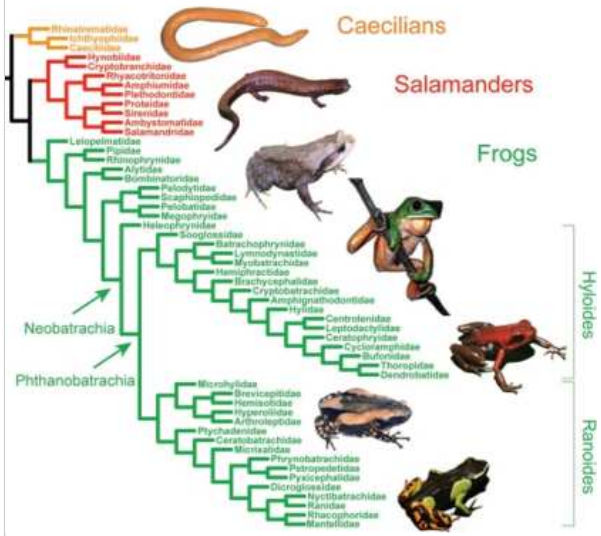
4 Easy Steps to Structure Highly Unstructured Big Data, via Automated Indexation

- Posted by Vincent Granville on February 28, 2016 at 12:00pm
- Send Message View Blog

You have gathered gigabytes or terabytes of unstructured text, for instance scraping the Internet, or pieces of email from your employees or users, or tweets, or millions of products that you want to categorize (only product description and product name is available - sometimes with typos). Now you want to make sense of it, and extract value, possibly design a nice search engine so that your customers can easily find your products. The core algorithm that you need is an automated cataloguer, also called indexer. I am going to explain in layman's terms how it works. First, let's assume that the data consists of

- pages or articles (a web page or the body of an email, etc.)
- subject lines (or page titles),
- and authors (for a web page or an email).

Typically, these "pages" are stored as large repositories containing millions or billions of (sometimes compressed) text files spread across a number of folders and sub-folders, or multiple servers. Sometimes a time stamp is attached to each document, and can be leveraged to increase the accuracy of the indexer.



Even if you only have pages (no user information, no titles), it will work. If you have pages and authors, you can classify the pages separately, then the authors separately (or in parallel), then blend the results to maximize accuracy. The same indexation algorithm (sometimes called tagging algorithm) is used in both cases. Despite the fact that classifying billions of documents seems mathematically unfeasible due to the computational complexity of traditional clustering algorithms (the time spent to cluster is growing much faster than linearly, as a function of the size of your repository), this algorithm is different, run very fast, and is easy to implement using a distributed architecture.

The indexer algorithm creates a taxonomy of your pages (or products, articles, documents etc.) Each page is assigned a category and sub-category.

Indexation algorithm

- *Step 1*: Create a data dictionary (that is, a frequency table) of all one-token and two-token keywords found in all pages (both in the title and in the body of the article). This assumes that you crawled all your articles to extract all the text.
- *Step 2*: Filter / clean results. Ignore keywords with less than 5 occurrences. Check all n-grams of a keyword (*data science* and *science data*) and eliminate n-grams with low frequency, for each keyword
- *Step 3*: Look at top 300 entries, called *seed keywords*. Manually assign seed keywords to 10-20 categories, (these categories are manually pre-selected, after looking at the top 300 entries.) For instance, the top category *data plumbing* will have the following seed keywords: data engineer, data architect, data warehouse,

Hadoop, Spark, data lakes, IoT and many more. Don't forget to have a top category called *Unknown*.
- *Step 4*. Based on keywords found in the title and body of an article, assign the article in question to the top category that has the biggest overlap with the article, in terms of seed keywords. Note that keywords found in the title might be assigned a higher weight than those found in the body. Likewise, a different weight can be attached to each seed keyword, in each top category.

I call this technique *indexation* because it is very similar to the creation of a search engine. We also have used and described this technique in the context of clustering thousands of data science websites (source code provided). This is a must-read article to get a better idea of the technical implementation.

Potential improvement

These improvements will improve the performance (accuracy).

- Add 3-token keywords in your dictionary, not just 1- and 2-token. For 3 tokens keywords, you have 3! (factorial 3) = 6 n-grams. Usually, only one or two of these 6 n-grams will show up in the articles, for any keyword (*data science central* will show up, but *central science data* won't).
- Use stop words to clean your data. Examples: it, where, how, why, for and so on. Be careful though: *IT Job* can not be reduced to *Job* by filtering out the token *IT*. You can replace plurals by singular, and normalize the keywords..
- Some one-token words don't make sense. Do not break San Francisco in San and Francisco. Used a table of keywords that should not be split.

Even without improvements, the methodology will work well, because you focus on top keywords in terms of frequency. For instance, in *Best San Francisco Hotels*, the keywords *Best San* and *Francisco Hotels* won't show up at the top, and if they do, you can remove them, as you manually review the top 3,000 entries (a process that takes 30 minutes).

Finally, the last search engine company I worked for relied on the BerkeleyDB open source software (combined with a bunch of lookup tables such as stop keywords, synonyms and so on) to do many of these tasks. Though it just take a few hours to write your own code.

About the author:

Vincent Granville worked for Visa, eBay, Microsoft, Wells Fargo, NBC, a few startups and various organizations, to optimize business problems, boost ROI or to develop ROI attribution models, developing new techniques and systems to leverage modern big data and deliver added value. Vincent owns several patents, published in top scientific journals, raised VC funding, and founded a few startups. Vincent also manages his own self-funded research lab, focusing on simplifying, unifying, modernizing, automating, scaling, and dramatically optimizing statistical techniques. Vincent's focus is on producing robust, automatable tools, API's and algorithms that can be used and understood by the layman, and at the same time adapted to modern big, fast-flowing, unstructured data. Vincent is a post-graduate from Cambridge University.

DSC Resources

- Career: Training | Books | Cheat Sheet | Apprenticeship | Certification | Salary Surveys | Jobs
- Knowledge: Research | Competitions | Webinars | Our Book | Members Only | Search DSC
- Buzz: Business News | Announcements | Events | RSS Feeds
- Misc: Top Links | Code Snippets | External Resources | Best Blogs | Subscribe | For Bloggers

Additional Reading

- What statisticians think about data scientists
- Data Science Compared to 16 Analytic Disciplines
- 10 types of data scientists
- 91 job interview questions for data scientists
- 50 Questions to Test True Data Science Knowledge
- 24 Uses of Statistical Modeling
- 21 data science systems used by Amazon to operate its business
- Top 20 Big Data Experts to Follow (Includes Scoring Algorithm)
- 5 Data Science Leaders Share their Predictions for 2016 and Beyond
- 50 Articles about Hadoop and Related Topics
- 10 Modern Statistical Concepts Discovered by Data Scientists
- Top data science keywords on DSC
- 4 easy steps to becoming a data scientist
- 22 tips for better data science
- How to detect spurious correlations, and how to find the real ones
- 17 short tutorials all data scientists should read (and practice)
- High versus low-level data science

Follow us on Twitter: @DataScienceCtrl | @AnalyticBridge

Views: 17736

Like
15 members like this

Share Tweet Facebook

Like 3

- < Previous Post
- Next Post >

Comment

Visual Mode HTML Editor

10pt

Follow – Email me when people comment