitigate them, it is important to understand the current market trends related to our product. In particular, we look at technology trends related to machine learning and big data.

As research in artificial intelligence (AI) bears fruit, we are increasingly seeing its applications in everyday consumer products, from smartphones that can take instructions directly from voice commands to online retail websites that can make uncannily accurate predictions of what goods we need or what movies we may want to watch next. Patent search/recommendation is another one of those applications which took advantage of recent developments in natural language processing and machine learning techniques. What used to be a task monopolized by lawyers who made use of their expertise in patent law to help clients determine whether their new product or technology is patentable (USPTO, 2014) can now be accomplished using software at not only a much faster speed but also at a fraction of the cost (NOLO, 2014). This is possible all thanks to the technological trend towards smarter and more cost-efficient computing.

In the past, the sheer size of the patent database (numbering in the millions and increasing by the thousands each week) and the fact that patent documents are notoriously hard to decipher due to their abundance of legal jargon (USPTO, 2013), are a huge deterrence to the layman who wish to extract some information (needle) from all the patents (haystack). However this is no

longer the case with the advent of intelligent search and recommendation systems that not only returns query matching results in the blink of an eye, but also directs the users straight to the relevant portion of the document, saving them the trouble of having to navigate through the dense text. The possibility of translating this previously exclusive domain knowledge of patent lawyers into computer algorithms opens up the opportunity for software to come in and contest the monopoly previously held by patent lawyers over the business of patent search (Carter, 2015). We plan to fully exploit this opening in the market brought about by new technology. By packaging our novel patent search algorithm as an online web service, we wish to market it as an attractive and widely accessible platform for any interested party to conduct their patent searches.

The business of patent search is very lucrative, particularly in the technology industry where new technical breakthroughs and their protection can make or break a company. Top technological giants such as IBM average 9.3 patent applications in a single day (IBM, 2014). This focus on creating and maintaining a technological advantage will only intensify with the proliferation of start-ups that are built on the latest technology. The huge opportunity presented here has attracted a lot of big players into this field including Google Patent Search, Espacenet and many others (Google Patent Search, 2011). As a new entrant, the competition posed by these established services and future entrants will be a big challenge. However since the advent of online patent search has been relatively recent, there is still room for us to mitigate the threat of intense competition by differentiating ourselves from our competitors by building smarter algorithms, a more intuitive user interface and more insightful data visualizations. By departing from the standard model of keyword search to document search, we will be able to serve an as yet unmet need of companies which need to be able to check in a timely fashion, if what they are currently

developing infringes on any published patents. Another way we can reduce the effects of market saturation is to expand our customer base. The traditional customers of patent search services are mainly organizations who want to find a certain technology or to know whether their inventions can be patented, and can afford the costly lawyer fees (Lawyers.com, 2014). A new group of users who will be receptive to easily accessible patent search web services are researchers, students or the curious browser who wish to gather more information about a certain invention or technology, regarding its history, inventor(s), inspirations etc. We can reach them by marketing our product to university departments and through our research circles.

Advances in AI technology has made our product possible and has given us an opportunity to disrupt the industry of patent search. However, this has also created intense competition within the industry which we hope to overcome through product differentiation and customer base expansion. With our novel algorithm targeted at small businesses, we have found a portion of the market which we believe we can succeed, and slowly grow our customer base through word of mouth advertising. In addition to strategy, however, remains its counterpart: operational effectiveness. Our strategy and analysis using Porter's model is not enough, but it does provide the necessary framework to be able to execute and ultimately carry out our vision.

Intellectual Property

In this section, we will describe our intellectual property (IP) and describe our choices that surround our IP. For a particular IP, there are several interesting options to explore, including filing a patent, making the work open source, etc. In our case, we have decided that while our algorithms are potentially patentable, we will not file a patent nor pursue any other alternate IP strategy. There are several reasons for this decision, which we discuss in the subsequent paragraphs. It is important for the reader to understand, however, that our decision not to pursue IP strategy is what we perceive to be the best choice in the present, and that it may change as our product and competitive landscape evolve.

