Comparing Patent Similarity to Detect Potential Competitors

Saurabh Jaju, Jeffrey Hsu, and Cameron Bell
Dec 2017 - w266 final project

Abstract

Patent examiners and patent attorneys closely examine claims to determine whether a proposed invention is patentable and whether a patent has been infringed [1]. Comparing patent similarity could lead to a more interesting analysis if it is used to detect potential new entrants and future competitors in industry based on the similarity of patents they file. We present an application which compares patent similarity by creating a pipeline to preprocess the claims text, and by using various techniques to create vector forms using advanced techniques like LSI, LDA and Doc2Vec which can be used for similarity analysis. The proposed application maps the relative similarity and overlap of each patent which can be used for competitor analysis to develop business strategy.

1. Introduction

Companies make strategic decisions based on in-depth industry analysis. Analyzing industry structure and how industry would evolve is essential to determining if an industry is going to be profitable over time; thus, it is at the base for any company to evaluate if it should enter the industry. This analysis is based on Porter's Five force model. One of the forces of this model is competitor analysis and one of the key aspects which keep a company profitable are the unique products and services it provides which are difficult for any company to imitate. Thus, to make its products inimitable companies file patents. But filing a single patent may not guarantee inimitability, thus companies file multiple patents to hide the most relevant patent. This makes identifying future competitors in the industry particularly hard. This problem is particularly interesting because, if future new entrants in the industry can be identified using patent similarity comparison, such information would be useful for taking strategic decisions and also to assess the potential profitability of the industry. Thus such research would be essential to both the companies and the investment firms in the market.

In this project we tried to develop a analysis framework which would map various companies at distances based on the similarity of patents filed by the company. It consists of two stages: 1) develop a framework to find similar patents and 2) using patent similarity to infer potential market competition. Our work in this paper targeted towards first stage. Concretely, we used the claims text to capture the characteristics of patents and further compare their topic similarity.

2. Background

The problem of working with patent data to compare similarity can be divided into two parts. Part 1: extract patent information from the large corpus of text data. Part 2: compute various measures of similarity to compare semantic similarity. Extensive research has been done for developing efficient patent retrieval system, patent classification, clustering, categorization and keyword retrieval [2,3]. Large body of work dealing with patent claims analysis also exists[1,4,5]. On highlevel, patent similarity can itself be divided at two levels: content oriented level and formal oriented level [6].

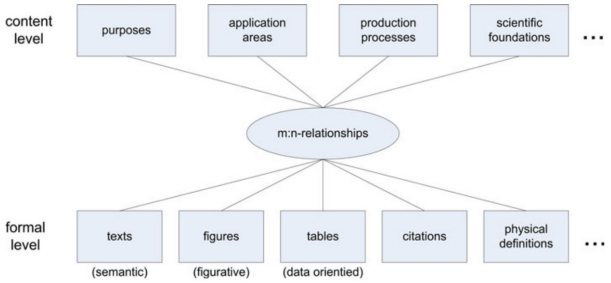

Figure 1
Two levels of Patent Similarity[6]

Patent similarity at the formal level can be derived and evaluated using NLP and the large text corpus. Though extensive research for patent similarity detection based on text, citation, claims and figures already exists, in this application we base our similarity model on claims text alone.

Another area of research which is relevant relates to the choice of the similarity coefficient to be used. Multiple similarity coefficients such as Jaccard, Sorensen, Sokal & Sneath 2, Kulczynski 1, Kulczynski 2, Cosine and Inclusion can be used to compare similarity of patents. In case of comparing patent claims, it is useful to use a two sided similarity coefficient like Jaccard, inclusion or cosine [6].

For this application, we choose to compare patents at the formal level using claims text to capture patent characteristics and using cosine similarity as the coefficient to compare the patents.

To compare similarity, patent documents need to be represented in a common vector space in order to allow computing the similarity measures. Traditional information retrieval approach relies on bag-of-words set up. Different methods have been proposed, either by adjusting the weighting or dimensionality reduction. Its major advantage is robust performance even on smaller datasets. However, the bag-of-words assumption filters out word position and phrase level information, which is a drag back if documents contain similar tokens but are semantically different. Another branch of methods maps documents to continuous vector space. These methods can capture phrase level and word position information. They generally work well on large corpus. In this project we explored both document representation methods.

