Web Search Engines and Linear Algebra

Γκόλφος Γεώργιος

April 25, 2021

1 Introduction

Nearly all the major Web search engines today use link analysis to improve their search results. That's exciting for linear algebraists because link analysis, the use of the Web's hyperlink structure, is built from fundamentals of matrix theory. Link analysis and its underlying linear algebra have helped revolutionize Web search, somuch so that the pre-link analysis search (before 1998) pales in comparison to today's remarkably accurate search.

HITS [13] and PageRank [2, 3] are two of the most popylar link analysis algorithms. Both were developed around 1998 and both have dramatically improved the search business. In order to appreciate the impact of link analysis, recall for a minute the state of search prior to 1998. Because of the immense number of pages on the Web, a query to an engine often produced a very long list of elevant pages, sometimes thousands of pages long. A user had to sort carefully through the list to find the most relevant pages. The order of presentation of the pages was little help because spamming was so easy then. In order to trick a search engine into producing rankings higher than normal, spammers used meta-tags liberally, claiming their page used popular search terms that never appeared in the page. Meta-tags became useless for search engines. Spammers also repeated popular search terms in invisible text (white text on a white background) to fool engines.

2 The HITS Algorithm

HITS[13], a link analysis algorithm developed by Jon Kleinberg from Cornell University during his post-doctoral studies at IBM Almaden, aimed to focus this long, unruly query list. The HITS algorithm is based on a pattern Kleinberg noticed among Web pages. Some pages serve as hubs or portal pages, i.e., pages with many outlinks. Other pages are authorities on topics because they have many inlinks. Kleinberg noticed that "good hubs seemed to point to good authorities and good authorities were pointed to by good hubs." So he decided to give each page i both an hub score h_i and an authority score a_i. In fact, for every page i he defined the hub score at iteration $k, h_i^{(k)}$, as

$$a_i^{(k)} = \sum_{j:e_{ji}\in E} h_j^{(k-1)} \tag{1}$$

and

$$h_i^{(k)} = \sum_{j:e_{ij}\in E} a_j^{(k)} \text{ for k=1,2,3,}\ldots, \tag{2}$$

where e_{ij} represents a hyperlink from page i to page j and E is the set of hyperlinks. To compute the scores for a page , he started with uniform scores for all pages, i.e.,

$$h_i^{(1)} = 1/n \tag{3}$$

and

$$a_i^{(1)} = 1/n \tag{4}$$

where n is the number of pages in a so-called neighborhood set for the query list. the neighborhood set consists of all pages in the query list plus all pages pointing to or from the query pages. Depending on the query, the neightborhood set could contain just a hundred pages or a hundred thousand pages. (The neighborhood set allows latent semantic associations

to be made.) The hub and authority scores are iteratively refined until convergence to stationary values.

Using linear algebra we can replace the summation equations with matrix equations. Let h and a be column vectors holding the hub and authority scores. Let L be the adjacency matrix for the neighborhood set. That is, $L_{ij} = 1$ if page i links to page j. and 0, otherwise. These definitions show that

$$a^k = L^T h^{(k-1)} \text{ and } h^{(k)} = La^{(k)}. \tag{5}$$

Using some algebra, we have

$$a^{(k)} = L^T La^{(k-1)} \tag{6}$$

$$h^{(k)} = LL^T h^{(k-1)}. \tag{7}$$

These equations make it clear that Kleinberg's algorithm is really the power method applied to the positive semi-definite matrices $L^T L$ and LL^T. $L^T L$ is called the hub matrix and LL^T is the authority matrix. Thus, HITS amounts to solving the eigenvector problems $L^T La = \lambda_1 a$ and $LL^T h = \lambda_1 h$, where λ_1 is the largest eigenvalue of $L^T L(andLL^T)$, and a and h are corresponding eigenvectors.

While this is the basic linear algebra required by the HITS method, there are many more issues to be considered. Fore example, important issues include convergence, existence, uniqueness, and numerical computation of these scores [5,7,14]. Several modifications to HITS have been suggested, each bringing various advantages and disadvantages [4,6,8]. A variation of kleinberg's HITS concept is at the base of the search engine TEOMA (http://www.teoma.com), which is owned by Ask Jeeves, Inc.

3 The PageRank Algorithm

PageRank, the second link analysis algorithm from 1998, is the heart of Google. Both PageRank and Google were conceived by Sergey Brin and Larry Page while they were computer science graduate students at Stanford University. Brin and Page use a recursive scheme similar to Kleinberg's. Their original idea was that a "page is important if its pointed to by other important pages". That is, they decided that the importance of your page (its PageRank score) is determined by summing the PagerRanks of all pages that point to yours. In building a mathematical definition of PageRank, Brin and Page also reasoned that when an important page points to several places, its weight (PageRank) should be distributed proportionately. In other words, if YAHOO! points to your Web page, that's good, but you shouldn't receive the full weight of YAHOO! because they point to many other places. If YAHOO! points to 999 pages in addition to yours, then you should only get credit for 1/1000 of YAHOO!'s PageRank.

