

UC Berkeley

UC Berkeley Previously Published Works

Title

Machine learning and natural language processing on the patent corpus: Data, tools, and new measures

Permalink

<https://escholarship.org/uc/item/48z2p287>

Journal

Journal of Economics & Management Strategy, 27(3)

ISSN

1058-6407

Authors

Balsmeier, Benjamin
Assaf, Mohamad
Chesebro, Tyler
et al.

Publication Date

2018-09-01


DOI

10.1111/jems.12259

Peer reviewed



Machine learning and natural language processing on the patent corpus: Data, tools, and new measures

Benjamin Balsmeier¹ | Mohamad Assaf^{2,3} | Tyler Chesebro⁴ | Gabe Fierro⁴ |
 Kevin Johnson⁴ | Scott Johnson⁴ | Guan-Cheng Li² | Sonja Lück⁵ | Doug O'Reagan² |
 Bill Yeh⁴ | Guangzheng Zang⁴ | Lee Fleming² 

¹Centre for Research in Economics and Management, University of Luxembourg, Esch-sur-Alzette, Luxembourg

²Coleman Fung Institute for Engineering Leadership, UC Berkeley, Berkeley, CA, USA

³Department of Electrical and Computer Engineering, American University of Beirut, Beirut, Lebanon

⁴UC Berkeley, Electrical Engineering and Computer Science, Berkeley, CA, USA

⁵Department of Economics, University of Paderborn, Paderborn, Germany

Correspondence

Lee Fleming, Coleman Fung Institute for Engineering Leadership, UC Berkeley, Berkeley, CA, USA.

Email: lfleming@berkeley.edu

Abstract

Drawing upon recent advances in machine learning and natural language processing, we introduce new tools that automatically ingest, parse, disambiguate, and build an updated database using U.S. patent data. The tools identify unique inventor, assignee, and location entities mentioned on each granted U.S. patent from 1976 to 2016. We describe data flow, algorithms, user interfaces, descriptive statistics, and a novelty measure based on the first appearance of a word in the patent corpus. We illustrate an automated coinventor network mapping tool and visualize trends in patenting over the last 40 years. Data and documentation can be found at <https://console.cloud.google.com/launcher/partners/patents-public-data>.

KEYWORDS

database, disambiguation, machine learning, natural language processing, patent, social networks

JEL CLASSIFICATION:

C80, C81, C88, O33, O34

1 | INTRODUCTION

Patent data have been used to study invention and innovation for over half a century (see Hall & Harhoff, 2012, for an overview). The popularity of patent data stems largely from the rich, consistent, and comparable information that can be obtained for a huge number of entities, that is, organizations, individuals, and locations. Aggregating patents remains difficult because entities are only listed by their names on each patent document and do not always receive a unique identifier from the patent office (at worse they remain inconsistent text fields). Looking at these fields as they appear on the patent document reveals various forms of misspellings or correct but different name spellings. The ambiguous names further limit the possibility to assemble patent portfolios for research as it is difficult to foresee all the kinds of name abbreviations that can occur. As a result of the lack of unique identifiers, individual researchers spend significant amounts of time and resources on labor-intensive manual disambiguations of relatively small numbers of patents. A few of these researchers (laudably) make these efforts available to the community, which results in a patchwork of retrospective coverages of different periods, types of data, and manual cleaning methods.

The problem of disambiguating inventor names has received considerable attention (Carayol & Cassi, 2009; Fleming & Juda, 2004; Lai, D'Amour, & Fleming, 2009; Li et al. 2014; Monath & McCallum, 2015; Pezzoni, Lissoni, & Tarasconi,

The authors wish to thank two very careful reviewers who greatly enhanced the work and manuscript and Google for hosting Bigquery. This work is supported by NSF grants 1360228 and 1536022, the US Patents and Trademark Office, the American Institutes for Research, and the Coleman Fung Institute for Engineering Leadership; errors and omissions remain the authors'. Balsmeier gratefully acknowledges financial support from the Flemish Science Foundation.

2012; Raffo & Lhuillery, 2009; Singh, 2005; Trajtenberg, Shiff, & Melamed, 2006). These efforts are gaining in sophistication, accuracy, and speed, such that fully automated approaches can now compete with smaller hand crafted and manually tuned datasets. Concurrent efforts have been made at the assignee level using automated and manual methods. Hall, Jaffe, and Trajtenberg (2001) disambiguated the assignees and introduced their patent data project under the auspices of the National Bureau of Economic Research (NBER). These data are widely used, partly because many assignees have also been matched to unique identifiers of publicly listed firms, which, in turn, enables easy matching with various other firm-level databases, for example, Compustat. Producing updates of the NBER patent data is costly, however, due to the quite sophisticated but still often labor-intensive process. Location data are available for most inventors (their home towns in particular) and while these data have been used previously, there does not exist a comprehensive and automated approach to their disambiguation.

Drawing upon recent advances in machine learning and natural language processing, the intent of this paper is to provide working prototypes for automating the disambiguation of patent entities and patent data manipulation, with the ultimate goal of providing tools and reasonably accurate and timely disambiguation data. Automation enables real-time updates with little manual effort and thus enables investigation of contemporary questions (e.g., many papers that study firms stop at the end of the NBER effort in 2006). The tools and data presented here are far from perfect and have not been fully characterized for their many potential applications; adopting an open innovation model, the thrust of this work is to put forth prototypes of automated disambiguation and data manipulation, with the hope that it greatly decreases manual cleaning and motivates further automation.

Illustrating the power of machine learning and natural language processing techniques in a different context than disambiguation, this paper also presents data and tools that open up new areas of investigation. It provides a measure of novelty, based on the first occurrence of a word in the patent corpus, following a baseline period of 1975–1985. This measure enables a prospective measure of a patent's novelty that remains orthogonal to the typically used measure of citations (which have been used to measure very different concepts such as impact, financial value, or knowledge diffusion). It also provides a real-time rendering of social networks, based on a list of patents or inventor input, along with two levels of surrounding indirect linkages. All code is available from the last author. Users are encouraged to consult the Bigquery documentation at <https://cloud.google.com/bigquery/quickstart-web-ui>; patent specific data and tool links are at <https://console.cloud.google.com/launcher/partners/patents-public-data>.

2 | DATA SOURCES, PARSING, AND PREPARATION

Preparing the data is more time-consuming and arduous than running the actual disambiguation. This section describes how data are obtained, parsed, cleaned, and structured. Knowing the necessary details should allow researchers to either circumvent this step by building directly on these efforts or at least save themselves some inevitable detours. Rather than parse weekly data available from the USPTO (in formats that have varied greatly over time and still include errors that have been since fixed by the USPTO), we scraped every granted patent from the USPTO web site and parsed, cleaned, and inserted the data into a SQL database. All patents up to the end of May 2017, including utility, design, plant, and reissues, are scraped and processed (some of the descriptive statistics below may reflect a December 2016 cutoff). Figure 1 illustrates the process.

As the data are extracted from USPTO documents, they are streamed into a relational database that contains records linking patents, citations, inventors, assignees, technology classes, and locations. The database itself uses the SQL database engine, but the patent library uses an object relational mapper (ORM) to allow database entries to be manipulated as Python objects. This simplifies development by removing the need to write code for a specific database back end, and facilitates use by not requiring the user to be familiar enough with relational databases in order to query and manipulate data. The raw tables in the database contain data as it appears in the patent record; this preserves the resolution of the data and gives the user freedom to develop their own processes over the original data. As the disambiguations are run, the raw records are linked to the disambiguated records.

The input data to be parsed come from the USPTO web site.¹ The output of the parsing algorithm is a tab-separated values (TSVs) file, with each row corresponding to a particular patent parsed. Going from raw HTML to TSV requires five steps.

2.1 | Step 1: Vectorization

Step 1 of the parsing algorithm iterates through the list of HTML files and extracts useful data from them. A python module called “Beautiful Soup” is used to convert the raw HTML text into a well-maintained python object, making it easier to extract text strings. The data are extracted into 23 fields, including grant ID/date, application ID/date, assignee names/locations, inventor names/locations, US Class, CPC Class, referred US patents, etc. If there is an error extracting data from a particular patent, the program would record that patent's ID, and skip it to continue to next patent (we encountered 195 errors, of which 183 resulted