3. Design and Implementation

3.1 Data

This application utilizes data available from USPTO. All patents are filed in English. The full text data is available in both XML and CSV format. USPTO has patent data since 1970s available on the website which includes data of 7 million patents granted and ongoing patent applications. Given the time and processing constraints, we used 100,000 granted patents as our dataset for further analysis. Each patent record contains detail regarding:

- Applicant information
- Abstract
- Claims
- Text describing diagrams
- References

The project uses full text patent claims as training and evaluation data. We did not include other numeric or handcrafted features. This is to avoid over relying on non-textual data for machine learning performance and also to fully exploit the semantic similarity between patent claims. The claims full text dataset alone takes up around 30 gigabyte. Raw data is formatted into pairs of unique patent ID and full claim texts before other text preprocessing and model building stages.

The raw claims texts are preprocessed prior to any modeling. The documents are separated into word token lists. Each token are lower-cased and stemmed so that frequent word variations are consolidated. Stopwords like ‘the’, ‘of’, ‘for’ are removed.

3.2 Analysis Pipeline

The analysis methods can be generally separated as 2 types: Bag-of-Words (BoW) and Continuous Vector Representations. Bag-of-Words type of models does not take into account of word positions or phrase information (n-gram). It computes token frequency in each document. Thus, documents with similar word composition are close in meaning in this model. We have created several different document representations based on Bag-of-Words approach. Continuous vector representations takes the text and map that to a single vector. In this project we focus on exploring Doc2Vec [9] method for continuous vector representations. With document vectors computed, we can use cosine similarity to determine the patent similarities. Note that since we only use claims text as our input data, there is no golden answer we can evaluate the document vectors on, or perform grid search in a supervised way. Instead, the quality of document vectors (whether they correctly put similar patents close in vector space) are assessed empirically and visualizations on lower dimensions. Having a systematic way of evaluating document vectors and further associate with market competition will be discussed in our future work. Below we describes the different models used and

their implementation detail.

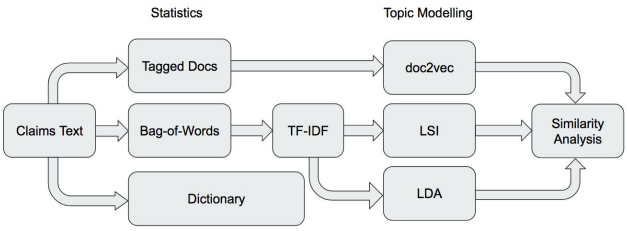

Figure 2
Our processing pipeline

TF-IDF

Term Frequency - Inverse Document Frequency (TF-IDF) is our basis for all following models based on document word counts. We first build the dictionary by iterating through the whole corpus. The dictionary maps words to token IDs, thus speed up computing on keys. Extremely frequent or very rare words are removed from the dictionary trim down our vocabulary. With 100K patents, trimming gives a vocabulary of near 2M unique tokens. Based on the dictionary, each document is transformed into a Bag-of-Words, with a token ID and word count pair format. This word count can then get us the term frequency (TF). Each term frequency is then weighted by inverse document frequency (IDF) to emphasize important words that can characterize the document. The resulting TF-IDF corpus is a matrix of shape (vocabulary size , document count).

LSI and LDA

Latent Semantic Indexing (LSI) and Latent Dirichlet Allocation (LDA) are techniques to get document vectors built on top of our TF-IDF corpus. Essentially, they perform dimensionality reduction on our TF-IDF matrix to get a smaller matrix with a few important topics. The difference being LDA considers the likelihood of each topic. The matrix decomposition is done similar to PCA (principal component analysis) and outputs a matrix of shape (topic count , document count). Each document is represented as the vector of those few topics.

Doc2Vec

Doc2Vec is similar to Word2Vec [12], but difference is it tries to represent the whole text document as a single vector, which embodies the text semantics. The training process concatenates the document vector with context word vectors and tries to predict the target words in the documents. This is similar to CBOW (continuous Bag-of-Words) approach where context word vectors are used to predict target words, but in this case the document vector is part of this. As the context window slides through the paragraphs, document vector is being updated to capture the characteristics of the whole document. In the original paper by Mikolov et. al, document vectors are computed on paragraph level. In our work, we make the whole patent claim text as the document vector, so instead of paragraph characteristics, it captures text characteristics of the whole patent.

