The Google PageRank Algorithm

Jamie Arians
College of William and Mary

What is the Google PageRank Algorithm?

Background Knowledge

In 1989 The World Wide Web (the internet) was invented by Tim Berners Lee
It revolutionized information access and storage
However searching for specific information was very difficult and inefficient

In 1998 link analysis revitalized computerized information retrieval

All the successful search engines adopted link analysis

Define:
Link analysis: a technique that exploits the information inherent in the hyperlink structure of the internet

Search engine: a virtual machine created by software which enables them to sift through virtual files to find the most relevant documents to a query

There are 3 traditional search engine models for searching information collections like the internet:
Boolean Search Engine Model
Vector Space Search Engine Model
Probabilistic Search Engine Model

Boolean Search Engine Model

This model uses exact matching to find relevant documents by considering what keywords are present or absent in a document

Documents are considered relevant or irrelevant, there is no in-between

Drawbacks:

- Synonymy
- Polysemy

Vector Space Search Engine Model

Transforms textual data into numeric vectors and matrices

This model uses matrix analysis techniques to discover key features and connections in the document collection

Allows documents to partially match a query by assigning it a number between 0 and 1 which represents the likelihood of relevance of the query

Retrieved documents are returned in an ordered list by degree of relevance

Drawback:

- There is a very large computational expense because each documents relevance score must be computed and ordered individually

Probabilistic Model Search Engines

This model attempts to estimate the probability a user will find a particular document relevant

Retrieved documents are ranked by odds of relevance

The ratio of probability that the document is relevant is divided by the ratio of probability that the document is not relevant

The search engine operated recursively and requires the algorithm to guess at initial parameters then iteratively tries to improve the guess to obtain a final relevancy rating

A major positive about this model is that the algorithm can keep track of a user's query history and use that to build a better initial guess
Drawback:

- This algorithm very hard to build and program
- The complexity grows very quickly

Meta-Search Engines

There is actually a 4th search engine model

Meta-search combines the 3 classic models
One search engine is good but 2 or more is better

This model sends the query to several search engines and returns the results in one long unified list

Comparing Search Engines

There are 2 common ratings used to differentiate search engine models

- Precision: ratio of relevant documents to total documents
- Recall: ratio of total relevant documents to documents retrieved

Hyperlink Structure

Link analysis looks at information inherent in the hyperlink structure of the internet
So what is the hyperlink structure?

It is basically a massive digraph

- nodes are web pages and directed arcs are the hyperlinks
- hyperlinks pointing to a page are inlinks
- hyperlinks going from a page are outlinks

Hyperlinks are created when one web page references another

PageRank

The PageRank algorithm uses the hyperlink structure to determine the importance and relevance of web pages

PageRank views the hyperlinks as recommendations

A page with more recommendations (inlinks) is more important than a page with few inlinks

A page is considered more important if it is pointed to by other important pages

Web pages are given a score 0-10 with 10 being the most important and 0 being spam pages

PageRank is query independent so it is faster and more effiecient than query dependent algorithms

Query independent:the popularity score is determined offline so at runtime no time is spent computing the popularity scores for webpages

PageRank (cont.)

PageRank began with a summation equation, in which the PageRank of a page P_i, denoted $r(P_i)$, is the sum of the PageRanks of all pages pointing into P_i

$$r(P_i) = \sum_{P_j \in B_{P_i}} \frac{r(P_j)}{|P_j|}$$

Where B_{P_i} is the set of pages pointing into P_i and $|P_j|$ is the number of outlinks from page P_j

Iterative PageRank

The problem with this initial equation is that the $r(P_j)$ values are unknown. To counteract this we use an iterative procedure
We assume at the begining that all pages have equal PageRank (say $\frac{1}{n}$, where n is the number of pages in the index of the web)
The rule in the previous equation is applied to compute $r(P_i)$ for every P_i in the index then the values of the previous iterate are substituted into $r(P_j)$

So let $r_{k+1}(P_i)$ be the PageRank of page P_i at iteration $k+1$
Then

$$r_{k+1}(P_i) = \sum_{P_j \in B_{P_i}} \frac{r_k(P_j)}{|P_j|}$$

We initiate this process with $r_0(P_i) = \frac{1}{n}$ for all pages P_i and repeat hoping that the PageRank scores will eventually converge to some final stable values

