An Hypernetwork Approach to Accurately Measure Technological Innovation

Methods and Results

J. Raimbault[1,2] and A. Bergeaud[3]

[1] UMR CNRS 8504 Géographie-cités
[2] UMR-T IFSTTAR 9403 LVMT
[3] London School of Economics

Abstract

This working paper details technical methods used in the project and gives a first overview of results.

1 Introduction

See project Proposal

2 Methods

Originality Measures The originality measure is defined by Hall *et al.* (2001) [Hall et al., 2001] as

$$O_i = 1 - \sum_{j=1}^{n_i} c_{i,j}^2$$

where $c_{i,j}$is the percentage of citations made by patent i to a patent in class j out of n_i technological classes to which patent i belongs. If the scope of technologies which the patent uses and cites is large, then the originality measure will be high. Radicalness is more difficult to define. It is constructed in the same way as the originality index but here we only consider the technology classes of patents cited by patent i but to which patent i does not belong. These two indicators are good proxies and great start to estimate if a patent is protecting a new product that can hardly be classified into the official technological field space.

Citation Network We define a binary relationship between each pair of patents $Cit(i,j) = 1$ if j cites i or i cites j, otherwise $Cit(i,j) = 0$.

Technological Class Network For each patent i, let B_i be the set of technological class of i. We then define a relationship between each pair i and j as 2 times the number of technological class in common divided by the total number of class of i and j.

$$Class(i,j) = 2\frac{|B_i \cap B_j|}{|B_i| + |B_j|}$$

Thus, if two patents have no class in common, $Class(i,j) = 0$ while if the two patents are exactly identical in terms of their sets of technologial class $Class(i,j) = 1$.

Computation of the Technological Network With a number of patents of magnitude 10^6, it is not considerable in memory and in time to compute pairwise distances on all patents. Fortunately, distance matrix is relatively sparse and non trivial proximities (i.e. where patents have more than one class in common) can be computed through an efficient set intersection and difference implementation.

More precisely, let $n = |\mathcal{P}|$ the number of patents. Classes are constructed in $O(n)$ from file giving patents classes, and then sorted in $\Theta(\sum_k |\mathcal{C}_k| \cdot \log |\mathcal{C}_k|)$ with $\mathcal{C}_k$ the classes. The number of classes K being fixed, it simplifies into a $O(n \log n)$. This sorting allows to compute overlaps and differences between classes in linear time by a direct comparison. To compute higher order overlaps, we have an exponential complexity in the overlap order o (in $O(K^o)$) but order three appears experimentally to have cardinals smaller enough for a reasonable one-to-one technological distance computation. A class partition strategy allows to know any distance : if $o(\mathcal{C}_i, \mathcal{C}_j)$ is first order overlap between classes i and j and $o_2(\mathcal{C}_i, \mathcal{C}_j, \mathcal{C}_k)$ second order overlap (i.e. $o(\mathcal{C}_k, o(\mathcal{C}_i, \mathcal{C}_j))$, then for any class i,

- between any $p \in \mathcal{C}_i$
  $\cup_k\, o(i,k)$, proximity is necessarily 1 (one class in common for one-class patents)
- for any $j \neq i$

2.1 Semantic Network

Significant Keywords extraction procedure We first assign to a patent $p \in \mathcal{P}$ a set of significant keywords $K(p) \in \bigcup_{n\in\mathbb{N}} \mathcal{A}^{*n}$, that are precisely extracted through a procedure similar to the one detailed in [Chavalarias and Cointet, 2013], consisting in the following steps

1. Text parsing and tokenizing.
2. Part-of-speech tagging, normalization.
3. Stem extraction and multi-stems constructions.
4. Relevant multi-stems filtering.

Text processing operations are implemented in `python` in order to use the `nltk` library [Bird, 2006] which is highly ergonomic and supports most advanced state-of-the-art natural language processing operations. Source code is openly available on the repository of the project[1]. Steps one to three are directly done using built-in functions of the library. Step four needs a particular treatment that we propose as an extension of the original method for large corpuses, which is detailed in the following.

Bootstrap on random sub-corpuses for relevance estimation Once multi-stems have been extracted, one scores them by *unithood*, defined for the multi-stem i by $u_i = f_i \cdot \log(1 + l_i)$ where f_i is the number of apparitions of the multi-stem over the whole corpus and l_i its length in words. Let K_w the maximal number of relevant keywords per patent. If N is the total number of relevant keywords extracted, that we consider as a parameter, the heuristic described in [Chavalarias and Cointet, 2013] proposes a first filtration of $4 \cdot N$ keywords on the whole corpus, and then a filtration on a secondary score called *termhood*, computed as a chi-squared score on the distribution of the stem, compared to an uniform distribution within the whole corpus. More precisely, one computes the co-occurence matrix (M_{ij}), defined as the number of patents where stems i and j appear together, what allows to define the *termhood* score as

$$t_i = \sum_{j\neq i} \frac{(M_{ij} - \sum_k M_{ik} \sum_k M_{jk})^2}{\sum_k M_{ik} \sum_k M_{jk}}$$

Let $X = \{x_1, \ldots, x_N\}$ a discrete set and $X_1, \ldots, X_K \in \mathcal{P}(X)$. We write $X_k = \{x_1^{(k)}, \ldots, x_{N_k}^{(k)}\}$. Exact probability distributions within sets are determined directly in $O(\sum_{k=1}^K N_k)$.

[1] at `url`

Possible Features *From proposal :*

- Citation relative centrality regarding technological or semantic classes : given a partition of $\mathcal{P}$ represented by a classification function C (constructed for example by clustering or community detection within technological or semantic networks), a vector feature is for patent i

$$\left(\frac{\sum_{j \in c} Cit^{out}(i,j)}{\sum_{j \in c} Cit^{in}(i,j)} \right)_{c \in C(\mathcal{P})}$$

- Dynamic evolution of classification vector : if C_t is stratified over successive time periods indexed by time t, either $\Delta \vec{C}_t(i)$ if $\vec{C}_t$ is a vector of probabilities to belong to each class (in case of a Bayesian approach), or $\Delta(C_t(j))_{Cit(i,j,t) \neq 0}$ in case of a deterministic approach, could both be interesting features.

- Deviation from the expected classes given position in other layers of the hypernetwork (it would need explorations if these conditional probabilities first can be well estimated, then if they contain relevant information).

Multilayer Network Analysis Check out the method proposed in [Iacovacci et al., 2015]. → Interesting for some kind of "between-layers correlation" ?

3 Results

References

[Bird, 2006] Bird, S. (2006). Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics.

[Chavalarias and Cointet, 2013] Chavalarias, D. and Cointet, J.-P. (2013). Phylomemetic patterns in science evolution—the rise and fall of scientific fields. *Plos One*, 8(2):e54847.

[Hall et al., 2001] Hall, B., Jaffe, A., and Trajtenberg, M. (2001). The NBER Patent Citations Data File: Lessons, Insights and Methodological Tools. Papers 2001-29, Tel Aviv.

[Iacovacci et al., 2015] Iacovacci, J., Wu, Z., and Bianconi, G. (2015). Mesoscopic structures reveal the network between the layers of multiplex datasets. *arXiv preprint arXiv:1505.03824*.