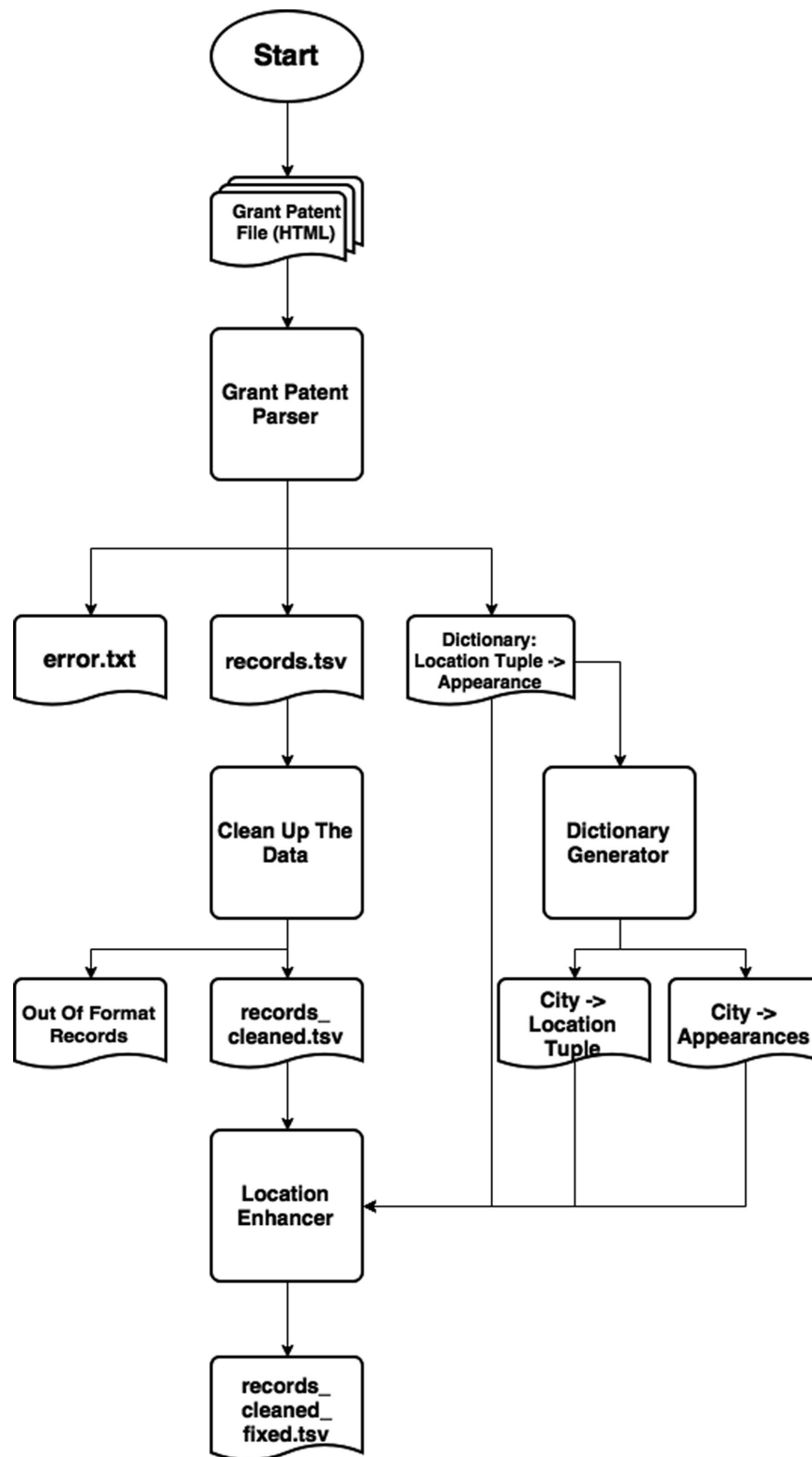


FIGURE 1 Tool and data flow for scraping and parsing USPTO web site data

in partial data and were still included in the database—12 had so many missing fields, such as all inventor names, that they were dropped). Moreover, in the process of extracting locations (for both assignees and inventors), the program maintains a dictionary file to keep track of the total number of appearances for each location tuple, which is defined as a “(CITY, STATE, COUNTRY)” tuple. All fields of this (Dictionary: Location Tuple→Appearance) dictionary are used later for enhancing location accuracy.

2.2 | Step 2: Soup to records

The second step cleans up the parsed records from step 1. This process is to ensure that parsed data are in a reasonable format that matches a correctly parsed patent's format. Specifically, we checked:

1. For each parsed patent (row), whether the total number of fields is 23 (the number of data fields scraped).
2. For each parsed patent (row), whether any of the fields contains a newline character.
3. For each parsed patent (row), whether the second last field is “0”, “1”, or “WITHDRAWN” (this field indicates whether there is an error flag (“*”) near the grant date).
4. Whether there are repetitions of patents parsed.

2.3 | Step 3: Geographical dictionary construction

Before starting to enhance the locations, step 3 of the algorithm constructs a dictionary: (City→Appearances), which will be useful in identifying the correct city name among subcity locations. To construct this dictionary, the program iterates through the (Location Tuple→Appearances) dictionary generated by step 1, identifies the city name in the location tuple, and adds the tuple appearance count to the city appearance count.

2.4 | Step 4: Geographical location cleaning

Step 4 enhances the locations with the help of the (City→Appearances) dictionary. The program goes through each patent, selects the city-level location, and splits the text string if possible. Then, the program finds the substring that has the highest appearances based on the (from city to appearances) dictionary, and changes the corresponding locations in the parsed data fields.

2.5 | Step 5: Manual validation of a random sample of records

In order to estimate the accuracy of the parsed data, we used the python function `random.sample(population, k)` to return a `k` length list of unique elements chosen from the population sequence without replacement. For each of the selected patents, we compared all relevant data fields parsed from the HTML file against its original PDF file (from www.google.com/patents/USXXXXXXX, where “XXXXXXX” is the seven-digit ID of the target patent). Having both the PDF and the parsed data available, we manually compared each of the data fields to see if they matched each other. Table 1 illustrates the specific data fields we compared and the result:

For each instance of mismatch shown above, we checked the HTML file. We found that the mismatches all resulted from inconsistencies between the HTML and PDF files (we maintained the original HTML data). Following the data preparation step, disambiguation can begin.

TABLE 1 Manual validation for a sample of 30 input records' parsing and cleaning

Data fields compared	Results
Patent Grant No.	30/30: match
Grant Date	30/30: match
Application No.	10/30: match; 20/30: simple convention difference, first two digits are missing in the PDF file
Application Date	29/30: match; 1/30: in PDF, Application Filed Date was incorrectly replaced by PCT Filed Date
Assignee Name/Location	30/30: match
Inventor Name/Location	29/30: match; 1/30: inventor name mismatch (in PDF: Günter, Bäumgen; in parsed: G/u/nter, B/a/umgen)
Referred US Patents	30/30: match
US Class	25/30: match; 5/30: convention difference, US Class updated
CPC Class	30/30: simple convention difference, CPC Class not appearing in PDF

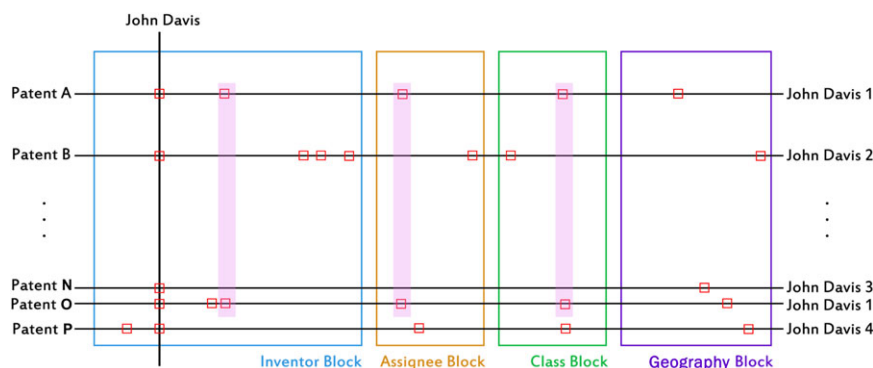


FIGURE 2 Toy example of a document (patent) by attribute (name, assignee, class, and geography) incidence matrix. A red box indicates an entry in the sparse matrix, in this case a 1, instead of the much more common 0. The black vertical line is the initial blocking of all inventors named exactly John Davis. The purple bars indicate shared attributes between two patents. For example, the purple line in the inventor block indicates a common coinventor, assignee block the exact same assignee, classification block the same section, class, subclass and group (everything to the slash in the CPCs listed on the patent), and, not shown in this toy example, an exact same city, state, and country match [Color figure can be viewed at wileyonlinelibrary.com]

3 | DISAMBIGUATION ALGORITHMS

3.1 | Inventors

We treat inventor disambiguation as a clustering problem; to disambiguate names, we seek high intracluster similarity and low intercluster similarity among patent–inventor pairs. While conceptually simple, the challenge is to accurately and quickly cluster the names from almost six million patents and 14 million names. We apply both bottom-up and top-down clustering approaches to avoid incorrect lumping and splitting of inventors. Inventor disambiguation uses the assignee and location disambiguations; though we describe it first, the inventor disambiguation runs last.

3.1.1 | Step 1: Vectorization

We model a patent as an unordered collection of that patent's attributes. The current version attributes are described below:

- Inventor name
- Coinventors (implicitly, by association on the same patent)
- The (disambiguated) assignees
- Any overlap in CPC technology classes
- The (disambiguated) city, state, and country location.

The occurrence of each attribute is used as a feature (i.e., independent variable) for training a classifier (the machine learning term for a tool that labels a record as a particular entity). We build a document-by-attribute incidence matrix, which is extremely sparse because a patent cannot use all possible attributes.

Figure 2 illustrates the matrix and submatrices conceptually. All the approximately 6M patents are arrayed horizontally. Each block arrays the different attributes as columns. For example, each inventor name has a column (such as John Davis), each assignee such as IBM, each CPC class, and each geographical location. The presence of an attribute is indicated with a “1” in that matrix entry. Using a discrete indicator may appear crude and wasteful of data; however, it is a conventional approach that is computationally efficient and favored with large datasets (and by computer scientists).

