Google and *HITS* Search Engines
Summary of Lecture 2
Data Mining, Fall 2002
Prof. Amos Fiat

Vered Kunik — ID 025483819

Dudi Deutscher — ID 033148628

October 2002

This lecture summary will discuss the foundations on which the search engines *Google* and *HITS* (Hyperlink Induced Topic Search) are based on.

The basic idea, for both engines, is grading each page in the set a-priori, independently of any specific query. Both algorithms try to assess the "importance" of each page, with the final goal of giving the users a better answer, i.e. one that has the better pages at the top of the results list. These algorithms are based on the assumption that the vast link structure of the Internet is an indicator of an individual page's quality (basically, and this idea will be shortly explained and made precise, a page that is referenced a lot is more important or "better"). They also totally ignore the page contents in the definition of quality.

The Google Search Engine

Introduction

We start by discussing the *PageRank* algorithm, the foundation of the *Google* search engine. *Google* is a next generation search engine, that became very popular on the Internet since its creation a few years ago (a fact that empirically proves its qualities). *Google* is based on an algorithm called *PageRank* [1].

PageRank relies on the uniquely democratic nature of the web by using it's hyper-links as an indicator of page values. In essence, *Google* interprets a link from page A to page B as a vote, by page A, for page B. But, *Google* looks at more than the sheer volume of votes, or links a page receives: it also analyzes the page that casts the vote. In other words, a page score is based on two factors not generally included in search engine rankings: the number of links referencing the page and the "importance" of the referencing pages themselves. Votes cast by pages that are themselves "important" weigh more heavily and help to make other pages "important".

The result of the *PageRank* mechanism is that important, high-quality pages receive a higher *PageRank*, which places them nearer the top of search results.

Of course, important pages mean nothing to the user if they don't match the query, so *Google* combines *PageRank* with text-matching techniques to find pages that are both important and relevant to the search.

Notice that the actual implementation of the commercial search engine adds many modifications to the basic *PageRank*, most of them are kept as commercial secrets. Among others, it adds the "Anchor" ("¡A HREF¿") HTML text to the referenced page, considers font sizes etc.

Realization

Let's make these notions more concrete. Denote the "importance" or "quality" of a page B as $Q(B)$. Given that exactly pages A and C reference our page B, we define

$$Q(B) = \frac{Q(A)}{outdegree(A)} + \frac{Q(C)}{outdegree(C)}$$

where outdegree(A) is the number of links appearing on page A. *Google* compute ranks that solve this equation for all the pages. This definition is of course recursive, and the problem now is whether a solution exists, and how to compute it (how to compute the ranks for all the pages). We will show that this problem is equivalent to the problem of finding the first left eigenvector of a stochastic matrix.

Let us number all the pages in the set (the Internet). Notice that with this notation, the "*Google* equation" is now

$$Q(i) = \sum_{j \text{ referencing } i} \frac{Q(j)}{outdegree(j)} \tag{1}$$

Now define the links matrix $M_{n,n}$, where

$$M_{i,j} = \begin{cases} 1 & \text{if page } i \text{ reference (link to) page } j \\ 0 & \text{otherwise} \end{cases}$$

and obviously $\sum_{j=0}^{n} M_{i,j} = outdegree(i)$ for all i.

Our description resembles a *finite Markov chain*, a process of randomly moving between states, that consists of a finite set of states and some known probabilities $p_{i,j}$ of moving from state i to state j. We transform M into a stochastic matrix (where the sum of the components on every row is 1):

$$\tilde{M}_{i,j} = \begin{cases} \frac{1}{outdegree(i)} & \text{if page } i \text{ reference (link to) page } j \\ 0 & \text{otherwise} \end{cases}$$

so $\tilde{M}$ now describes the probabilities of moving from a given page i, to any other page j ($p_{i,i} = 0$), if the motion is random. $\tilde{M}$ is called the *Transition Matrix* of the finite Markov process.

