Analyzing Patent Data

Mandeep Kaur
(ID- 40059801)
Department of Computer Science
and Software Engineering
Concordia University
k_ndeep@encs.concordia.ca

Jasraj Singh Bedi
(ID- 40046931)
Department of Computer Science
and Software Engineering
Concordia University
j_ed@encs.concordia.ca

Abstract- Traditionally, patent information searches are carried out before filing patent applications or during the planning and preparation of patent lawsuits, if at all, as part of the drafting process. In recent times, this traditional micro-level use of patent information has evolved into a much more strategic use of patent information. Now, businesses are more interested in knowing their competitors, on which new technologies they are working or planning to introduce so as to formulate better strategies to expand their own business.

This project aims at analyzing Google's public patent data and getting useful insights from it such as similar patents, top potential competitors (companies or assignees or inventors of the patent) and technologies in trend using both unsupervised and supervised learning methods by processing the data. We analyze the patents' title, abstract, description in order to create features to better train the model in python (mostly using PySpark's and Scikit learn's APIs). Using these features, we will cluster the patents together using k-means and analyze their similarity metrics followed by classifying them using random forest classifier to predict the class of newly encountered patent in the data.

I. Introduction

A patent is an exclusive right to use a new technological solution; it is considered as one of the intellectual property's strongest rights. From the point of view of the patent owner, it constitutes a resource and a potential market value. It's one of the stages of the innovation process in the economic dimension. For research and development activities a patent is a crowning point in the scientific or statistical sense. The properties of the description of a patent and the exclusive right itself cause a situation in which patent information is a bridge between the results of the research and development (R&D) and bridge between their possible economic use.

Motivation: There are several approaches that, depending on the circumstances, can reveal patent or patent application information about a competitor. Information about a competitor's patent activity may give information about the planned activity of the competitor before it is seen in the marketplace. Doing such analysis can help businesses at least to have an idea of what their competitors are up to and plan or modify their own strategy with new advancements accordingly. This ignited the idea that by doing the intelligent analysis we can actually get this information to benefit businesses. Currently, there has been a lot of work done and in progress in the scientific industry regarding this. We are trying to implement a few techniques to carry out the same idea by analyzing the information extracted from patents.

Objective: To extract and analyze useful insights from patent data such as top potential competitors, technologies in trend and similar or related patents using both unsupervised and supervised learning. Our main focus is to perform exploratory data analysis on Google public patent data to better understand the data, by discovering patterns in it, testing hypothesis and checking assumptions with help of summary statistics and graphical representations first. Then, selecting and building relevant features from the patent's title, abstract and description, such as word embedding using TF-IDF after preprocessing of the data. Our assumption of finding top potential competitors is that if we have similar or related patents in one group then assignees/ inventors in that group must share same specifications in some way or the other (say based on the technical field they are working on), so they all can be potential competitors to each other. Also, the technologies having a maximum count among the group of patents (checked using CPC codes) can be the technology in trend.

II. Materials & Methods

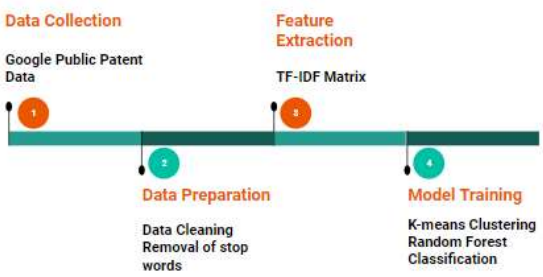

Figure 1 Approach

Data Collection

For the project, we took google patent data set which is a public patent data set and it contains bibliographic information on more than 90 million patent publications with the data from more than 17 countries. We decided to work on the 2018's patent data set.

BigQuery was used to access the dataset. BigQuery is a restful web service which enables an analysis of massively large datasets working together with Google storage. BigQuery is a serverless platform service (PaaS) that may be used with MapReduce. The Design of BigQuery involves providing access to technology an ad hoc query system which is used for the analysis of read-only nested data.