Suppose one or more inventors named John Davis have filed five patents, A, B, N, O, P. Let the inventors named John Davis initially be one column (depicted by the black vertical line in Figure 2 (essentially blocking on the exact name “John Davis,” where blocking is a machine learning term for temporarily restricting consideration to a subset of records, in this case, the five records with an inventor named John Davis). We look closer at the five rows that have 1s along that column, namely, the index of the patents being filed by the inventor named John Davis. Having formed a submatrix from just these five rows and their attributes, we can compute correlations between rows (patents) and columns (inventors).

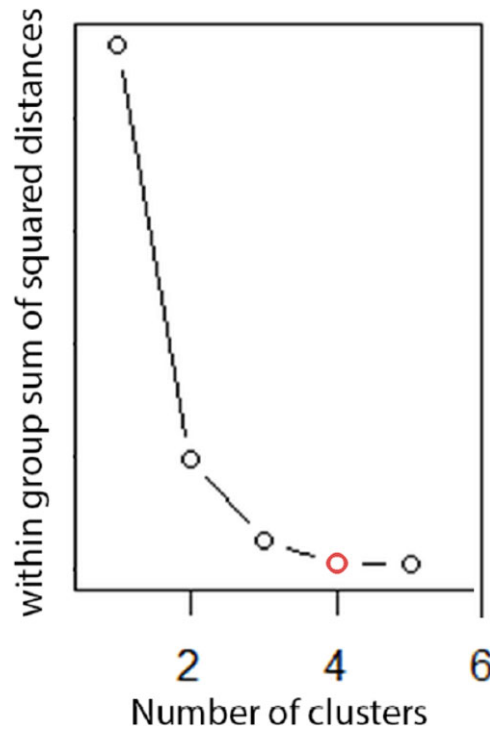


FIGURE 3 The optimal number of clusters is determined by the point of decreasing returns (4 in this example) [Color figure can be viewed at wileyonlinelibrary.com]

3.1.2 | Step 2: Bottom-up K-means clustering

Common names usually represent multiple persons, for example, John Davis. If, in our toy example, there were four such different people across the USPTO dataset, the algorithm should cluster and report four unique identifiers for John Davis.

From step 1, we take the short and very wide matrix of exactly similar inventor names and their attributes. To initialize, we treat each of these observations as a distinct person, by assigning unique initial identifiers. Then, we place all these unique identifiers into one block (there would be five in this toy example of John Davis). We then apply the K-means clustering algorithm based on a bottom-up approach (i.e., we start with five clusters). This step potentially merges some of these names and replaces them with a single identifier. K-means works iteratively to place an observation into the cluster with the most similar mean. It stops this process when the objective function described below reaches its optimum.

The goal of the K-means algorithm is to produce high intracluster and low intercluster similarity between inventors; the objective function to be maximized is expressed in Equation (1) as:

$$\phi^{(Q)}(\mathbf{X}, \lambda) = 1 - \frac{\sum_{i=1}^k \frac{n_i}{n - n_i} \sum_{j \in \{1, \dots, i-1, i+1, \dots, k\}} n_j \cdot \text{inter}(\mathbf{X}, \lambda, i, j)}{\sum_{i=1}^k n_i \cdot \text{intra}(\mathbf{X}, \lambda, i)}, \quad (1)$$

where $\phi^{(Q)}$ represents the cluster quality; the higher the score, the better. If the quality is 0 or lower, two items of the same cluster are, on average, more dissimilar than a pair of items from two different clusters. If the quality rating is closer to 1, it means that two items from different clusters are entirely dissimilar, and items from the same cluster are more similar to each other. Taking each term, in turn, the first term in the numerator normalizes for the size of the cluster, the second sums the distances between clusters, and the denominator sums the distances within clusters.

Figure 3 illustrates how the algorithm decides when to stop splitting clusters for the toy example. In this case, there appears to be little if any additional benefit of splitting into fewer than four clusters. When the sum no longer decreases (within some very small amount), there is almost no benefit to continuing to split clusters, and an increasing risk of splitting the same person into different clusters. Oversplitting is also minimized by following this initial splitting with the name merging algorithm described next.

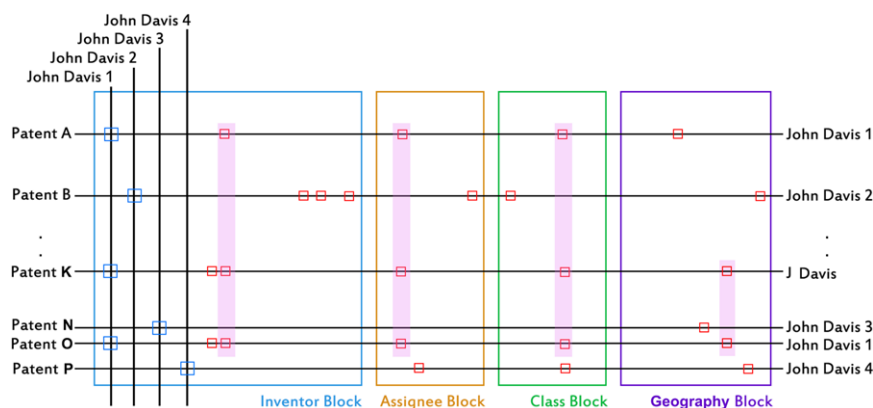


FIGURE 4 Augmentation with unique identifiers: as individuals are identified, additional columns are split out [Color figure can be viewed at wileyonlinelibrary.com]

3.1.3 | Step 3: Top-down name merging

Once the k-means bottom-up merging completes, we assign the most central observation of each cluster to be the unique identifier of that cluster and augment the matrix columnwise with these unique identifiers, as depicted in Figure 4. In other words, we uniquely label each cluster believed to be a different John Davis and affix this label to each column. This results in columns being split, and hence the number of columns in the working matrix increases, as separate inventors are identified. Compare Figures 2 and 4, where Figure 2 has one column marked John Davis and Figure 4 has four columns.

Following the k-means algorithm (a bottom-up approach), we add an additional step that attempts to merge incorrectly split inventors (a more top-down approach). The typical case occurs when there is matching upon the nonname attributes (coinventor, assignee, class, and geography) and yet the name fields (first name, middle initial, and last name) do not match exactly. This step is designed to merge Jim and James, Bob and Robert, or, Arnaud Gourdol, Arnaud P. Gourdol, Arnaud P. J. Gourdol, and Arno Gourdol as same persons (in our toy example, J. Davis might be one of our inventors named John Davis).

There are three cases where merging occurs if the k-means algorithm determines high similarity except for the name fields: 1) the last names agree, the first names disagree, and the first letter of the middle names agree (if any), for example, Jim and James; 2) the last names agree, the first names disagree, and the first letter of the first names disagree, for example, Robert and Bob; and 3) the last names disagree, the first names agree, and the first letter of the middle names agree (if any), due to marriage and last name change. In essence, if there is high nonname matching (coinventor, assignee, class, geography), there needs to be a match of at least one of the three name fields.

3.1.4 | Summary

The disambiguator considers each inventor's name, coinventors, geographical location, overlap in CPC technology class, and assignee to determine lumping and splitting of inventor names across patents. The approach treats each field as a vector and models its presence as an indicator (i.e., a 1 in that matrix location); the advantage of vectorizing attributes is that it allows for easy incorporation of new data sources. Current work plans to incorporate new data fields, such as bags of words or citations.

Relative to previous efforts that disambiguated the entire U.S. patent corpus (Lai et al., 2009; Li et al., 2014), these results are more and less accurate; our splitting error was 1.9% (compared with 3.3% of Li et al., 2014) and our lumping error was 3.8% (compared with 2.3%). The current method is much simpler and runs much faster and has approximately 1/10th of the code of Lai et al. (2009), however. Accuracy was assessed in the same manner as Li et al. (2014), by comparing to a manually curated sample, with the exception that patents after May of 2014 are not considered. Comparison files are available at the linkage address in the abstract.

Asian names are particularly challenging, in that first and last names often match, city and state data are often lacking, and much invention occurs within large firms. This makes it nigh impossible for any algorithm, including manual, to determine unique inventors. For example, Kim Lee is a very common Korean name, Samsung accounts for a large proportion of Korean patents, and the only location data are often the country of Korea. The disambiguation must rely on technology and coinventors only and understandably, sometimes fails.

We strongly recommend that all users sample their data and assess its accuracy for their research question. We provide metrics where possible that flag our doubts about any particular data point and encourage users to check such points most carefully.

Although we strive to provide the most accurate disambiguation possible, the user should conceptualize the data in its current state as a starting point (and help the community progress toward fully automated and blissfully accurate disambiguations).

3.2 | Assignees

The assignee records are used to determine the ownership of patents at the point the patent was initially applied for. This is helpful for studies that use archival data, but it limits the possibility to assemble patent portfolios of firms that bought or sold ownership rights to specific patents or firms that were involved in mergers and acquisitions (M&As) over time. Ownership changes are not tracked by the USPTO or any other institution yet though this may change (Stuart 2013).