This reasoning led Brin and Page to formulate a recursive definition PageRank. They defined

$$r_i^{(k-1)} = \sum_{j \in I_i} \frac{r_j^{(k)}}{|O_j|}, \tag{8}$$

where $r_i^{(k)}$ is the PageRank of page i at iteration k, I_i is the set of pages pointing into page i and $|O_j|$ is the number of outlinks from page j. Like HITS, PageRank starts with a uniform rank for all pages, i.e., $r_i^0 = 1/n$ and successively refines these scores, where n is the total number of Web pages.

Like HITS, we can write this process using matrix notation. Let the row vector $\pi^{(k)T}$ be the PageRank vector at the k^{th} iteration. As a result, the summation equation for PageRank can be written compactly as

$$\pi^{(k+1)T} = \pi^{(k)T} H,$$

where H is a row normalized hyperlink matrix, i.e., $h_{ij} = 1/|O_i|$, if there is a link from page i to page j, and 0, otherwise. Unfortunately, this iterative procedure has convergence problems-it can cycle or the limit may be dependent on the starting vector.

To fix these problems, Brin and Page revised their basic PageRank concept. Still using the hyperlink structure of the Web, they build an irreducible aperiodic Markov chain characterized by a primitive (irre-

ducible with only one eigenvalue on the spectral circle) transition probability matrix. The irreducibility guarantees the existence of a unique stationary distribution vector π^T, which becomes the PageRank vector. The power method with a primitive stochastic iteration matrix will always converge to π^T independent of the starting vector, and the asymptotic rate of convergence is governed by the magnitude of the subdominant eigenvalue λ_2 of the transition matrix [19].

Here's how Google turns the hyperlink structure of the Web into a primitive stochastic matrix. If there are n pages in the Web, let H be the $n \times n$ matrix whose element h_{ij} is the probability of moving from page i to page j in one click of the mouse. The simplest model is to take $h_{ij} = 1/|O_i|$, which means that starting from any Web page we assume that it is equally likely to follow any of the outgoing links to arrive at another page.

However, some rows of H may contain all zeros, so H is not necessarily stochastic. This occurs whenever a page contains no outlinks; many such pages exist on the Web and are called "dangling nodes". An easy fix is to replace all zero rows with $e^{T/n}$, where e^T is the row vector of all ones. The revised (now stochastic) matrix S can be written as a rank-one update to the sparse H. Let a be the dangling node vector in which

$$a_i = \begin{cases} 1 \text{ if page i is a dangling node,} \\ 0 \text{ otherwise.} \end{cases} \quad (9)$$

Then, $S = H + ae^{T/n}$.
Actually, any probability vector $p^T > 0$ with $p^T e = 1$ can be used in place of the uniform vector $e^{T/n}$.

We're not home yet because the adjustmnet that produces the stochastic matrix S isn't enough to insure the existence of a "unique" stationary distribution vector (needed to make PageRank well defined). Irreducibility on top of stochasticity is required. But the link structure of the Web is reducible-the Web graph is not strongly connected. Consequently, an adjustment to make S irreducible is needed. This last adjustment brings us to "Google matrix", which is defined to be

$$G = aS + (1 - a)E,$$

where $0 \leq a \leq 1$ and $E = ee^{T/n}$. Google eventually replaced the uniform vector $e^{T/n}$ with a more general probability vector v^T (so that $E = ev^T$) to allow them the flexibility to make adjustments to PageRanks as well as to personalize them. See [10,15] for more about the personalization vector v^T.

Because G is a convex combination of the two stochastic matrices S and E, it follow that G is both stochastic and irreducible. Furthermore, every node is now directly connected to every other node (although the probability of transition may be very small in some cases), so G¿0. Consequently, G is a primitive matrix, and this insures that the power method $\pi^{(k+1)T} = \pi^{(k)T}G$ will converge, independent of the starting vector, to a unique stationary distribution π^T [19]. This is the mathematical part of Google's PageRank vector.

There are more to discuss about Google's Power method which is a computational method of choice and this brief introduction describes only the mathematical component of Google's ranking system. However, it's known that there are non-mathematical "metrics" that are also considered when Google responds to a query, so the results seen by a user are in fact PageRank tempered by other metrics.

4 References

References

[1] Jon Kleinberg, *Authoritative sources in an hyperlinked environment.* Journal of the ACM, 46, 1999

[2] Sergey Brin and Lawrence Page. *T*he anatomy of large-scale hypertextual web search engine. Computer networks and ISDN System, 33:107-117, 1998.

[3] Michael W.Berry and Murray Browne. *U*nderstanding Search Engines: Mathematical Modeling ans Text Retrieval.

[4] Sergey Brin, Lawrence Page, R. Motwami, and Terry Winograd. *T*he PageRank citaton ranking: bringing order to the web. Technical report, Com-

puter Science Department, Stanford University, 1998