We used BqHelper a BigQuery module, provided in python to fetch the data. The data set consisted of 30 columns out of those we picked most relevant fields needed for our model i.e. 9 samples:

- Publication_number: Patent publication number (DOCDB compatible), eg: 'US-7650331-B1'
- Country_code: Country code, eg: 'US', 'EP', etc
- Title_localised: The publication titles in different languages
- Abstract_localised: The publication abstracts in different languages
- Description_localised: For US publications only, the description, limited to the first 9 megabytes
- Publication_date: The publication date.
- Inventor: The inventors
- Assignee: The assignees/applicants
- CPC codes: The Cooperative Patent Classification (CPC) codes. They have 9 sections representing technological fields, which are used for searching/ classifying the patents.

"A": "Human Necessities",
"B": "Operations and Transport",
"C": "Chemistry and Metallurgy",
"D": "Textiles",
"E": "Fixed Constructions",
"F": "Mechanical Engineering",
"G": "Physics",
"H": "Electricity",
"Y": "Emerging Cross-Sectional Technologies"

Figure 2 Sections in CPC codes

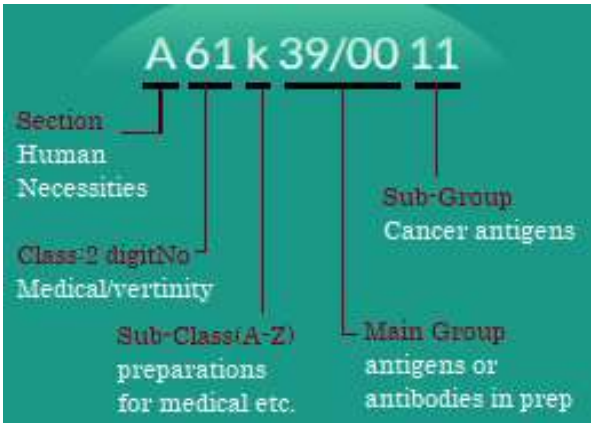

Figure 3 Example of CPC code

and 1000 features, shown in the figure below:

Figure 4 Google Patent Public Dataset

Data Preparation

The most important part of our application is the data preprocessing stage, it is mainly divided into the following parts for our algorithm:

Data Cleaning: Data we collected using BigQuery was stored in tables and of the type record with nested fields (example of one record is shown in the figure 5), so we had to convert the data fields into actual lists or dictionaries in python and extract the relevant raw data from them. Therefore we selected abstract, title and description columns and removed punctuations and stop words using NLTK (The natural language toolkit) which provides a list of English stop words.

title_localized	RECORD	REPEATED
title_localized.**text**	STRING	NULLABLE
title_localized.**language**	STRING	NULLABLE
abstract_localized	RECORD	REPEATED
abstract_localized.**text**	STRING	NULLABLE
abstract_localized.**language**	STRING	NULLABLE

Figure 5 Example of Nested Record Fields in Dataset

Data Transformation: Raw data cannot directly be used for clustering or classification.

This step includes the creation of our features from raw textual data by tokenization them first then creating a feature matrix by assigning weights to each token of a word.

Frequency Matrix: The tokenized words are used as features against the patents' publication_number. Frequency of occurrence of each token in the given document is stamped as the value of the token for that document. This gives us a discrete feature set of tokens.

TF-IDF Matrix: For a selection of the principal words we used TF-IDF algorithm to get values for respective feature sets. TF-IDF is term frequency (the Frequency of the

Term in the document) multiplied by the Inverse Document Frequency.

pub num	accompanying	accordance	according
US-2001040298-A1	0.001009510837231...	0.0	0.013942460964556003
US-2002037896-A1	0.0	0.001717422991359...	0.016489601139623306
US-2002055159-A1	0.0	8.765549063751532E-4	0.001618484928584...
US-2002095050-A1	0.0	0.0	0.030350213992690462
US-2003052383-A1	0.002018845291143...	0.0	0.026414986623573177
US-2003052813-A1	0.001124542035742...	0.0	0.008991729430214146
US-2003099932-A1	0.0	5.101002503098 64E-4	0.00263720021567408
US-2003118999-A1	0.0	0.0	0.003136759733571136
US-2004038943-A1	0.0	0.0	0.030147466904678115
US-2006189588-A1	0.0	0.0	0.007329744212343488
US-2007066615-A1	0.0	0.003692910196711...	0.001363729631201624
US-2003195118-A1	0.0	0.0	0.00398677077152442 5
US-2003215833-A1	0.001239962434963...	0.001220376547787...	0.005407944292383587
US-2003219428-A1	0.0	0.0	0.002160257527911836