Consider the following results from a cursory search for assignee records that resemble General Electric:

- General Electric Company
- General Electric
- General Electric Co..
- General Electric Capital Corporation
- General Electric Captial Corporation
- General Electric Canada
- General Electrical Company
- General Electro Mechanical Corp
- General Electronic Company
- General Electric Company

This incomplete list of some of the creative interpretations of General Electric demonstrates the most basic challenge of getting past typos and misspellings of the same intended entity. The unaltered records differ in structure from the inventor records because the majority of assignee records are organizations rather than individuals. Entity resolution of personal names struggle most to differentiate people who have the same name but are not the same individual. Organizational names are intrinsically free of this sort of false clumping. Instead, the disambiguation of organizational names is made difficult by three common patterns: acronyms (e.g., “IBM” for “International Business Machines”), corporate affixes (“Corporation” vs. “Corp” vs. nothing at all), and general misspellings.

We treat assignee disambiguation as a clustering problem as well, but with a different set of approaches than that of the Inventors. Assignee disambiguation poses some unique challenges. The number of patents per assignee is skewed; a very small number of assignees, for instance, IBM, hold thousands of patents, whereas many assignees are small organizations that hold very few patents. A second challenge is the many spellings of a single organization, as illustrated above.

We start with the dataset of disambiguated assignees generated by the National Bureau of Economic Research, herein referred to as the “NBER data.” These data contain over two million disambiguated patents from the US Patent Office with grant dates from 1976 through 2006. With the assumption that the NBER data are accurate, we use it to build a dictionary of all instantiations of the assignee name (basically, a complete list for each firm similar to the sample for General Electric above). We also build a feature space or description of each patent that contains a 1) location in country, state, city format, 2) first and last name of inventors, 3) list of other patents cited, and 4) Cooperative Patent Classification (CPC).

Disambiguation starts by examining each patent after the 2006 NBER data individually. If a patent's assignee is found in the dictionary (in other words, there is an exact lexical match of the assignee name with any entry in the assignee dictionary), the patent is linked to that assignee. If none is found, we attempt to find a match by looking at the patent's k -nearest neighbors (where $k = 5$ in this case). If no assignee within the k -nearest neighbors passes the similarity threshold, the assignee is considered to be a new label (i.e., an assignee not present in the NBER data before 2006). Clustering is performed on these remaining unassigned patents separately, as they are assumed to be (most often recently founded) firms that patented for the first time after 2006.

3.2.1 | Step 1: Dictionary build

First from the NBER data, we build a lookup dictionary (technically, a hash table) that links all abbreviated and raw names within the NBER-disambiguated patents to the unique identifier created by the NBER pdpass (patent database project assignee). Because the table includes both the standardized name given by the NBER and the raw name from the original patent, we are able to account for many common misspellings and abbreviations of each assignee already existing in the dataset. The lookup

dictionary aims to account for all abbreviations and misspellings of an existing assignee that have occurred in patents within the NBER database, between the years of 1976 and 2006. This creates quick matching during the first phase of the disambiguation process for representations of an assignee that have already been seen.

Second, we consider each assignee on a per patent per assignee basis. For each patent, the attributes noted in the above paragraph are recorded as binary vectors (1 where they occur and 0 where they do not) so that a binary matrix is built from the NBER data with one line per patent per assignee. This is the data matrix that we use to cluster the nearest neighbors of new assignees.

3.2.2 | Feature selection

The assignee matching features were further divided as the following:

- Location represented as Assignee country, state, and city.
- Inventor: Inventor first name with middle initial, inventor last name.
- CPC classification: Main class, subclass.
- Citation: All patents cited.

To standardize the names, we followed the same procedure as the NBER PDP process.

3.2.3 | Step 2: Dictionary match based on prior NBER database

New assignees names are first standardized and queried for in the lookup dictionary constructed from the NBER data. If an exact match is found, the assignee is grouped with the assignee of its match, the dictionary is updated, and the algorithm moves on. We find that approximately 74% of the assignee names are already contained within our lookup dictionary for the post-2006 time period.

3.2.4 | Step 3: k-Nearest neighbors classifier

If no match is found in the lookup dictionary, we then use an attribute matrix to find the k-nearest neighbors to a patent. The features of each assignee are binary, meaning they cannot exist along a gradient scale; each feature is either present or absent (0 or 1). The best way to represent an assignee is as a vector in N -dimensional space. In our algorithm, the binary feature space contains about five million different observations composed of the possible entries for location, inventor, classification, and citation. We use cosine similarity because it measures the cosine of the angle between each vector; the more features two vectors share, the smaller the angles between them and the greater the cosine similarity. Cosine similarity was chosen over other distance metrics because it is more effective when comparing categorical or binary features than a more traditional distance metric like Euclidean distance.

This algorithm uses cosine distance to find the closest five assignees in the NBER dataset to the current patent based on the attributes specified in the above section. The assignee's name is compared to that of the five closest neighbors, meaning the five assignees with the closest matches in locations, inventor names, citations, and classifications. The comparison is done through Levenshtein string distance, a measurement of the number of single-character edits (i.e., deletions, substitutions, or insertions) that are required to change the first string into the second, and any two names that are found to have a 75% similarity or greater are lumped together. If no match is found, then the assignee is assumed to be a new company and is assigned a new label. If a match is found, the dictionary is updated.

The combination of k-nearest neighbors and string matching has several advantages over conventional assignee matches of simple pairwise comparison of all assignees. First, it is much less computationally expensive as only five string comparisons per assignee are made rather than n comparisons, where n is the size of the entire dataset. Second, this helps reduce false positives by only comparing assignees that have a high probability through comparison of attributes. Third, assignees are matched to already existing clusters of names from the NBER dictionary, which include many common variations, abbreviations, and misspellings of company names. This method is especially effective at finding acronyms (e.g., “IBM” for “International Business Machines”) and variations in corporate affixes (“Corporation” vs. “Corp” vs. nothing at all).

The process ends when the assignees of all patents are either linked to an existing assignee identifier, or are determined to be an assignee that is not present in the NBER database. The disambiguated results are organized into a table with one entry per patent, per assignee with the following features:

- Patent: the patent number.

- **OriginalAsgName:** the “raw” assignee name.
- **MatchedAsg:** This is the assignee name it was matched with. If a match was not found, this is the raw assignee capitalized and punctuation removed for standardization.
- **MatchCertainty:** The degree of certainty for the match. 100 means dictionary match, –1 means the assignee found no match and is a new label
- **MatchMethod:** how the match was determined.
- **AsgIdent:** unique assignee identifier. If eight digits, this is the pdpass identifier from the NBER disambiguation. If 9 digits, this is our unique identifier.
- **Matches:** The specific features that each assignee matched with the assignee name being clustered.

3.2.5 | Step 4: Common name voting

After disambiguation occurs, newly linked assignees are clustered into a single Assignee Object that contains each raw name and patent occurrence of the assignee. The occurrence of each variation of a new assignee's name is counted and the one with the greatest counts provides the label as the standard name used for the assignee for the patent. Also listed are the first and last assignee names by patent grant date. Although we do not provide the link between all new assignees and Compustat, researchers can use “AsgIdent” and NBER data to extend coverage beyond 2006 for all Compustat firms that have patented at least once before 2007. Coverage will naturally decrease over time and with less frequently patenting firms.

3.2.6 | Step 5: Manual validation of a random sample of records

We randomly selected 30 patents without replacement from the post-2006 data that were not included within the NBER dataset and investigated their matches manually. Of the 30 tested, 27 were dictionary matches and 3 were considered new assignees (see Table 2). From inspection, it appears that only one of the new assignees was incorrectly assigned “Verigy (Singapore) Pte. Ltd.” because of the parenthetical city inside of the name. The confidence of each match is indicated as well; users are strongly encouraged to investigate poor matches.

3.3 | Locations

Geographic disambiguation has been greatly simplified by using the scraped data directly from the USPTO web site. The locations have already been cleaned quite thoroughly; although we perform some splitting of fields, for the most part, we merely pass the data through and rely on Google to assign latitude and longitude. For this process, we consulted with Jeffrey Oldham, an engineer at Google. He used an internal location tool similar to that used by Google Maps and returned the results. For each location, Google's API returns five fields:

- **city**, the full name of the city. For example, “San Francisco” or “Paris.”
- **region**, the name or two-letter abbreviation of a state, province, or other major institutional subdivision, as appropriate for the country. For example, “TX” or “Île-de-France.”
- **country**, the two-letter country code corresponding to the country. For example, “US” or “FR.”
- **latitude and longitude**, the latitude and longitude of the precise location found, depending on how much information is available.