The most patentable item of our project is the our algorithm for recommendations, but the modularity of our algorithm makes it inherently difficult to patent. An often cited example of a "good" and "patentable" algorithm is Google's PageRank algorithm, filed by Larry Page during the birth of Google. In their algorithm, there is a very clear statement of how their patent differentiates current work, asserting that the number of links going to a particular webpage is proportional to its importance (Page). In our algorithm, however, while it uses state-of-the-art and current optimization techniques, is more modular by design. This means that our algorithm has many different components that work together (extracting topics from the corpus, running similarity metrics, etc.). Each of these components aren't particularly new: they reference longstanding literature and papers in the field. We are able to achieve good results because of our creativity in putting these modules together, not because we developed a landmark breakthrough. One could argue that patenting the method of combining modular algorithms together would make

sense. However, this would make our algorithm available for the public to build upon, which would put us at a competitive disadvantage. We discuss our reasons against this potential solution in the following paragraph. Additionally, investigation into the patent database shows that there are almost no patents, from a machine learning perspective, that do a similar "combining" of existing work in the software/algorithms field (Google Patent Search). This suggests that it will be difficult to patent our algorithm, likely because there is not enough of a differentiating factor.

A brief cost benefit analysis of patenting our algorithm shows that the resources invested will not provide significant returns, as well as open the door for competitors to see our "trade secret". Filing a patent is estimated to cost $2000, with $130 alone for the actual submission and approval of the patent by the United States Patent Trademark Office (USPTO). In theory, our primary motivation for filing a patent would be to protect ourselves in the case where an independent developer comes up with our algorithm. This however, is very unlikely because of the complexity of our algorithm and our use of ideas from different fields within computer science: optimization, probabilistic models, machine learning, and efficient database access and retrieval. Rarely do software corporations dispute over particular algorithms, but rather specific features in products such as touch screen display for phones (Bosker). This is because it is difficult to draw the line between the "same" and "different" algorithms, especially when there are many moving parts like our algorithm. A more pressing concern for us is the danger of leaking our algorithm to competitors. One nice aspect of developing a web application is that the algorithm is hidden, and there is no way for people to see how we provide the recommendations. Consumers (and competitors) can only see the final result, the ranking of the patents. If we file a patent, however, it gives competitors full access to our algorithm and be able to improve upon it. Again, because it is

difficult to differentiate an algorithm, our patent may not be significantly similar to their improved methods and we would be unable to file a patent lawsuit. In our case, filing for a patent gives an unwanted consequence, in addition to the time and money incurred throughout the process. More succinctly, filing for a patent will reduce our competitive advantage in our industry and provide more of a hinderance than a benefit.

Keeping our IP secret will allow us to continue to iterate and improve our algorithms with more customer feedback. Part of our continued strategy moving forward is to use customer feedback and user studies to refine and tweak our algorithm. If we file for a patent, we are in a sense "binding" ourselves to that patent. In software engineering, a common practice is agile development, where small incremental improvements are made in 1-2 week "sprints". Having adopted this method for our group, we are able to make quick improvements because of the flexibility to tweak algorithms and rapidly iterate. As a result, filing for a patent will inhibit this structure and make it more difficult to innovate. Nonetheless, a big part of the software engineering culture is giving back by contributing to open source. Open source is the idea where software developers make their code readily available online, so that other developers can build upon this work and push the field of computer science (High). Some extend this idea further and assert that software engineers have an obligation to contribute to open source communities such as Github (Masson). While we will not be making our algorithm publicly available, we will make parts of our code available such as data parsing and database indexing, under a copyright, a free license for software such as BSD or GNU Public License. This is consistent with many startups and technology companies today, who do not place their "secret sauce" on a publicly available repository.

Our group's IP strategy is to not pursue a patent, copyright, trade secret, or other forms of legal recognition of our work. While we plan to open source some of our code, in align with the culture of software engineers, we plan to keep our algorithm secret. Our motivation for doing so fall into two large categories: the difficulty of getting our algorithm patented, and the consequences incurred as a result of this course of action. We view the latter as more damaging to our ability to produce a great, differentiating product because it releases our ideas to competitors and prevents us from innovating at a fast rate. However, as we move forward, it is important to keep our options open. If we find a new way to conduct user studies, for example, it might be worth filing a patent, as it is more concrete and less susceptible to misinterpretation. Similarly, if we decide to make more parts of our code open source, then we will accordingly copyright the work under some software license.