Figure 6 TF-IDF Matrix

The Tf-IDF factor is calculated as below:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

tf_{ij} = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

Feature Selection: This is the final step of data preprocessing. It includes removal of unwanted features from the feature sets. The selection of unwanted features was done on the basis of whether the occurrence of the token happened in more than a threshold percentage of documents, as the feature is relevant only if it occurred in a sufficiently high number of documents. The sparsity threshold factor was set to 0.4 i.e. min_df in sklearn tfidfvectorizer with some other set of parameters as follows:

- min_df=0.4 (filters all words having document frequency less than 0.4)
- smooth_idf=True
- lowercase=True
- analyzer='word'
- use_idf=True

Unsupervised Learning:

Unsupervised learning is usually applied to datasets with unlabelled data. The most common unsupervised learning technique is clustering.

K-Means Clustering

We decided to go with k means clustering, as the dimensions were very high, and k means clustering has an advantage that it computes rapidly when the data is in high dimension. The distance formula used was Euclidean distance. Initially, some data points are selected(these are known as initial centroids) at random or by using an algorithm, these data points selected are equal to the number of clusters. Then for each of the remaining data point, the Euclidean distance is calculated from the initial centroids and the data point belongs to the initial centroid with which it has the least distance. After each data point is assigned to a centroid, this concludes the first iteration and the centroids are calculated again by taking the mean of the cluster. These iterations are carried on until the centroid of the previous and current centroid are the same, other words until the centroids do not change. Now, the question arises how to find out how many clusters we should make.

We can find the k value from the elbow method. The idea is to run the k means for a set of value (say 1 - 10) and for each k value, SSE is calculated (Sum of squared errors). Then the k values are plotted against the SSE values, if the line looks like an arm graph, then the elbow gives the best value of k.

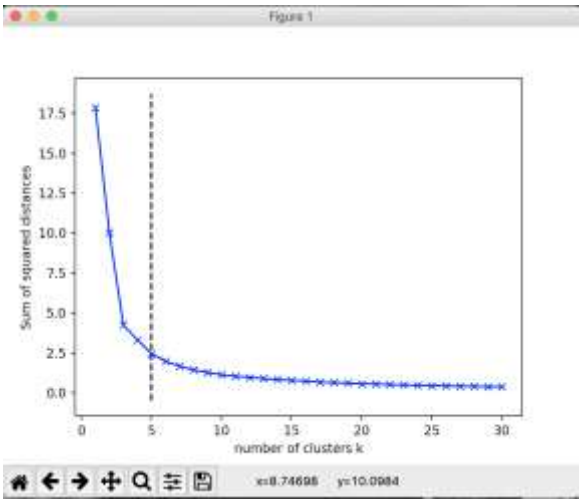

Figure 7 Elbow graph - Elbow point is represented by dashed line

For our data, we got 5 as the value of k. Thereby, 5 clusters were made. The silhouette score measures how similar an object is to its own cluster compared to other clusters. The silhouette score ranges from −1 to +1, where a high value indicates that the data point is well matched to its own cluster and poorly matched to the rest of the clusters. All the 5 clusters were analysed and scrutinized**.** The most dominant technology was found by examining the CPC codes and the patents were filtered based on the dominating technology inside the cluster and their assignees were plotted. We can say that these assignees are potential competitors in that field (see the diagrams below).