4 | SUMMARY RESULTS AND DESCRIPTIVE STATISTICS

A variety of SQL tables are built and available to users (primary key(s) in parentheses):

- `dbo.raw_us_patent_claim` (`patent_id`, `claim_number`)
- `dbo.raw_us_patent_claim_related` (`patent_id`, `claim_number`, `related_claim_number`)
- `dbo.raw_us_patent_foreign_reference` (`patent_id`, `sequence_number`)
- `dbo.raw_us_patent_international_class` (`patent_id`, `sequence_number`)
- `dbo.raw_us_patent_other_reference` (`patent_id`, `sequence_number`)

TABLE 2 Manual validation results for a random sample of the assignee disambiguation

MatchAsgName	OriginalAsgName	MatchMethod	Patent	Check
INTEL CORPORATION	Intel Corporation	dictionary	8560484	correct
QUALCOMM INCORPORATED	QUALCOMM Incorporated	dictionary	8862113	correct
THE PROCTER & GAMBLE COMPANY	The Procter & Gamble Company	dictionary	7402554	correct
LG ELECTRONICS INC	LG Electronics Inc.	dictionary	8626437	correct
CANON KABUSHIKI KAISHA	Canon Kabushiki Kaisha	dictionary	7549734	correct
CONTINENTAL AUTOMOTIVE GMBH	Continental Automotive GmbH	new_asgn	8541959	correct
RESEARCH IN MOTION LIMITED	Research In Motion Limited	dictionary	8175275	correct
KABUSHIKI KAISHA TOSHIBA	Kabushiki Kaisha Toshiba	dictionary	9007846	correct
SYNOPSYS INC	Synopsys, Inc.	dictionary	8091055	correct
SAMSUNG ELECTRONICS CO LTD	Samsung Electronics Co., Ltd.	dictionary	7190939	correct
HITACHI KOKUSAI ELECTRIC INC	Hitachi Kokusai Electric Inc.	dictionary	8535479	correct
RED HAT INC	Red Hat, Inc.	dictionary	9060274	correct
MICROSOFT CORPORATION	Microsoft Corporation	dictionary	7605804	correct
LECO CORPORATION	Leco Corporation	dictionary	7851748	correct
ILLINOIS TOOL WORKS INC	Illinois Tool Works Inc.	dictionary	8551562	correct
VERIGY (SINGAPORE) PTE LTD	Verigy (Singapore) Pte. Ltd.	new_asgn	7519827	incorrect
SAMSUNG ELECTRONICS CO LTD	Samsung Electronics Co., Ltd.	dictionary	8027545	correct
KONICA MINOLTA BUSINESS TECHNOLOGIES INC	Konica Minolta Business Technologies, Inc.	dictionary	7476481	correct
MITSUBISHI CHEMICAL CORPORATION	Mitsubishi Chemical Corporation	dictionary	7830472	correct
DEPUY PRODUCTS INC	DePuy Products, Inc.	dictionary	7993407	correct
KYOCERA CORPORATION	Kyocera Corporation	dictionary	8045829	correct
LG ELECTRONICS INC	LG Electronics Inc.	dictionary	8077572	correct
SIEMENS AKTIENGESSELLSCHAFT	Siemens Aktiengesellschaft	dictionary	7194381	correct
MONTEFIORE MEDICAL CENTER	Montefiore Medical Center	dictionary	8765399	correct
OMRON CORPORATION	Omron Corporation	dictionary	7649491	correct
SUMCO TECHXIV CORPORATION	Sumco Techxiv Corporation	new Label	8251778	correct
GENERAL ELECTRIC COMPANY	General Electric Company	dictionary	7988420	correct
GM GLOBAL TECHNOLOGY OPERATIONS INC	GM Global Technology Operations, Inc.	dictionary	7491151	correct
MOLTEN CORPORATION	MOLTEN CORPORATION	dictionary	7621008	correct
EUV LLC	EUV LLC	dictionary	7196771	correct

- `dbo.raw_us_patent_pdpass` (`original_assignee_name`, `patent_id`)
- `nice.us_patent_cpc_class` (`patent_id`, `sequence_number`)
- `nice.us_patent_id_inventor_id` (`patent_id`, `inventor_id`)
- `nice.us_patent_lexical_novelties` (`patent_id`, `word`)
- `nice.us_patent_us_citation` (`patent_id`, `citation`)
- `nice.us_patent_us_class` (`patent_id`, `section`, `main`, `sub`)
- `dbo.us_patent_pdpass` (`pdpass`, `patent_id`)
- `nice.pdpass_assignee_name` (`pdpass`)
- `nice.us_patent` (`patent_id`)
- `nice.us_patent_assignee` (`patent_id`, `sequence_number`)
- `nice.us_patent_inventor` (`patent_id`, `sequence_number`)

Table 3 lists some of the variables that are available and their descriptive statistics when appropriate (each variable is unique only within its given table). Figure 5 provides a histogram of the number of patents by each unique inventor, up to 50 patents,

TABLE 3 Descriptive statistics of selected variables. Nonnumeric observations are not included in calculations (e.g., the first-row calculation is based on 5,701,096 observations due to reissue, design, and plant patents) and backward cites include applications

Table name	Variable name	Description	<i>n</i>	Mean	Std dev	Min	Max
nice.us_patent	patent_id	patent no.	6,287,338	6796433	na	3,930,272	9,668,395
nice.us_patent	application_date	Filing date	6,287,290	3/7/2000		8/15/1868	3/24/2017
nice.us_patent	grant_date	Grant date	6,287,338	10/10/2002		1/6/1976	5/30/2017
nice.us_patent	num_inventors	Number of inventors	6,287,338	2.34	1.73	0	76
nice.us_patent	num_assignees	Number of assignees	6,287,338	0.90	0.39	0	35
dbo.raw_us_patent_pdpass	pdpass	pdpass of assignee	5,038,950	na	na	-19,840	209,357,348
dbo.raw_us_patent_pdpass	match_certainty	Certainty of pdpass match	5,038,950	99.97	0.57	80	100
nice.us_patent	num_times_cited	Forward prior art cites to patent	6,287,338	11.93	30.16	0	3,830
nice.us_patent	num_us_citations	Backward prior art cites from pat	6,287,338	17.74	53.01	0	7,705
nice.us_patent	num_self_future_citations	Future prior art cites from same pdpass	6,287,338	1.03	6.38	0	1,941
nice.us_patent	num_self_citations	Backwards future prior art cites to same pdpass	6,287,338	1.03	5.59	0	560
nice.us_patent	num_foreign_citations	Foreign citations made by patent	6,287,338	3.44	12.59	0	1,071
nice.us_patent	num_other_citations	Other citations made by patent	6,287,338	4.86	25.89	0	2,964
nice.us_patent	num_claims	Count of claims	6,287,338	14.07	12.15	0	887
nice.us_patent_lexical_novelities	current_use	Count of novel words	2,816,425	2.79	5.97	1	1,472
nice.us_patent_lexical_novelities	future_use	Future reuse of novel words	2,816,425	1.98	48.30	0	55,922

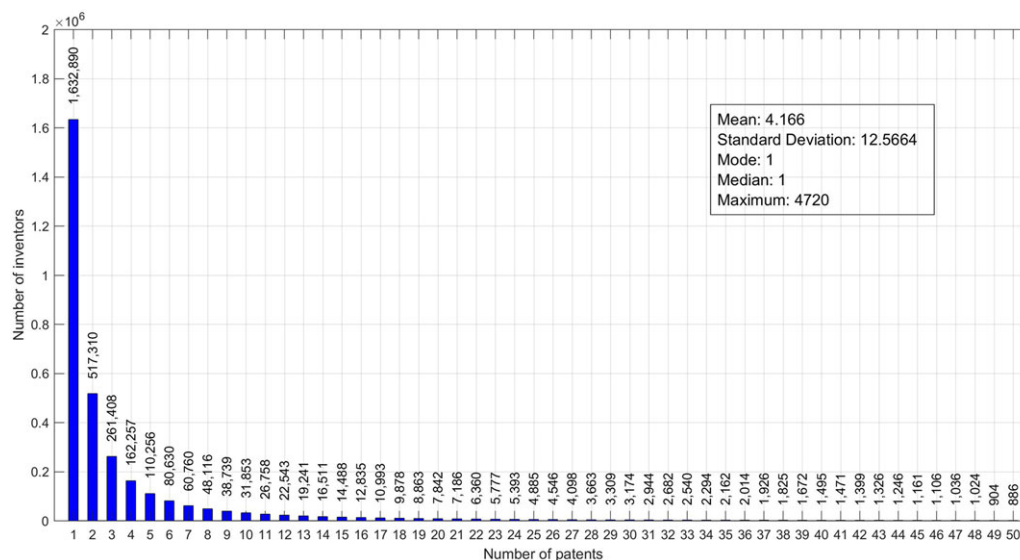


FIGURE 5 Histogram of number of patents by inventor: 22,286 inventors, or 0.69% of the total, have more than 50 patents. Data from 1976 to 2016 [Color figure can be viewed at [wileyonlinelibrary.com](#)]