Technical Contributions

In this section, I will focus on my technical contributions to this project, looking at the various methods used to scale our algorithm from a small dataset to an extremely large dataset. In relation to my group's technical contributions: Johanna will be discussing the core of our algorithm (on a small dataset) and Weijia will dive deep into evaluation metrics and how we score and benchmark our relevance rankings. While there other aspects of our project such as user interface design (and user experience design), database development (querying and indexing), and code architecture (client-server framework), our project's competitive advantages lies in the algorithm we develop, how we scale the algorithm, and ultimately how we evaluate and score our algorithm. As a brief overview for the rest of the paper, I first discuss existing work in scalable algorithms for recommender systems and how we can use and improve upon them. Then, I will talk about our methodologies (and choices) that build upon the existing work and discuss our results, from both algorithmic and systems point of view. Finally, we discuss why this work is significant in the context of our project, future work and how our work can be extended to yield better results.

There are over 9 million patents in the United States Trademark Patent Office (USPTO) database, and our algorithms have to be fast enough to deliver near-instantaneous to our users. Scaling algorithms from small to large datasets is not an easy task, and it is an ongoing topic of research. Current research, in both industry and academia, explores a few channels to solve this problem. The first is developing more efficient or domain-specific algorithms, which exploit certain characteristic of the dataset to design faster algorithms. For example, if our dataset is sparse, then certain optimization techniques can be applied to yield better results. In fact, in recent years, a new field of sparse machine learning has gained traction because of the nature of many datasets to be

sparse. It turns out that our dataset exhibits these sparse properties, allowing us to apply research methods to tackle our problem. Current research focuses on an overarching concept called the "lasso", which given sparse inputs, will also tend to give sparse outputs. As an example, an output for a sample problem might be which words correlate with a certain word the most, and a sparse output limits the number of correlated words. The lasso therefore gives more interpretable results, because it limits the amount of outputs information by enforcing sparsity. The lasso was first developed in 1996 by Robert Tibshirani, who described its canonical form, or its objective function (Tibshirani, 1996). Later, a famous paper by Michael Osborne described the lasso in the "dual form", a form commonly used in optimization along with its counterpart, the "primal form" to efficiently solve sparse problems (Osborne, 2000). In recent years, however, it was found that the traditional lasso does not scale well to large datasets, and improvements have been made through feature selection and elimination (El Ghaoui, 2012), and using low-rank approximations to the lasso using a new concept called iterative hard thresholding (Yadlowsky, 2014). Additionally, work on lasso and sparse machine learning has been done on text datasets, similar to the dataset we are using. These techniques have been used to uncover word associations in large text corpora (Gawalt, 2010; Zhang, 2011), and have proven to be effective both in accuracy and in speed. However, these methods have been applied to specific generic text datasets (such as the New York Times dataset), and in our work we will extend it to the patent database.

The second is preprocessing the data and extracting the relevant information so that is can be accessed faster at runtime. Because our service is focused on offering patent recommendations in real-time, or near real-time, and so our algorithm must be able to return query results fast. There are many methods for preprocessing the data commonly used in industry, ranging from efficient

indexing in databases to clustering. In this work, we will focus on clustering, a method which effectively "clusters" the data into relevant topics (e.g. all documents in cluster 1 contains patents related to medicine, all documents in cluster 2 contains patents related to machine learning, etc.). In the methodology section, we will describe how and why this method achieves faster results. Clustering is a technique that has been developed for many decades and in recent years there have been more complex and fancy models (e.g. Bayesian generative models or Latent Dirichlet Allocation). In practice, however, many of these models do not perform well on large datasets, as their theoretical properties fail. In this work, we discuss commonly used preprocessing methods for clustering, techniques that have been used for many years but work well in practice. Ward clustering uses a distance metric between data points and creates a hierarchical structure that shows the relationship among data points, often visualized as a dendrogram (Ward, 1963). The tree-like structure gives us nice properties that we can exploit during runtime. Similarly, we can use a Gaussian mixture model or k-means, which effectively partitions into a set number of clusters based on their similarity (McLachlan, 1988). Recently, several novel clustering developments have given us surprisingly good results, especially since the results are in low-dimensional manifolds rather than high dimensional space. However, we defer from discussing these methods as they as still in development. There are other common clustering methods that we have tried but have decided not to use, and we direct the reader to a paper by Charu Agrawal, whose ideas we've adopted in our work (Agrawal, 2012). With these two methods, our algorithm builds upon this work by applying these ideas in novel ways to our patent dataset. While we do not make significant changes to these clustering algorithms, we present them in a the context of a new problem, for patent data and recommendations.