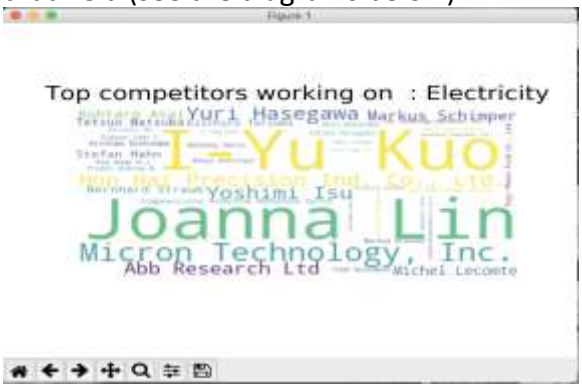

Figure 8 Word cloud

Principal Component Analysis

Principal component analysis (PCA) simplifies the high-dimensional data complexity while keeping trends and patterns. It does this by transforming the data into fewer dimensions, which act as features summaries. To minimize the total distance between the data and their projection to the PC, the first PC is selected. We also maximize the variance of the projected points, $\sigma 2$, by minimizing this distance. Similarly, the second (and subsequent) PCs are selected with the additional requirement to be uncorrelated to all previous PCs. After applying TF-IDF, we got feature embedding with 1440 dimensions, for which visualization was quite challenging. So, we used PCA to reduce that data into 2-dimensions for better visualization of clusters obtained through k means clustering.

Supervised Learning: Now, the data is converted into supervised learning data, since we have labels on the data and supervised learning algorithms can be applied. We turned our unsupervised learning problem to supervised learning by assigning labels (from 0 - 5) to the clusters (output of k means clustering) of patents and used this data as an input for classification of our problem.

Random Forest Classification

Random Forest algorithm is a supervised and ensemble learning method for classification. We can see it from its name, somehow creating a forest and randomizing it (Each tree makes a prediction and most predicted value is taken as the answer for the final prediction). There is a direct relationship between the number of trees in the forest and the results it can achieve:

the greater the number of trees, the better the result. However, we need to balance out the computing cost with the number of trees. Furthermore, creating the forest is not the same as building the decision with an approach to information gain or index gain.

$$\sigma = \sqrt{\frac{\sum_{b=1}^{B}(f_b(x') - \hat{f})^2}{B-1}}.$$

Figure 9 Standard Deviation

We divided the labelled data into 70% training and 30% test set, to apply this classification method. We trained our classifier, so that, it should be able to predict the correct class of a new patent arrived from test data. We tried out our classifier with a different number of trees and the variation in accuracy was seen to be constant after t = 100 (t = number of trees). So, we choose 100 trees for classification and using 10-fold cross-validation technique, we were able to achieve accuracy of 81% by the random forest classifier.

III. Results

Exploratory data analysis: We explored our dataset by performing exploratory data analysis. We determined the count of a total number of patents that lies in each of the CPC sections depicting top technologies in our dataset of 1000 features to analyze later the formulation of clusters based on these technologies. Also, to know how well the patent data is diverse in technologies. It is represented by a bar chart shown below:

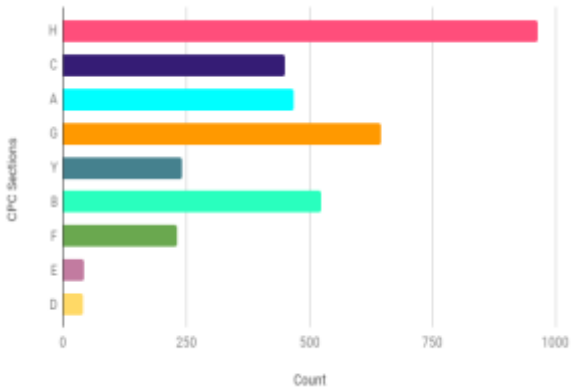

Figure 10 Total number of patents in Sections of CPC

Descriptive data analysis

Using BigQuery, we managed to extract CPC codes from the entire data set of 1TB corresponding to the set of 10 publication numbers taken as input. We extracted publication number of patents that share same cpc codes and those who do not share cpc codes. We found 229 similar patents to our input list from google's research patent dataset. PCA was used to reduce the dimensions and visualize the data as shown in the figure