3.3 Implementations

Implementations of the models are mainly based on Python's Gensim package[13]. Its simply built in API supports preprocessing corpus, building dictionary and Bag-of-Words corpus, TF-IDF/LSI/LDA corpus and doc2vec training. One of the key implementation consideration, given the size of our patent corpus, is to process documents one by one with Python generators. This avoids loading the full text set (30 GB) into memory. Luckily, all Gensim classes we needed supports generator as input corpus and line-by-line processing during runtime. Throughout the modeling stages (e.g. BoW to TF-IDF to LSI), we materialized the model and the resulting corpus on disk. This allows reusing the weights in later stage and avoids losing results in case of crashing cloud instances. Lastly, the models can be updated given new documents as new patents are released. The Bag-of-Words approach just needs to update the dictionary and the word counts per document. For Doc2Vec, adding a new document will append a new vector to the end of the matrix. Fixing the word vectors while training on the newly added document will get us the new document vector.

4. Results and Discussion

We successfully built an application that receives a patent number as input, and returns the n most similar patents by patent number (see Figure 3).

Figure 3

This is one giant step toward our original (and ultimate) goal (see Figure 4):

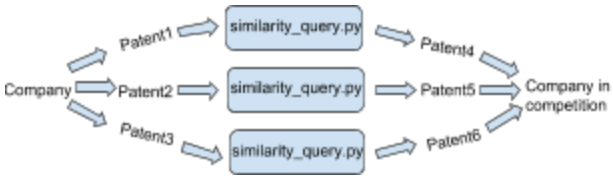

Figure 4

We want to be able to find similar companies based on their patents, and finding similar patents is an essential step in that process. Unfortunately, this second phase would require finding and formatting another dataset, for which we have not had enough time.

Latent Semantic Indexing (LSI)

Our LSI model helps us to demonstrate the most similar patents as shown below:

```
> python demonstration.py
> patent_number = 3930278
> 10 most similar patents:
Patent 3930278: 1.000
Patent 3982691: 0.850
Patent 3978539: 0.827
Patent 4024637: 0.825
Patent 4026366: 0.814
Patent 3938206: 0.801 ...
```

The first one, at 100% most similar, is the original patent. But the others can be used to find competitors in the market.

The previous example generated its results with an LSI topic model with 100 topics. Here are a few sample topics, with words scaled according to their weight in the topic [14]:

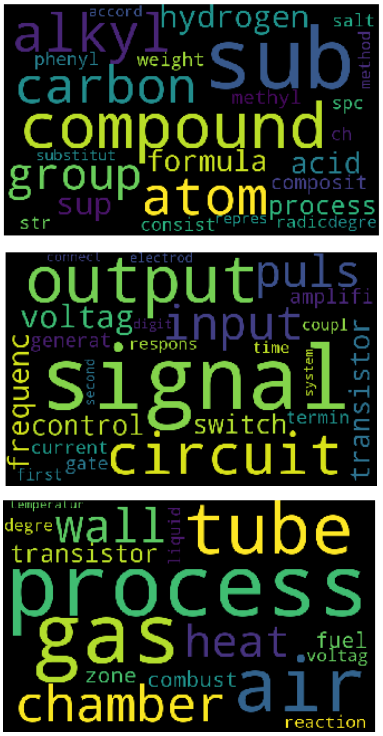

Figure 5

Even with a small dictionary of only 17658 tokens (after filtering), the topics are coherent.

Latent Dirichlet Allocation (LDA)

The LDA model allowed us to use an interactive visualization called pyLDAvis[15]. It depicts the topics in two dimensions using principal component analysis (PCA), and gives an overview of the prominent tokens in each topic, among other features. This visualization is hosted here temporarily. A screenshot has also been included below (see Figure 6):

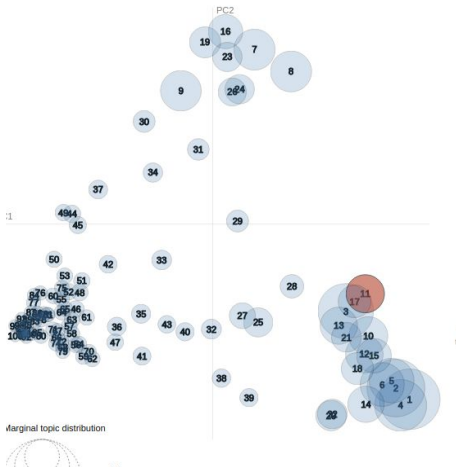

Figure 6

Using a corpus of patent claims gives way to clear and intuitive topics, because patents are so specific about the materials and technologies involved.