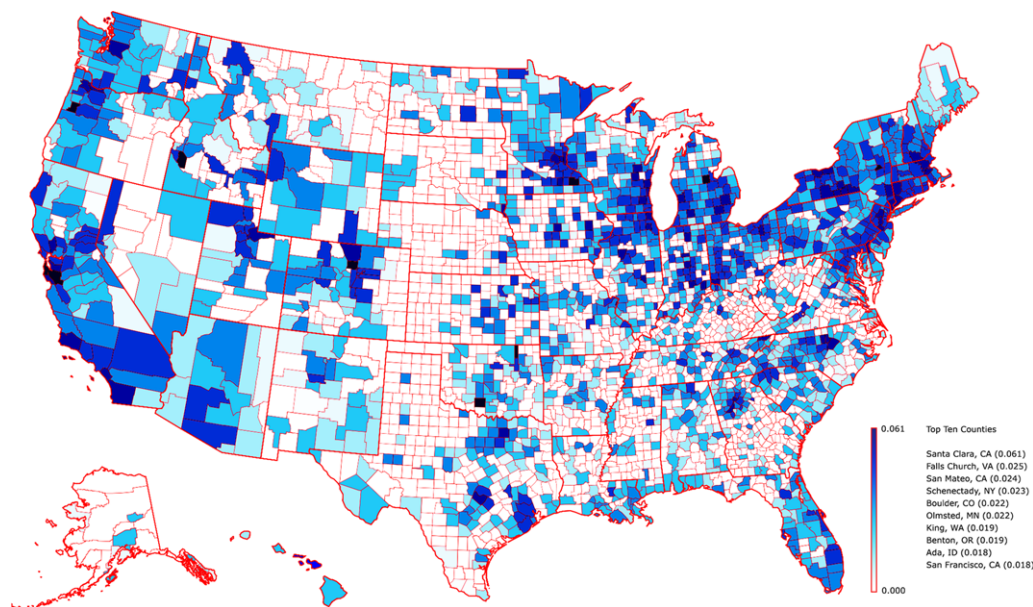


FIGURE 6 Patenting per capita by county, 1976–2016 [Color figure can be viewed at [wileyonlinelibrary.com](#)]

from January of 1976 through December of 2016, inclusive. The distribution is highly skewed; 1.6 million inventors have only one patent, and 22,286 inventors—0.69% of the whole sample—have more than 50 patents.

Figures 6 and 7 illustrate county-level choropleths in the United States. Figure 6 illustrates patenting per capita and indicates an outlier in Santa Clara, California, corresponding to the heart of Silicon Valley (where there are 0.061 patents per person). The list of top 10 counties highlights newly emerging hubs in the west, such as San Francisco and environs, Boulder, Seattle, and Corvallis, and northern Virginia in the east. The high technology regions of the industrial age in the northeast continue to patent (though that is weakening, as illustrated in Figure 7).

Figure 7 illustrates the change in patenting between recent activity and the availability of data in 1976. Blue indicates a drop in recent patenting and includes much of the northeast, including Pittsburgh (in Allegheny county), as well as a decline in Los Angeles county in California. Raleigh (in Wake county), North Carolina, and Merced, California, have increased their patenting, assumedly due to high technology development in the former case and the construction of a new University of California campus in the latter.

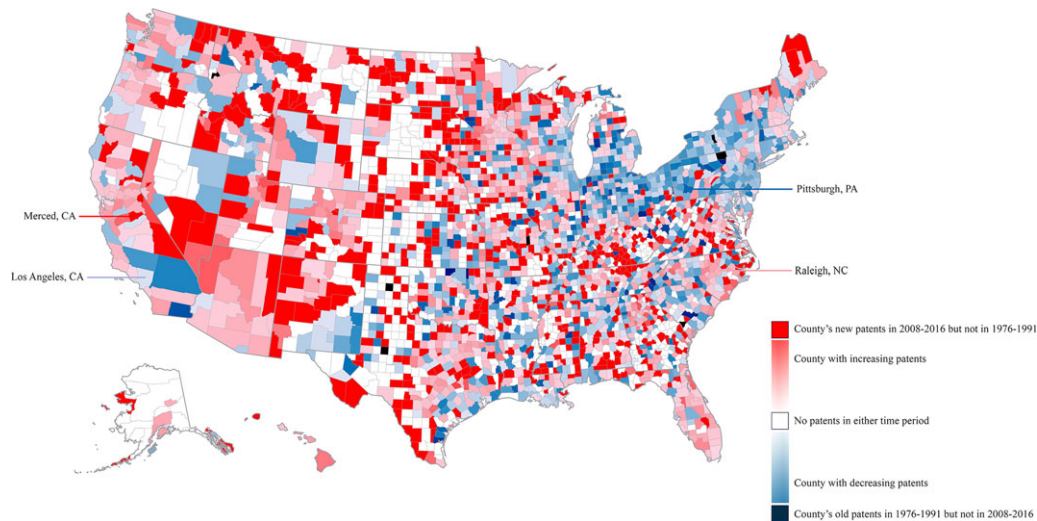


FIGURE 7 Change in patenting by county, comparing 1976–1991 to 2008–2016 [Color figure can be viewed at wileyonlinelibrary.com]

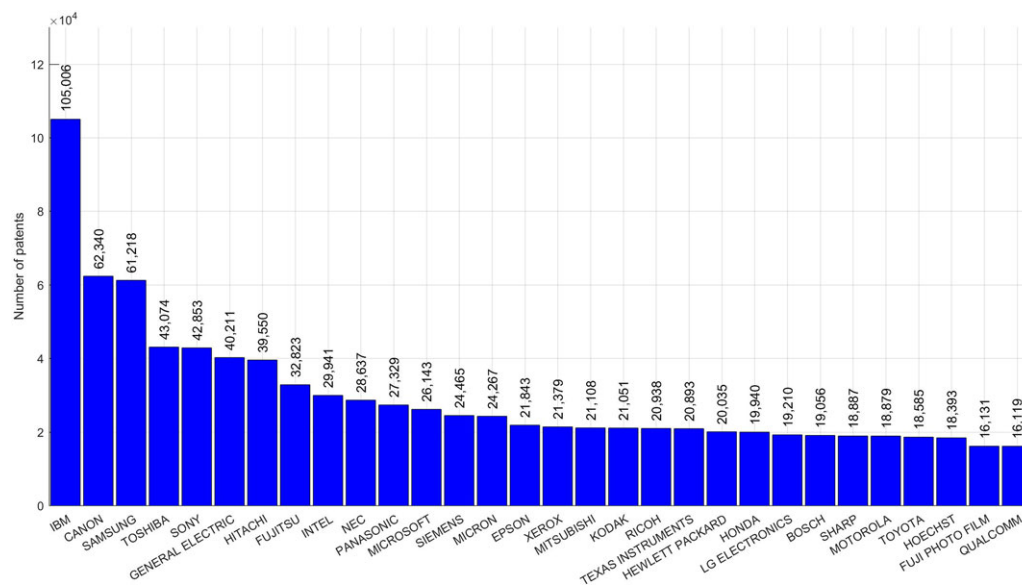


FIGURE 8 Histogram of top 30 assignees by number of patents. Data from 1976 to 2016 [Color figure can be viewed at wileyonlinelibrary.com]

Figure 8 provides a histogram of the top 30 patenting assignees from 1976 to 2016. IBM is the top patenter, though only three U.S. firms (IBM, GE, and Intel) are in the top 10. Japanese and one Korean firm take the other spots in the top 10 assignees.

5 | FIRSTPATWORD: A LEXICAL MEASURE OF PATENT NOVELTY

Scholars have long sought to explain the sources and process of creativity and innovation, in an effort to optimize the production and reap the benefits of novelty. Unfortunately, this research continues with little resolution in sight. Varied definitions and measures cause part of the problem; for example, in the patent literature, citations have been used to measure both creativity (Bernstein, 2014) and financial impact (Hall & Harhoff, 2012), even though highly cited patent portfolios can result from an exploitation strategy that focuses on extant firm capabilities rather than exploratory search (Balsmeier, Fleming, & Manso, 2017).

To provide a cleaner measure of novelty, we developed a baseline dictionary of all words used in the patent literature between 1975 and 1985, and then identified the patents from 1985 to 2014 that used a new word that was not included in that dictionary (a reviewer suggested a rolling window, which would be preferable, though require greater computation). Only the first patent in a year is counted. All subsequent patents are not counted as novel, regardless of how much time elapsed between first and second

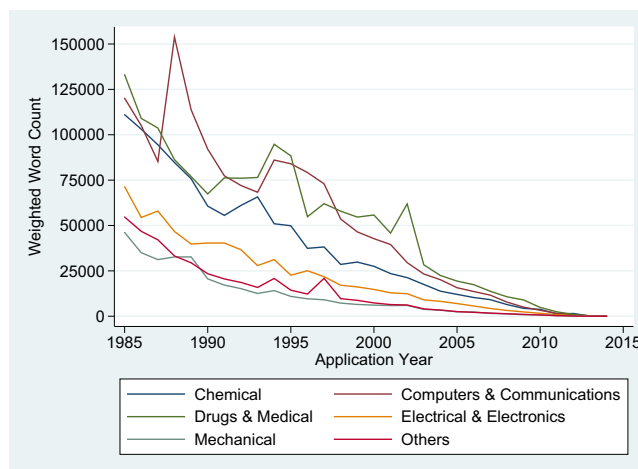


FIGURE 9 Novel words applied for in a given year, weighted by their future reuse, NBER technology category [Color figure can be viewed at wileyonlinelibrary.com]