Example

Iteration 0	Iteration 1	Iteration 2	Rank at Iteration 2
$r_0(P_1) = \frac{1}{6}$	$r_1(P_1) = \frac{1}{18}$	$r_2(P_1) = \frac{1}{36}$	5
$r_0(P_2) = \frac{1}{6}$	$r_1(P_2) = \frac{5}{36}$	$r_2(P_2) = \frac{1}{18}$	4
$r_0(P_3) = \frac{1}{6}$	$r_1(P_3) = \frac{1}{12}$	$r_2(P_3) = \frac{1}{36}$	5
$r_0(P_4) = \frac{1}{6}$	$r_1(P_4) = \frac{1}{4}$	$r_2(P_4) = \frac{17}{72}$	1
$r_0(P_5) = \frac{1}{6}$	$r_1(P_5) = \frac{5}{36}$	$r_2(P_5) = \frac{11}{72}$	3
$r_0(P_6) = \frac{1}{6}$	$r_1(P_6) = \frac{1}{6}$	$r_2(P_6) = \frac{14}{72}$	2

So after iteration 2, P_4 has the highest PageRank score and is considered the most important and P_1 and P_3 are tied for the least important

Matrix Representation

The summation equations compute PageRank one page at a time, but with matrices we can compute a page rank vector at each iteration

This uses a single 1xn vector to hold the PageRank values for all pages in the index.

To do this, we introduce an nxn matrix $\mathbf{H}$ and a 1xn row vector π^T
$\mathbf{H}$ is a row normalized hyperlink matrix with $\mathbf{H_{ij}} = \frac{\mathbf{1}}{\mathbf{|P_i|}}$ if there is a link from node i to node j and 0 otherwise

Consider again our example web of six pages

The $\mathbf{H}$ matrix for this graph is

$$\mathbf{H} = \begin{array}{c} \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{array} \begin{array}{c} \begin{array}{cccccc} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \end{array} \\ \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Matrix Representation (cont.)

$$\mathbf{H} = \begin{array}{c} \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{array} \begin{array}{c} \begin{array}{cccccc} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \end{array} \\ \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

The nonzero elements of row i correspond to the outlinking pages of page i, whereas the nonzero elements of column i correspond to the inlinking pages of page i.

Now we introduce a row vector $\pi^{(k)T}$, which is the PageRank vector at the kth iteration

Using this matrix notation, the previous summation equation can be written as

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

Observations

- Each iteration requires one vector-matrix multiplication, which on average requires $O(n^2)$ time complexity, where n is the size of the square matrix $\mathbf{H}$
- $\mathbf{H}$ is a very sparse matrix, because most pages only link to a few other pages

  This is welcome because sparse matrices require much less time complexity than a $O(n^2)$ dense computation.

  On average, sparse matrix multiplication requires $0(nnz(\mathbf{H}))$ time complexity where $nnz\mathbf{H}$ is the number of nonzeros in $\mathbf{H}$

  The average webpage has about 10 links so $\mathbf{H}$ has about $10n$ nonzeros so the vector-matrix multiplication reduces to $0(n)$ time complexity
- The iterative process of $\pi^{(k+1)T} = \pi^{(k)T}\mathbf{H}$ is a simple linear stationary process, in fact it is the classical power method applied to $\mathbf{H}$
- $\mathbf{H}$ is very similar to a stochastic transition probability matrix for a Markov chain.

  The dangling nodes of the network, those nodes with no outlinks, create 0 rows in the matrix. All the other rows, which correspond to nondangling nodes create stochastic row. Thus $\mathbf{H}$ is substochastic

Questions and Problems with Iterative Process

Questions:

- Will the iterative process continue indefinitely or converge?
- Under what circumstances is it guaranteed to converge?
- Will it converge to something that makes sense in the context of the PageRank problem?
- Will it converge to just one or multiple vectors?
- Does the convergence depend on the starting vector $\pi^{(0)T}$?
- If it will converge, how many iterations will it take?

Problems:

- rank sinks
- cycles

Markov Chain Theory

In our observations, we noted that the equation resembled the power method applied to a Markov chain with transition probability matrix $\mathbf{H}$

With Markov Theory we can make adjustments to the equation that insure desirable results, convergence properties, and answers to the our questions

In particular we know that for any starting vector, the power method applied to a Markov matrix $\mathbf{P}$ converges to a unique positive vector called the stationary vector as long as $\mathbf{P}$ is stochastic, irreducible, and aperiodic (Aperiodicity plus irreducability implies primitivity)

Therefore the Convergence problems caused by rank sinks and cycles can be overcome if $\mathbf{H}$ is modified so that it is a Markov matrix with these properties