Doc2Vec

An evaluation check on inferred document vector versus all set of similar documents performs poorly with Doc2Vec. The results shows the inferred vector from the model is not even able to recognize itself as similar documents. There are several potential reasons for this:

1. In Doc2Vec's original work, they trained paragraph vectors other than our approach of the full patent claim document. Full document may not matter much on predicting single target words. The training stage may be updating mostly on the word vectors, not much on the document vectors.
2. We used embedding size of 160 for each document. This may result in a very sparse space for our 100k training documents. Ideally we would want to use the full patent corpus and perform a grid search on which embedding size works best.
3. Not enough training epochs. For each document, we trained 50 rounds. That could be insufficient for updating the parameters.

5. Conclusion

We developed a simple, effective patent similarity comparison framework with the following workflow:

Raw claims text → Dictionary → Bag-of-Words

→ TF-IDF → LSA/LDA

6. Future Work

As there is no standard answer for document similarity, we are not able to evaluate different vector representation methods in this project. One way that can help evaluating document vector quality is by using patent tags. By checking if similar documents suggested by our model are overlapping in their tags given by USPTO, we can assess the relative performance of each modeling methods. Also, we would like to scale the number of patents analized from 100k to all 7 million available granted patents.

Other extensions will help infer market competition. Additional functionality could be built on top that would compares companies to companies, not just patents. The same system while analyzing multiple patents can be developed and used for Inter- and intra-industry comparison of patents. Another interesting application could be to analyze timelines of similar patent filings and eventual entry into the industry. Predictive systems can be developed based on stock market price comparison, patent similarity scores and product portfolios. Each of these additional features can make this application an ideal tool for competitor analysis and to analyze threat of new entrants in the industry which in turn can be used to analyze the profitability of the industry and aid in development of competitive strategy for firms.

References

[1] Similarity Analysis of Patent Claims Using Natural Language Processing Techniques. Kishore Varma Indukuri, Anurag Anil Ambekar, Ashish Sureka. International Conference on Computational Intelligence and Multimedia Applications 2007

[2] Introduction to the special issue on patent processing. A. Fujii, M. Iwayama, N. Kando. Inf.Process. Manage., 43(5):1149–1153, 2007.

[3] Text mining techniques for patent analysis. Y.-H. Tseng, C.-J. Lin, and Y.-I. Lin. Inf. Process. Manage., 43(5):1216–1247, 2007.

[4] Natural language analysis of patent claims. S. Sheremetyeva. In proceedings of the ACL-2003 workshop on patent corpus processing, pages 66–73, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[5] Patent claim processing for readability: structure analysis and term explanation. A. Shinmori, M. Okumura, Y. Marukawa, and M. Iwayama. In Proceedings of the ACL-2003 workshop on Patent corpus processing, pages 56–65, Morristown, NJ, USA, 2003. Association for

Computational Linguistics.

[6] Measures for textual patent similarities: a guided way to select appropriate approaches. Martin G. Moehrle Scientometrics 2010

[7] Towards content-oriented patent document processing. Leo Wanner et. al. World Patent Information 2008

[8] Patent-to-Patent Similarity: A Vector Space Model. KA Younge. SSRN Papers

[9] Distributed Representations of Sentences and Documents. Quoc Le and Tomas Mikolov. Google Inc. 2014

[10] Deep Sentence Embedding Using Long Short-Term Memory Networks: Analysis and Application to Information Retrieval. Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, Rabab Ward. cs.CL 2016

[11] Skip-Thought Vectors. Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, Sanja Fidler. 2015

[12] Distributed Representations of Words and Phrases and their Compositionality. Tomas Mikolov et. al. 2013

[13] Software Framework for Topic Modelling with Large Corpora https://github.com/RaRe-Technologies/gensim Rehurek, Radim, and Sojka, Petr. 2010

[14] Word Cloud: A little word cloud generator in Python. https://github.com/amueller/word_cloud Mueller, Andreas. 2017

[15] pyLDAvis: Python library for interactive topic model visualization. https://github.com/bmabey/pyLDAvis Mabey, Ben 2014