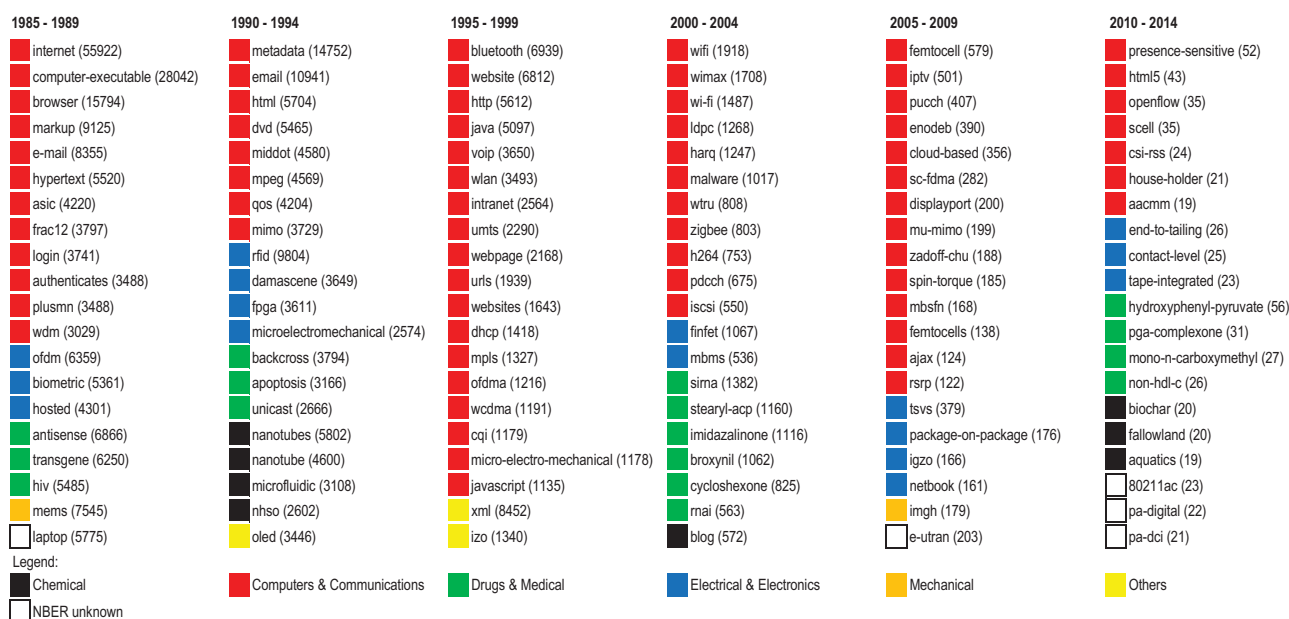


FIGURE 10 Top 20 most reused novel words within a 5-year period [Color figure can be viewed at wileyonlinelibrary.com]

occurrences. Words with leading or ending hyphens or a double hyphen were dropped, as well as words consisting entirely of numeric characters. Even though many words could be typographic errors, further cleaning was avoided, as it is very difficult to differentiate between such errors and truly new words. This effort resulted in:

- 4,791,515 patents with an application date 1985 and 2014 (including 530,799 patents where the NBER categories are not known),
- 1,121,468 patents with at least one new word,
- 3,670,047 patents with no lexical novelty,
- 2,816,425 new words used from 1985 to 2014 that had not been previously used.

To get more detailed information about the lexical novelties, we separated the patents into the six technological categories introduced by Hall et al. (2001). To get the technological categories for all patents up to 2014, we linked our dataset with the USPTO Historical Master File described by Marco, Carley, Jackson, and Myers (2015). The graphs above illustrate the new measure. Figure 9 illustrates novel words that are frequently reused and could be interpreted as “successful” novelty (it is a raw metric and must be normalized by time at risk of reuse) and Figure 10 lists the top 20 reused words in each 5-year time period.

This measure could be used somewhat analogously with future prior art citations. The peaks in the Figure 9 can be partially explained by the following particular words:

Computers & Communications:

Peak 1988–1990:

- 1988: Internet (55,922) e-mail (8,355) hypertext (5,520)
- 1989: computer-executable (28,042) browser (15,794)
- 1990: metadata (14,752)

Peak 1994–1996:

- 1994: email (10,941) html (5,704)
- 1995: http (5,612) java (5,097)
- 1996: web site (6,812) (and web sites (1,643))

Drugs & Medical:

Peak 1991–1995:

- 1991: abiotic (2,486) transgenes (2,114) cdr3 (1,256) cdr1 (1,191) cdr2 (1,180) (note: CDRs [Complementarity determining regions] are part of the variable chains in immunoglobulins [antibodies] and T cell receptors.)
- 1992: backcross (3,794) apoptosis (3,166) unicast (2,666) chemokine (1,401) scfv (1,170)
- 1993: wap (1,179)
- 1994: introgressed (1,184)
- 1995 (many novel words with future reuse, because of the peak in the number of novel words in 1995 in Drugs & Medical): arylc (686) irinotecan (562) atorvastatin (522) leptin (478) polyaxial (434) aptamers (427) cycloalkylc (409)

Peak 2002:

- sirna (1,382) broxynil (1,062) cyclohexone (825)

Chemical:

Peak 1993: nanotubes (5,802) nanotube (4,600) paclitaxel (1,993) micro-electromechanical (1,319)

Mechanical:

Peak 1987–1989:

- 1987: drm (763)
- 1988: nanocrystalline (1,076) batio (1,045)
- 1989: mems (7,545) nanocrystals (1,081)

Others:

Peak 1994:

- oled (3,446)
- Peak 1997: xml (8,452) izo (1,340)

6 | A COINVENTOR NETWORK VISUALIZATION TOOL

Social networks have become increasingly popular, yet visualizing such networks requires time and technical expertise. Using the disambiguated patent database, we have developed a tool that dynamically renders the immediate coinventor networks for any inventor(s) or patent(s) chosen by the user. Users may start with either a list of inventors or a list of patents. The BigQuery site listed in the abstract has the link to the network tool.

If starting from chosen (“seed”) inventors, the tool can find all patents whose inventors have applied for within the chosen dates. For each patent, it then creates a coinventor link between all possible pairs of the patent's inventors. For a coinventor network (defined as two “generations” or levels of coinventors), it then uses this larger set of coinventors as the “seed” inventors and cycles through again. In principle, this process can be repeated to n -generations of coinventorship. To limit demands