Adjustments

First, what we call the stochasticity adjustment because the 0^T rows of $\mathbf{H}$ are replaced with $\frac{1}{n\mathbf{e^T}}$, making $\mathbf{H}$ stochastic
For our example, the stochastic matrix $\mathbf{S}$ is

$$\mathbf{S} = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Mathematically, $\mathbf{S}$ is created from a rank-one update to $\mathbf{H}$, $\mathbf{S} = \mathbf{H} + \mathbf{a}(\frac{\mathbf{1}}{\mathbf{ne^T}}$
where $a_i = 1$ if page i is a dangling node and 0 otherwise

The binary vector a is called the dangling node vector
$\mathbf{S}$ is a combination of the raw original hyperlink matrix $\mathbf{H}$ and a rank-one matrix $\frac{1}{n\mathbf{ae^T}}$

Adjustments (cont.)

The Stochasticity Adjustment guarantees that $\mathbf{S}$ is stochastic but it cannot guarantee that a unique positive π^T exists and that the equation will converge to this π^T quickly

So another adjustment is needed, this time a Primitivity Adjustment.
So we create the Google Matrix $\mathbf{G}$ such that

$$\mathbf{G} = \alpha\mathbf{S} + (\mathbf{a} - \alpha)\mathbf{1}/\mathbf{n}\mathbf{e}\mathbf{e}^{\mathbf{T}}$$

where α is a scalar between 0 and 1

Results of primitivity adjustment

- $\mathbf{G}$ is stochastic, it is the convex combination of the two stochastic matrices $\mathbf{S}$ and $\mathbf{E} = \frac{\mathbf{1}}{\mathbf{nee^T}}$
- $\mathbf{G}$ is irreducible, every page is directly connected to every other page so irreducibility is trivially enforced
- $\mathbf{G}$ is aperiodic. The self-loops ($\mathbf{G_{ii}} > \mathbf{0}$ for all i) create aperiodicity
- $\mathbf{G}$ is primitive because $\mathbf{G^k} > \mathbf{0}$ for some k (This holds for $k = 1$). This implies that a unique π^T exists and the power method applied to $\mathbf{G}$ is guaranteed to converge to this vector
- $\mathbf{G}$ is completely dense, which is very bad computationally. Fortunately $\mathbf{G}$ can be written as a rank-one update to the very sparse hyperlink matrix $\mathbf{H}$ which is very computationally advantageous
- $\mathbf{G}$ is artificial in the sense that the raw hyperlink matrix $\mathbf{H}$ has been modified twice to produce desirable convergence properties. A stationary PageRank vector does not exist for $\mathbf{H}$ but it does exist for $\mathbf{G}$, and is remarkably good at giving a global importance value to webpages

Example

To summarize, Google's adjusted PageRank method is $\pi^{(k+1)T} = \pi^{(k)T}\mathbf{G}$ which is simply the power method applied to $\mathbf{G}$

To finish our example, for $\alpha = .9$, the stochastic primitive matrix $\mathbf{G}$ is

$$\mathbf{G} = .9\mathbf{H} + (.9\begin{pmatrix}0\\1\\0\\0\\0\\0\end{pmatrix} + .1\begin{pmatrix}1\\1\\1\\1\\1\\1\end{pmatrix})\frac{\mathbf{1}}{\mathbf{6}}\begin{pmatrix}1 & 1 & 1 & 1 & 1 & 1\end{pmatrix}$$

$$= \begin{pmatrix} \frac{1}{60} & \frac{7}{15} & \frac{7}{15} & \frac{1}{60} & \frac{1}{60} & \frac{1}{60} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{19}{60} & \frac{19}{60} & \frac{1}{60} & \frac{19}{60} & \frac{1}{60} & \frac{1}{60} \\ \frac{1}{60} & \frac{1}{60} & \frac{1}{60} & \frac{7}{15} & \frac{1}{60} & \frac{7}{15} \\ \frac{1}{60} & \frac{1}{60} & \frac{1}{60} & \frac{11}{12} & \frac{1}{60} & \frac{1}{60} \end{pmatrix}$$

Example (cont.)

Google's PageRank Vector is the stationary vector of $\mathbf{G}$ and is given by

$$\pi^T = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ .03732 & .05396 & .04151 & .3751 & .206 & .2862 \end{pmatrix}$$

The pages in this example web can be ranked by their importance as

$$\begin{pmatrix} 4 & 6 & 5 & 2 & 3 & 1 \end{pmatrix}$$

Meaning 4 is the most important page and 1 is the least important

Thank you for your attention!