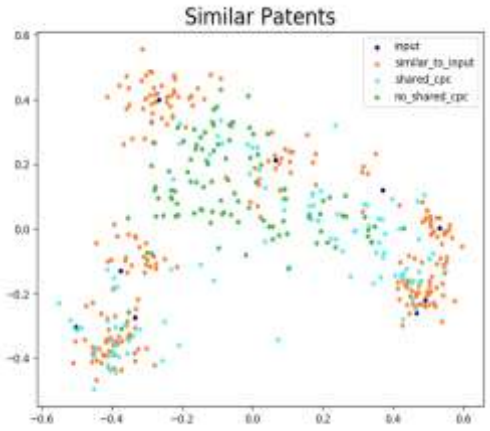

Figure 11 Similar Patents

K-Means Clustering

We picked two principal components using PCA for our patent clusters and generated a scatter plot to visualize the high dimensional data in 2-D representation.

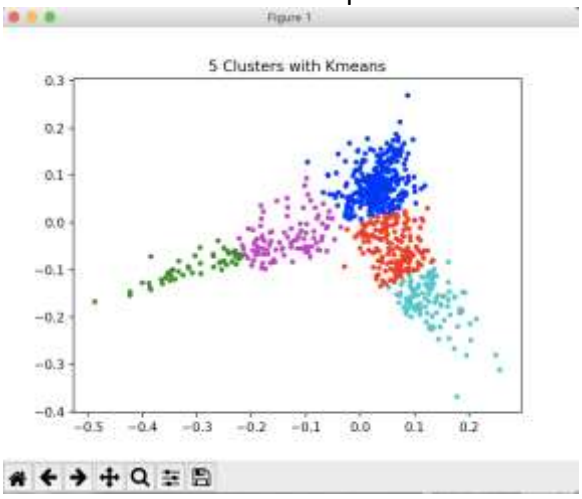

Figure 12 Clusters

By analyzing the clusters, we assumed that if these patents are clustered together then their assignees must be sharing some similar or related specifications, so we plotted word cloud of assignees, representing them as top potential competitors from two random clusters. Also, we found a different set of technologies in the patent clusters and represented the ones with the highest count in the cluster.

Figure 13 Word cloud

Random Forest Classifier

Using k-fold cross-validation, we got the result given below:

```
Test Error = 0.182573
accuracy 0.8174273858921162
RandomForestClassificationModel (uid=RandomForestClassifier_3dd733255c57) with 100 trees
```

IV. Discussion

We are able to find potential competitors, which can benefit a company by knowing its competitor. Also, by analyzing similar CPC codes (matching the first character of the CPC code) in the cluster; we predicted in which cluster does a new patent belongs to, using classification. Our solution can be seen on github link:

https://github.com/kjbedi/PatentDataAnalyser

As discussed in the result analysis, we generated word clouds for two random clusters representing top potential competitors clustered together. Technologies in trend were found to be Electricity, Chemistry and Metallurgy which had a maximum count in the clusters. Although if we compare this result with our exploratory analysis done initially, we analyzed that, other than Electricity and chemistry, physics and Operations & transport also had the second and third highest number of patents in our dataset but after clustering we found only two dominant technologies(can be seen in the figure). This could be one possible limitation of our model. But our random forest classifier did very well on its prediction with an accuracy of 81%.

Future Work: The results are restricted to 1000 patents, so in future we would like to work consider more features for better prediction and analysis. The dates can be

used to indicate the technology trend between specific time periods. The country code (Which basically is the country name) can be used in the feature matrix by taking its longitude and latitude. We would like to do the analysis on complete patent dataset using multiple nodes which would yield more accurate results.

References:

- https://en.wikipedia.org/wiki/BigQuery
- https://medium.com/@Synced/how-random-forest-algorithm-works-in-machine-learning-3c0fe15b6674
- https://www.alexejgossmann.com/patents_part_1/
- http://www.ericksonlawgroup.com/law/patents/patentfaq/how-do-i-find-my-competitors-patents-or-patent-applications/
- https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15