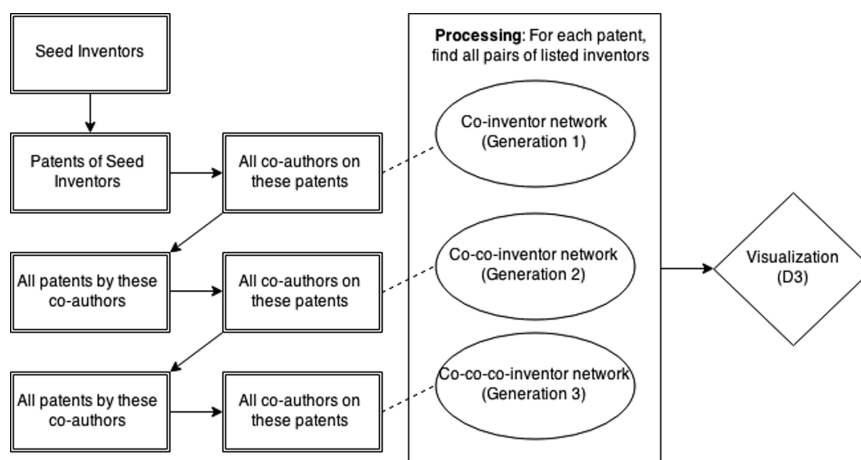


FIGURE 11 Flow diagram for network visualization tool

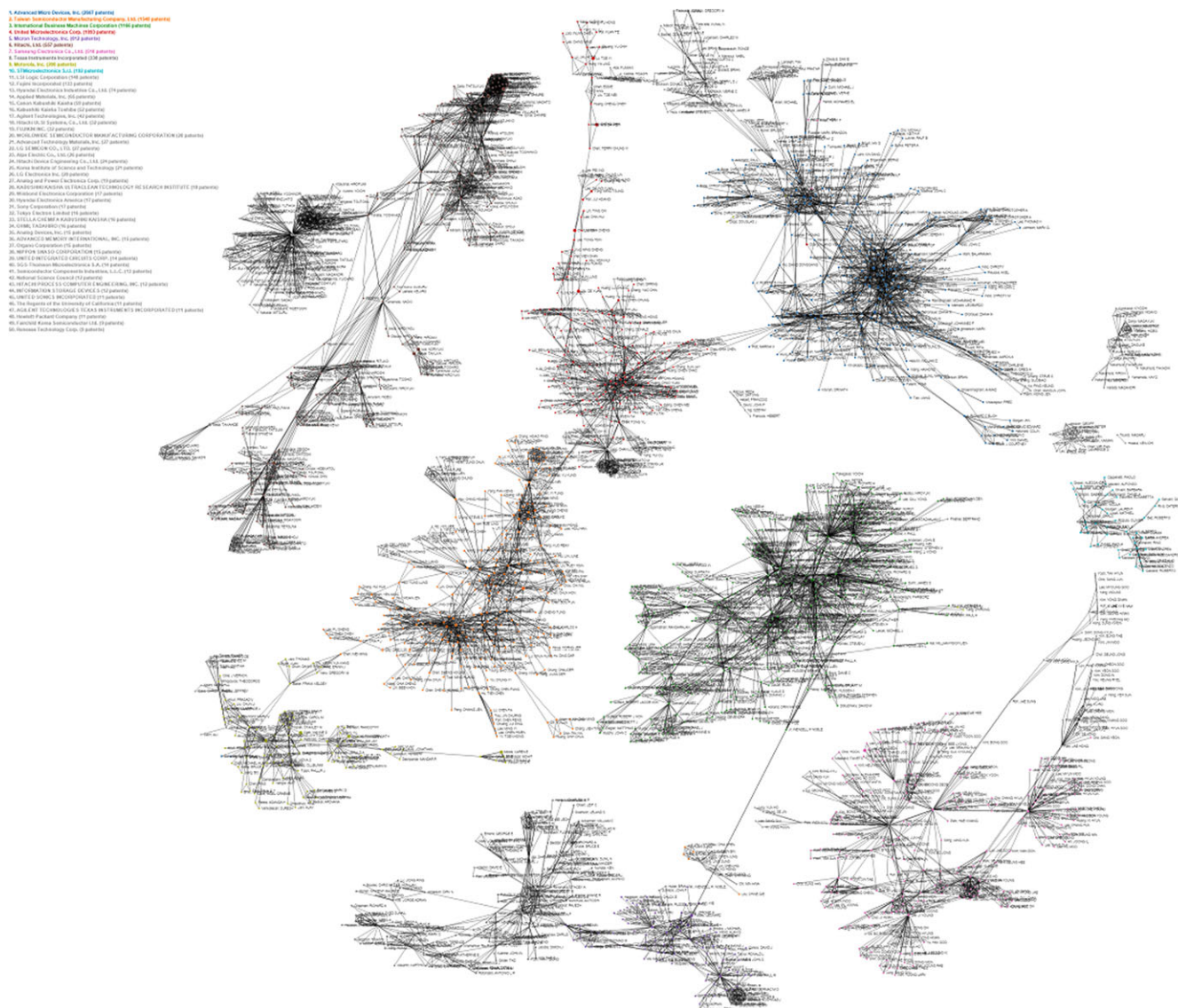


FIGURE 12 USPTO semiconductor patents in subclass 438/283 from 1998 to 2000, with two additional levels of coinventors included [Color figure can be viewed at wileyonlinelibrary.com]

on bandwidth and processing, users can currently only choose one, two, or three generations. Figure 11 diagrams a process flow.

This program is driven by PHP, drawing data from the SQL patent database and storing results in a separate, local SQL database. After coinventor relationships have been calculated, the end result is stored in a JSON-formatted file for visualization. The annotated PHP/SQL code is available from the last author.

Once all coinventor relationships have been identified, the tool generates a visualization in which inventors are represented by dots that act as charged particles, repelling one another, but with coinventors bound together. The visualization itself uses the Data-Driven Documents (D3) force layout graph, modified from the publicly available version at <https://github.com/mbostock/d3/wiki/Force-Layout>. This graph uses a Barnes-Hut algorithm (<http://arborjs.org/docs/barnes-hut>) to calculate the forces between the charges and find an equilibrium position. This process renders directly on the user's browser, typically within seconds. Particularly, large networks can take up to minutes to generate and render.

Figure 12 illustrates an example from the semiconductor industry, with two additional coinventor relationships, for patent subclass 438/283 from 1998 to 2000. This follows from the search option that takes a list of patents as input and graphs all inventors associated with that input. The top patenting firms are all clearly clustered (AMD, TSMC, IBM, etc.), indicating relatively little mobility. This is not surprising, as these firms are isolated geographically (Silicon Valley, Taiwan, and New York), strategically (microprocessors, foundry, and internal capacity), and in labor markets (fluid and external Silicon Valley, Taiwan, and an internal labor market). Such a fragmented collaboration network would not support knowledge spillovers, though they may have been more common in the industry's earlier years.

7 | CONCLUSION

Recent advances in machine learning and natural language processing have enabled more automated updates and disambiguations of the U.S. patent corpus. Access to disambiguated current data can greatly facilitate fundamental and policy research into innovation, innovation policy, and technology strategy. Unlike prior and often manual efforts, automation facilitates scaling and can be (re)run at relatively low cost, thus enabling iterative analyses of recent innovations in almost real time. We provide an initial set of fully automated tools aimed at accomplishing this goal for patents from 1976 through 2016. The tools are simple but provide a platform for future improvement. They first provide inventor, original assignee, and geographical location disambiguations. The new data enable visualization of patenting trends and the geographic distribution of invention over the last 40 years. The work also provided a tool that dynamically renders patent inventor networks and a measure of patent novelty based on the first appearance of a new word in the patent lexicon. The methods presented here are simple, and while less accurate than some prior and competing approaches, the hope is that they will provide an automated platform for improvement. Ideally, a sponsor will be found, such that these data can be curated and updated on a weekly basis.

Much work remains to be done (and we encourage the community to continue it). A variety of disambiguation approaches are now available (e.g., the UMass Amherst inventor disambiguation or the OECD HAN name harmonization); comparison and identification of each dataset's strengths and weaknesses would be invaluable. The social network illustration tool could reflect failed applications as well as successful patents. Ideally, this would be rendered such that the two networks are laid out similarly, allowing easy identification of the failed inventions and relationships. To improve the inventor accuracy, one could introduce additional attributes, including law firms, nonpatent and patent prior art, and a distance between technology class measure. The last has typically been done with patent classes (see Li et al., 2014). Technology classes evolve, however, and get abolished or established over time (indeed, the USPTO now uses the international classes). Another approach would develop a bag of words or n-grams, or descriptive "tags," which would compare and contrast each patent from another. By assuming that the words, or tags, or certain keywords are statistically specific to inventors, we may feed patent contents as another block into the disambiguation. Distance measures can also be relaxed, such that closed but nonexact matches can contribute to the correct clustering. For the assignee disambiguation, one could compile comprehensive sets of patent portfolios across conglomerates and large organizations that operate under different names of their subsidiaries. A test download of the recently available Bureau van Dijk (BvD) database suggests that significant improvements over previous approaches might be possible.

ENDNOTE

¹ The URL is <http://patft.uspto.gov/netacgi/nph-Parser?Sect2=PTO1&Sect2=HITOFF&p=1&u=/netahtml/PTO/searchbool.html&r=1&f=G&l=50&d=PALL&RefSrch=yes&Query=PN/XXXXXXX>, where "XXXXXXX" is the seven-digit ID of the target patent.

ORCID

Lee Fleming  <http://orcid.org/0000-0003-1363-4478>

REFERENCES

- Balsmeier, B., Fleming, L., & Manso, G. (2017). Independent boards and innovation. *Journal of Financial Economics*, 123(3), 536–557.
- Bernstein, S. (2014). Does going public affect innovation? *Journal of Finance*, 70(4), 1365–1403.
- Carayol, N., & Cassi, L. (2009). *Who's who in patents. A Bayesian approach*. Cahiers du GREThA 2009–07.
- Fleming, L., & Juda, A. (2004). *A network of invention*. USPTO Economics Working Paper No. 2013–2.
- Hall, B. H., & Harhoff, D. (2012). *Recent research on the economics of patents*. NBER Working Paper 17773.
- Hall, B. H., Jaffe, A. B., & Trajtenberg, M. (2001). *The NBER patent citations data file: Lessons insights and methodological tools*. NBER Working Paper.
- Lai, R., D'Amour, A., & Fleming, L. (2009). The careers and co-authorship networks of U.S. patent-holders, since 1975. <https://dataverse.harvard.edu/dataset.xhtml?persistentId=hdl:1902.1/15705>.
- Li, G., Lai, R., D'Amour, A., Doolin, D., Sun, Y., Torvik, V., .. Fleming, L. (2014). Disambiguation and co-authorship networks of the U.S. Patent Inventor Database, 1975–2010. *Research Policy*, 43, 941–955.
- Marco, A. C., Carley, M., Jackson, S., & Myers, A. F. (2015). *The USPTO Historical Patent Data Files. Two centuries of invention*. USPTO Economic Working Paper No. 2015-1, http://www.uspto.gov/sites/default/files/documents/USPTO_economic_WP_2015-01_v2.pdf.
- Monath, N., & McCallum, A. (2015). *Discriminative Hierarchical coreference for inventor disambiguation*. Patents View Conference Presentation. <http://www.patentsview.org/data/presentations/UMassInventorDisambiguation.pdf>.
- Pezzoni, M., Lissoni, F., & Tarasconi, G. (2012). *How to kill inventors: testing the Massacrator© algorithm for inventor disambiguation*. Cahiers du GREThA no 2012–29. <http://ideas.repec.org/p/grt/wpegrt/2012-29.html>.
- Raffo, J., & Lhuillery, S. (2009). How to play the “Names Game”: Patent retrieval comparing different heuristics. *Research Policy*, 38, 1617–1627.
- Singh, J. (2005). Collaborative networks as determinants of knowledge diffusion patterns. *Management Science*, 51, 756–770.
- Trajtenberg, M., Shiff, G., & Melamed, R. (2006). *The names game: Harnessing inventors patent data for economic research*. NBER Working Paper.

How to cite this article: Balsmeier B, Assaf M, Chesebro T, et al. Machine learning and natural language processing on the patent corpus: Data, tools, and new measures. *J Econ Manage Strat*. 2018;27:535–553. <https://doi.org/10.1111/jems.12259>