In our work, we use existing research in both domain specific algorithms (improvement to the lasso), preprocessing via clustering, and parallelization from the systems point of view. We further argue that these methods are able to work together in tandem to scale our algorithm effectively, as well as show the potential limitations. Throughout this discussion, I will make frequent references to the size of the feasible dataset - the number of datapoints (or documents) that our patent recommendation algorithm is able to process in a reasonable amount of time (hence scaling the algorithm to larger datasets). In this paper, we define a reasonable amount of time as less than 5 seconds, and our goal is ultimately process more documents in the given timeframe.

When scaling algorithms to perform on large datasets, the common approach is to use more computing power (faster computers, GPUs, etc.). Recent advances in hardware have made computing power more inexpensive and easier to use, especially for large datasets. An algorithm that runs on a local machine, such as a laptop, is often orders of magnitude slower than running in on a distributed cluster. Additionally, when multiple machines are used in parallel, the computing power and processing speed increases even more. However, while this approach is appealing, we lack the resources to purchase a large amount of computing power. This is one of our primary disadvantages to our competitors, for example, Google has large amounts of computing power, making it easier for them to "throw resources at" the problem. Therefore, we must turn first to developing smarter algorithms and obtain results that are provably better than existing benchmark data before using computing power.

Even with the recent advances in scalable algorithms, there are still many options to explore, and we must decide what area of research we want to explore and ultimately implement. Because of the wealth of the research in machine learning, our time is limited and we must focus

on what we believe to be the most lucrative options. We have decided to pursue algorithms that use optimization techniques such as lasso, low-rank approximation, linear programming, etc. There are a few reasons for this decision, the first being that our advisor is a leading research in this field and can provide us with the guidance and suggestions for scaling our algorithm. Many of the research papers discussed earlier were published by El Ghaoui and other collaborators, so the insight he can give for these algorithms will be helpful. Second, and perhaps more importantly, optimization has provable guarantees for runtime, meaning that there are upper and lower bounds to how long and short an algorithm should take in theory. This is unlike a probabilistic model, which more common, does not have the same convergence properties that optimization models have. Moreover, probabilistic models are often difficult to interpret and the results are obscure because of its many layers.

Our algorithm that works well on a small dataset, as described by Johanna, uses a low-rank approximation combined with a lasso. In short, suppose a small business has a new (draft) patent. We write this draft patent as a linear combination of all the patents in the database, and the patents that are most related have the highest weight in our model. We further enforce the results be sparse, so that we only get a subset of all patents (with non-zero weights). However, this algorithm is fast only for approximately 5000 patents; in other words, comparing our target patent against 5000 other patents, far fewer than the total 9 million patents. Therefore, we use ward clustering, an agglomerative clustering method to first partition the patent dataset into different clusters. For example, all patents dealing with machines might be partitioned into one cluster. We take this one step farther and develop a hierarchical clustering extension: within clusters there are additional clusters. Continuing from the example, within a cluster of machines might be a cluster of patents

related to robotics, others related to air conditioning machines, and so forth. None of these cluster formations happen during the actual query, it happens prior, therefore making it a preprocessing method (it can take hours to form the clusters). The primary bottleneck of our algorithm comes when we have to write our draft patent as a linear combination of all the other patents (the lasso). What clustering enables us to do is to first find which subset of patents it is most similar to (via the precomputed clusters), which we can then run our lasso on. This means that each cluster can have up to 5000 documents and we can still achieve the same performance. If we have 10 clusters, then we can now process 10 times more documents, or 50000 documents in approximately the same amount of time. However, this method must perform clustering in an intelligent way: if there are related topics in different clusters, then our algorithm will achieve poor results. Cluster analysis and the relevant scoring functions will be described in Weijia's paper.