Starting with given probabilities of being at each state, and repeating the Markov step infinitely many times (with certain conditions applying, as can be achieved in our case), the Markov process settles into a steady condition, where the probability of being at a given state i is constant. These are called the *stationary probabilities*, and we will shortly prove that for the Markov process represented by the matrix $\tilde{M}$, these are exactly the solutions of the *Google* equation (1).

A few remarks on eigenvalues & eigenvectors

Only the observations that we will need.

- Definition: v is a *left eigenvector* of a matrix M if there exists a scalar λ for which $vM = \lambda v$.

- Definition: The *first eigenvalue* of M is

$$\max_{v} \lambda \text{ (where } vM = \lambda v, \|v\|_2 = 1)$$

Geometrically, these are the direction and magnitude of the maximum stretch that M produces on the linear space.

Finding them algorithmically can be achieved, among other ways, using the *Power method*: Randomly choose a vector x and multiply it with the matrix M over and over again (normalizing the result at each step to keep using representable numbers). The result will converge into the first eigenvector of M. (Intuitive reason: since λ is the largest of all eigenvalues, it grows the fastest when powered, and soon overshadows all the others). Notice the rare possibility that x will happen to be perpendicular to the first eigenvector, and the power method will fail. Another way (that is theoretically sound, but fails numerically) relies on the fact that

$$Mv = \lambda v \Leftrightarrow (M - \lambda I)v = 0$$

and so the eigenvalues are the roots of the characteristic polynomial of M.

Wrapping it up and final proofs

To complete the discussion on *Google*, we will show that:

1. The stationary probabilities exist, i.e. that the Markov process described by our stochastic-links-matrix $\tilde{M}$, settles into a steady condition.

2. The stationary probabilities form the first left eigenvector of $\tilde{M}$, and thus can be computed using the power method.

3. These stationary probabilities are the solution to the *Google* equation (1).

Proving the first point is beyond the scope of our discussion, but here is an intuitive remark - one of the problems preventing the existence of the stationary probabilities are "sinks": pages without any links. These "attract" probability that can not continue to other areas of the net. *Google* handles this and other problems by adding some "noise" - a matrix that is full with some small ϵ values. One can think of this as some small probability that the process will jump to a random new page, on each step.

Considering the second point: for the stationary probabilities $q = (q_1, q_2, \ldots, q_n)$ the following holds:

$$\forall i \sum_j q_j p_{j,i} = q_i$$
$$\sum_i q_i = 1$$

where

$$P = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,n} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,n} \\ \vdots & \vdots & & \vdots \\ p_{n,1} & p_{n,2} & \cdots & p_{n,n} \end{pmatrix}$$

is the transition matrix. The first equation states that the probability of being at a state i (q_i) is the sum on all states j of previously being at j (q_j) and moving to i ($p_{j,i}$).

If we use $\tilde{M}$ as the transition matrix, and writing in matrix form, we get

$$q = q\tilde{M} \tag{2}$$

so, q is a left eigenvector of $\tilde{M}$, matching the eigenvalue λ=1. We take it without proof that the first eigenvalue of a *stochastic* matrix is 1, and that all the other eigenvalues are properly less, and therefore q is actually the first left eigenvector of $\tilde{M}$. Thus, we have proved the second point above.

For our final debt, rewriting equation 2 explicitely we get:

$$q = (q_1, q_2, \ldots, q_n) = (q_1, q_2, \ldots, q_n) \begin{pmatrix} \tilde{M}_{1,1} & \tilde{M}_{1,2} & \ldots & \tilde{M}_{1,n} \\ \tilde{M}_{2,1} & \tilde{M}_{2,2} & \ldots & \tilde{M}_{2,n} \\ \vdots & \vdots & & \vdots \\ \tilde{M}_{n,1} & \tilde{M}_{n,2} & \ldots & \tilde{M}_{n,n} \end{pmatrix}$$

or

$$\begin{aligned} q_1 &= q_1\tilde{M}_{1,1} + q_2\tilde{M}_{2,1} + \cdots + q_n\tilde{M}_{n,1} \\ q_2 &= q_1\tilde{M}_{1,2} + q_2\tilde{M}_{2,2} + \cdots + q_n\tilde{M}_{n,2} \\ &\vdots \\ q_n &= q_1\tilde{M}_{1,n} + q_2\tilde{M}_{2,n} + \cdots + q_n\tilde{M}_{n,n} \end{aligned}$$

but remembering that

$$\tilde{M}_{i,j} = \begin{cases} \frac{1}{outdegree(i)} & \text{if page } i \text{ reference (link to) page } j \\ 0 & \text{otherwise} \end{cases}$$

we see that equation 2 is exactly our earlier definition of the the "*Google* equation" (equation 1).

