ONLINE APPENDIX

Appendix 1: Description code

The code is written in Java and relies on Java standard libraries. We describe in general terms the different processes conducted and the data files created in order to compute the similarities between patents. We start the process in a working directory `[WD]` where we store the original data file `patents_raw.csv` and the subsequent generated files. `Patents_raw.csv` is a raw comma separated value file containing one row for each patent in the following format:

```
[Patent Number],[Filing Year],[Title],[Abstract]
```

As an example of the format, we show a patent in the file:

3631874,1970,FLUIDIC OVERSPEED SENSOR FOR A POWER TURBINE,A fluidic sensor having two parallel frequency-to-analog circuits whose output is summed to provide an error signal is disclosed.

Starting with the raw patent data file, we carry out the following steps:

1. We read each row from the data file `patents_raw.csv` and we do the following:
    a. Parse the row per sections using the comma separator.
    b. Concatenate the title and the abstract sections as a single content string.
    c. Lowercase the content string.
    d. Tokenize the content with `w[[-]w&&[^ ]]+w]` as the regular expression to extract keywords. This will match as keywords strings that are composed of any combination of characters[1] except _, and allows the hyphen (-) in order to consider compound keywords such as chemical names (e.g., bi-color, 4-di-n-oxide). Each unique keyword from the extraction is added in a list.
    e. Sort the list in ascending alphabetical order and clean it by removing stop words[2], keywords formed only by numbers (e.g., 1974, 1-3-4-4) and keywords with one character.
    f. Output the clean list of keywords as a row in the file `patents bow.txt` in the following format:
    `[Patent Number],[Filing Year],{List of Keywords}`
    where the keywords in the list are separated by a space.
    For example, extracting the keywords for the patent shown above, we have:

    3631874,1970,circuits disclosed error fluidic frequency-to-analog output overspeed parallel power provide sensor signal summed turbine

2. We read each row from the file `patents bow.txt` and we execute the following.
    a. Parse it per sections using the comma separator and take the list of keywords.

[1] See http://www.regular-expressions.info/shorthand.html
[2] See https://gist.github.com/shuson/b3051fae05b312360a18

    b. Aggregate the unique keywords in a list to form a single vocabulary for the whole patent corpus, counting in how many rows (patents) each keyword occurs.
3. After reading all the rows and forming a general vocabulary, we remove from the vocabulary keywords that occur in only one patent. Keywords appearing in only one patent are uninformative about similarity with other patents and are likely to be spelling errors.
4. We output in the file `vocabulary.txt` the list of keywords, one keyword per line.
5. We load the formed vocabulary in memory, we read each row from the file `patents_bow.txt` and then we carry out the following.
    a. Parse it per sections using the comma separator and take the list of keywords.
    b. Eliminate from the list of keywords the ones that do not appear in the vocabulary.
    c. Output the clean list of keywords as a row in the file `words.csv` in the following format:
    `[Patent Number],{List of Keywords}`
    For example, after cleaning the list of keywords from the patent shown above, we have:

    3631874,circuits disclosed error fluidic frequency-to-analog output overspeed parallel power provide sensor signal summed turbine

    d. Output the patent number and the filing year as rows in two individual files: patents_numbers.txt and patents_years.txt, respectively. There is a correspondence 1 to 1 between rows in files words.csv, patents_numbers.txt and patents_years.txt.
6. Using the files `patents_numbers.txt` and `patents_years.txt` we split the file `words.csv` per filing year, and thus create a set of files `patents_[year].txt`, in order to compute the similarities among patents in the same year.
7. For each file `patents_[year].txt` we take each row as a focus patent A, we parse it using the comma separator and take the list of keywords, and then we execute the following
    a. Iterate over all the rows of the file `patents_[year].txt`, except the row of patent A, with each subsequent row being patent B, we parse it using the comma separator and take the list of keywords.
    b. Compute the Jaccard coefficient between patents A and B: $J(A,B) = |A \cap B|/|A \cup B|$, where A and B are the set of keywords of the corresponding patent.
    c. Store the pair of patents with their correspondent coefficient in the file `jaccard_[year].txt`, with the format:
    `[Patent Number A],[Patent Number B],[Jaccard Coefficient]`

In case we would wish to compute similarities between patent portfolios, we would only need to concatenate the keywords from every patent in each portfolio and assign a portfolio number in a file `words_portfolio.csv`. We could then proceed in the same manner as from step 6.

Appendix 2: Description data files

The following data files are available from https://dataverse.harvard.edu/dataverse/patenttext

The first data file "***words.csv***" contains one row and two columns for each patent. The first column contains the patent number, the second column contains the set of unique and cleaned keywords separated by a space and ordered alphabetically. This database can be used to calculate the similarity between any pair of patents, or between two groups of patents by aggregating the keywords at the group level and calculating the similarity between the two groups.

The second data file "***closest match.csv***" contains for each patent the closest text-matched patent filed in the same year (with a minimum Jaccard index of 0.05). It consists of three columns: The first column contains the patent number of the baseline patent, the second column contains the patent number of the closest text-matched patent filed in the same year, and the third column contains the Jaccard index based on the overlap in keywords between the two patents. This database can be used to select a case-control group for a given set of patents.

The third data file "***200 closest matches.csv***" is constructed in an identical way as closest match.csv but contains the two hundred closest matches (with a minimum Jaccard index of 0.05) filed in the same year. It can be used to select a case-control group for a given set of patents conditional on a number of additional criteria that might exclude the single closest patent. For instance, the control patent needs to be assigned to a different firm or to a different set of inventors. Due to the size of this dataset, it is split into ten different datasets (e.g., *200 closest matches 1 out of 10*).