Our algorithm has scaled by 10x by using clustering, but to improve the algorithm for even larger datasets, we turn to optimization techniques and state-of-the-art research to improve the lasso. First, we use SAFE feature elimination, a technique developed by El Ghaoui and a few collaborations, which simplifies the lasso problem by eliminating certain features. In our case, when we write our target document as a draft of other patents in our corpora, it is able to eliminate several documents prior to running the lasso. Second, we use a new method recently developed at UC Berkeley called iterative hard thresholding, which cleverly speeds up the lasso problem. The details are beyond the scope of this paper, so we direct the reader to the paper for technical specifics. Combining these two methods allows us to process around 500000 documents, approaching 1 million in some cases, a 10-20x improvement. Unfortunately, our methods still fall well short of the 9 million documents we are interested in processing in near real-time, but we have

several proposed methods moving forward. This algorithm has been developed solely without using large amounts of computation power. At this point, we will begin to explore using computation to speed up our algorithm. A somewhat related option is to use a programming language that is "closer to the machine" and runs faster because it is compiled directly into binaries such as C or C++ instead of Python. Even with more computing power, it will still require some thoughts because algorithms don't necessary scale well. Generally, we look to parallelize our algorithms, but the current implementation of the lasso is not parallelizable. Therefore, we will have to rewrite our algorithm to achieve the same task while distributing the workload among several machines. Luckily, there has been much research done in this field, as the gap between systems and theoretical research gets smaller.

Our approach uses not a single method, but combines a few methods together (clustering, SAFE feature elimination, low-rank approximations, IHT, parallelization). At each step, the additional method allows more data to be processed and scales our algorithm by a certain factor. This approach is what distinguishes our methodologies from other approaches: using research from various subfields and combining them to ultimately solve the problem of patent recommendations. However, it is important that while the algorithm may run fast, it must still have good patent results. Again, Weijia's paper will discuss more on evaluation and scoring metrics in an unsupervised approach.

There are several results that must be evaluated during the scaling of the algorithm. The accuracy of the results is an unsupervised problem: there is no "correct" answer. Weijia has devoted a lot of time developing and trying to understand which metrics make sense and how to integrate external sources such as Google Patent Search to evaluating these results. Consequently,

in this paper, I will focus less on the accuracy on the results and instead more on the scalability metrics: number of documents per unit time and amount of resources for a given document. Ultimately, the goal of scaling an algorithm from a quantitative point of view is to increase the number of documents processed per unit time which decreasing the amount of resources for a given document. A more quantitative technical report with graphs and figures will be made available if the reader, but in this report we omit them for the sake of simplicity and brevity.

Originally, the number of documents we were able to process in under 5 seconds was approximately 5000. With clustering as a preprocessing method and optimization techniques, the number of documents is upward of 500000, an increase of over 100x. Note, however, that the time required for preprocessing for clustering grows exponentially with more documents. Preprocessing for 1000 documents is around 1 second, but for 1 million documents can take up to 1 hour. Our primary goal is to provide a real-time service to small businesses who intend to use our service. In order to achieve this, the preprocessing step (while computationally expensive) is necessary, and it only needs to be run one time. Measuring the amount of computational resources for a given document is a tricky yet crucial task, as the term “resources” is loosely defined in our context. There are several definitions that we may use, and each has its own benefits and drawbacks. A resource could be the number of read/writes to memory (RAM not hard disk), a common denomination for systems researchers. However, this tends to be used in database design and implementations, and not in machine learning research. Another definition for resource could be the number of operations performed on each document. This is difficult to quantify, however, because each operation is subject to other factors that will invariably affect these values (CPU lag, memory read/write inconsistencies, etc.). Finally, a unit of resources can be thought as related to

time. While this definition may seem trivial and unreasonable, it is commonly used as benchmarks for technical reports within the machine learning field. Moreover, it scales well to parallel algorithms, as each unit of time becomes each unit of time per machine. Consequently, we use this definition for the second metric, and because we have not yet exploring parallelization to a great extent, the analysis of resources consumed is analogous to the previous metric. The implementation of measuring time is an easy operation in code, we've used the time library of the Python language to time our code at the start and at the end, taking the difference as the total time. We also timed the core process of our code (the lasso and its improvements).