We conclude that the first left eigenvector of the stochastic matrix $\tilde{M}$, is the vector of stationary probabilities of the Markov process described by $\tilde{M}$, and is the solution to the *Google* equation presented earlier. By finding the first left eigenvector of the stochastic matrix (using the power method, for example) we compute the rankings of the entire set of pages.

The HITS search engine algorithm

The *HITS* (Hyper-link Induced Topic Search) ([2, 3]) search engine algorithm is very similar to the algorithm employed by Google, only it was not implemented commercially, but was more thoroughly researched academically. *HITS* uses refinements on the idea of a-priori computed "importance" as a tool for ordering the search results. Its definition of "importance" is composed of two scores - the *authority* score: how good is the page in its field, and the *hub* score: how good is the page as a list of references to other pages. Translating these ideas to algebraic equations, HITS defines recursively:

$$hub(i) = \sum_{u \to v} auth(v)$$

$$auth(i) = \sum_{v \to u} hub(v)$$

and sorts pages according to their *auth* score.

It turns out that the HITS formulae are solved using a different algebraic concept, the SVD, which we will now discuss:

SVD - Singular Values Decomposition

Definition: The *reduced-* (or thin-) *Singular Value Decomposition* (SVD) is a decomposition of a matrix $M_{m \times n}$ with the following form:

$$M_{m \times n} = U_{m \times k} \cdot \Sigma_{k \times k} \cdot V^T_{k \times n}$$

$(k = rank(M))$ where

1. If $\{u_i\}$ are the column vectors of U, then $\forall i \neq j \; \|u_i\| = 1, u_i \perp u_j$, i.e U is orthonormal.
2. Σ is a diagonal matrix, where $\{\sigma_i\}$ are the components on the main diagonal, such that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k$.
3. If $\{v_i^T\}$ are the row vectors of V^T, then $\forall i \neq j \; \|v_i^T\| = 1, v_i^T \perp v_j^T$, i.e V is also orthonormal.

The full SVD is obtained when enlarging the matrixes by setting $k = \min m, n$. This is achieved by adding zero-valued singular values (and choosing appropriate singular vectors so the orthonormality of U and V is maintained).

It can be shown that every matrix has an SVD, it is unique, and that for all $1 \leq i \leq k$:

1. $u_i^T M = \sigma_i v_i^T$
2. $M v_i = \sigma_i u_i$

An alternative definition: A right-hand singular vector is the unit vector that M enlarges the most, i.e it is the direction which M pulls the most (and optionally rotates - this is the difference between singular and eigenvectors).

More formally:

$$\sigma_1 = \max_{\|v\|=1} \|Mv\|$$

$$\sigma_{i+1} = \max_{\|v\|=1,\; v \perp \sum_{j=1}^{i} \alpha_i v_j} \|Mv\|$$

and v_i are the corresponding vectors.

Back to HITS

As mentioned before, and without explicitly proving it (though the proofs are very similar to those we gave for *Google* and eigenvectors), the HITS formulas are solved using SVD: the first left and right singular vectors are exactly the *auth* and *hub* scores respectively.

References

[1] S.Brin and L.Page, *"The Anatomy of a Large-Scale Hypertextual Web Search Engine"*, Proc. 1998 WWW Conf., 1998

[2] J.Kleinberg, *"Authoritative sources in a hyperlinked environment"*, Research Report RJ 10076 (91892), IBM, 1997

[3] J.Kleinberg, *"Hubs, Authorities, and Communities"*, ACM Computing Surveys, 31(4es, Article No.5), 1999