Without scaling, we wouldn't be able to offer a quality service to our customers. Amazingly, we've managed to achieve great results without much excess computing power that most competitors use. Small businesses who have a draft patent are interested in finding all patents that they could potentially be infringing on. A smaller subset of all patents could potentially miss many of these relevant patents, but with the techniques developed, the breadth of our service becomes much greater and much more attractive to potential consumers. Our competitive advantage lies in our ability to provide a fast service (relevant set of patents), and a slower service would allow our competitors to provide a better customer experience. We anticipate as our algorithm achieves better results and we move to commercializing this technology, more computing power will be available to us, allowing to move our research from a purely algorithmic approach to a blend of algorithm and systems research. Of course, scaling an algorithm is meaningless if the results are poor as the dataset increases. To combat this, scoring functions and other evaluation metrics are crucial and work together with the scaling of the algorithm.

Our work provides a strong framework for future iterations and improvements. The modularity of our code because of how we've pieced different areas of research together makes it easy to swap different ideas in and out. Surprisingly, despite the modularity, various parts of our algorithm works well together and scales gracefully. This allows us to keep up with current research, and as new algorithms or ideas for modules of our code become available, we can improve our results without have to re-write large portions of the codebase.

Concluding Reflections

The results we achieved were good, but they certainly were not good enough for a product. It was an extremely difficult problem, and we made great strides in approaching the problem and trying new methods. A large part of our difficulty was our lack of resources: we didn't have distributed clusters, fast relational databases, etc., which prevented us from simply "throwing resources" at the problem, a technique used by many companies and even non-technical individuals in academia. Consequently, we were forced to think smarter and develop new methods to tackle this problem. As the lead for our team, I had many ideas for methods we could use, and together we tried many of them, with varying amounts of success. Only a few of the methods we used have been documented here, and we had a laundry list of experiments we have yet to implement. Going forth, we'd recommend trying these methods that have initially shown great promise, but have yet to be implemented on a large scale. However, there is always an inherent tradeoff between the complexity of the model and the scalability of the model. A more elegant and mathematically intense model may work well on a smaller dataset but fail to scale as more data is added. Because one of our goals for this project was real-time relevant results, both the quality of the results *and* the speed of the results mattered. Probably the largest project management insight I had was we needed more concrete goals throughout the semester. This was because the problem we were tackling was unsupervised, and only toward the end did we reevaluate our method and work toward a semi-supervised approach. Additionally, software is difficult partly because of all the different parts that have to work together (web, database, algorithm, etc.). Our priority as a

team was on the algorithm, and so we often were left trying to scrape together the other parts together when it wasn't working or when we needed it.

Acknowledgements

We would like to thank Prof. Laurent El Ghaoui for advising us during the span of our Capstone project. We would also like to thank StatNews research group we are a part of for giving constructive advice and technical support.

Works Cited

Bosker, Bianca. Apple-Samsung Lawsuit: What You Need To Know About The Verdict. http://www.huffingtonpost.com/2012/08/24/apple-samsung-lawsuit-verdict_n_1829268.html, 2012: Huffington Post.

Google Patent Search. http://www.google.com/advanced_patent_search, 2015: Google Patents

High, Peter. Gartner: Top 10 Strategic Technology Trends For 2014 http://www.forbes.com/sites/peterhigh/2013/10/14/gartner-top-10-strategic-technology-trends-for-2014/, accessed February 15, 2015.

Masson, Patrick. Open Source Inititative: 2014 Annual Report. http://opensource.org/files/2014AnnualReport_0.pdf, 2014: OSI.

Page, L. Generic Method for node ranking in a linked database. http://www.google.com/patents/US6285999, 2001: Google Patents.

Carpenter, Brian; Hart, Judith; Miller, Jeannie
Jump starting the patent search process by using subject-oriented databases. http://www.engineeringvillage.com/search/doc/abstract.url?&pageType=quickSearch&searchtype=Quick&SEARCHID=152b761cM93aeM4669Mac30Me7750989c781&DOCINDEX=7&database=1&format=quickSearchAbstractFormat&tagscope=&displayPagination=yes, accessed February 11, 2015.

Carter, Brittany
IBISWorld Industry Report: Trademarks & Patent Lawyers & Attorneys in the U.S. http://clients1.ibisworld.com/reports/us/industry/default.aspx?entid=4809, accessed February 11, 2015.

Google Patent Search
2011 Advanced Patent Search. http://www.google.com/advanced_patent_search, accessed December 1, 2014.

High, Peter
Gartner: Top 10 Strategic Technology Trends For 2014 http://www.forbes.com/sites/peterhigh/2013/10/14/gartner-top-10-strategic-technology-trends-for-2014/, accessed February 15, 2015.

Costowl.com
http://subscriber.hoovers.com/H/industry360/overview.html?industryId=1063, accessed February 11, 2015.

How Much Does a Patent Lawyer Cost?

http://www.costowl.com/legal/lawyer-patent-cost.html, accessed March 11, 2015.

IBM
2014 IBM Research. http://www.research.ibm.com/careers/, accessed February 15, 2015.

Laurent El Ghaoui, Guan-Cheng Li, Viet-An Duong, Vu Pham, Ashok Srivastava, Kanishka Bhaduri.
2011 Sparse Machine Learning Methods for Understanding Large Text Corpora. Proc. Conference on Intelligent Data Understanding.

Lawyers.com
2014 Find a Patents Lawyer or Law Firm by State.
http://www.lawyers.com/patents/find-law-firms-by-location/, accessed December 1, 2014.

McCarthy, Jerome E.
Basic Marketing. A Managerial Approach, 1964.

Mintel. (2014). Pharmaceuticals: The Consumer - U.S. - January 2014.
http://academic.mintel.com/display/692963/, accessed February 11, 2015.

Mintel. (2014). Coffee - U.S. - August 2014. http://academic.mintel.com/display/713740/, accessed February 11, 2015

Nolo LAW for ALL (NOLO)
2014 Patent Searching Online.
http://www.nolo.com/legal-encyclopedia/patent-searching-online-29551.html, accessed December 1, 2014.

Porter, Michael E. "The Five Competitive Forces That Shape Strategy." Special Issue on HBS Centennial. Harvard Business Review 86, no. 1 (January 2008): 78–93.

Samuel Smith, Jae Yeon (Claire) Baek, Zhaoyi Kang, Dawn Song, Laurent El Ghaoui, and Mario Frank.
2012 Predicting congressional votes based on campaign finance data. In Proc. Int. Conf. on Machine Learning and Applications.

SBA
Small Business Trends, https://www.sba.gov/offices/headquarters/ocpl/resources/13493, accessed February 15, 2015.

United States Patent and Trademark Office (USPTO)
2014 Patents for Inventors: Patents - November 2014. http://www.uspto.gov/inventors/patents.jsp, accessed December 1, 2014.

United States Patent and Trademark Office (USPTO)

2013 Glossary - September 2013. http://www.uspto.gov/main/glossary/, accessed December 1, 2014.

David M. Blei, Andrew Y. Ng, Michael I. Jordan.
Latent Dirichlet allocation. Journal of Machine Learning Research, 3, 993-1022, 2003.

Marcus A. Butavicius, Michael D. Lee.
An empirical evaluation of four data visualization techniques for displaying short news text similarities. International Journal of Human-Computer Studies, 65, pp. 931-944, 2007.

Xi Chen , Yanjun Qi , Bing Bai , Qihang Lin , Jaime G. Carbonell
Sparse Latent Semantic Analysis
SIAM International Conference on Data Mining (SDM), 2011.

Tibshirani, R.
1996 “Regression shrinkage and selection via the lasso.”
Journal of the Royal Statistical Society. Series B (Methodological), Volume 58, Issue 1, 267-288.