PageRank Algorithm

Guxiao Hu, Puyuan Zhang, Xiu Chen, Zipei Zhu

Thursday 4^{th} December, 2025

Table of Contents

How do search engines work?

You are an engineer in Gugloo designing a search engine in the 1990s.

Consider: how to rank results for each query?

How do search engines work?

You are an engineer in Gugloo designing a search engine in the 1990s.

Consider: how to rank results for each query?

- Relevance

How do search engines work?

You are an engineer in Gugloo designing a search engine in the 1990s.

Consider: how to rank results for each query?

- Relevance
  - NLP-focused approaches. May be discussed at the end of this pre (also linear-algebra heavy!).

How do search engines work?

You are an engineer in Gugloo designing a search engine in the 1990s.

Consider: how to rank results for each query?

- Relevance
  - NLP-focused approaches. May be discussed at the end of this pre (also linear-algebra heavy!).
- Importance

How do search engines work?

You are an engineer in Gugloo designing a search engine in the 1990s.

Consider: how to rank results for each query?

- Relevance
  - NLP-focused approaches. May be discussed at the end of this pre (also linear-algebra heavy!).
- Importance
  - Sorting on accessing counts?

How do search engines work?

You are an engineer in Gugloo designing a search engine in the 1990s.

Consider: how to rank results for each query?

- Relevance
  - NLP-focused approaches. May be discussed at the end of this pre (also linear-algebra heavy!).
- Importance
  - Sorting on accessing counts?

PageRank algorithm focuses on *how important* each webpage is.

The "Random Surfer" Model

The PageRank algorithm

- iterates on a **graph** where webpages are **nodes**, and links are **directed edges**,
- calculates the probability of a "random surfer" visiting each webpage,
- outputs a probability distribution vector.

$$\boldsymbol{p} = \begin{bmatrix} \Pr(1) & \Pr(2) & \cdots & \Pr(n) \end{bmatrix}^{\mathsf{T}}$$

Algorithm

Notations and Conventions

- Let n denote the number of nodes (i.e. total number of webpages).
- Let $\deg^+(u)$ denote the **out degree** of node u (i.e. number of outreaching links on webpage u).

Definition

Google PageRank[a] **iteration**: Initially, $\Pr(i) := 1/n$ for each webpage $1 \leq i \leq n$. Then iterate by

$$\Pr(v) := \frac{1-d}{n} + d \sum_{\text{edge } u \to v} \frac{1}{\deg^+(u)} \Pr(u)$$

where d is the **damping factor** ($0 < d < 1$).

[a] Trademark of Google; U.S. patent 6,285,999.

Computing PageRank

- A node is important if important nodes point to it.
- A node contributes part of its importance to the nodes it points to.

Matrix Algebra I

A Simplified Model

Let **transition matrix**

$$\mathscr{M}_{ij} := \begin{cases} \dfrac{1}{\deg^+(j)} & \text{edge } j \to i \\ 0 & \text{otherwise} \end{cases}$$

i.e., given **adjacent matrix** $A_{ij} = [\text{edge } i \to j]$ and diagonal matrix K with the outdegrees in the diagonal,

$$\mathscr{M} := (K^{-1}A)^{\mathsf{T}}$$

Matrix Algebra II

Definition

Let **probability distribution vector** of the k-th iteration be

$$\boldsymbol{p}(k) := \begin{bmatrix} \Pr(1) & \Pr(2) & \cdots & \Pr(n) \end{bmatrix}^{\mathsf{T}}$$

which is initially set to

$$\boldsymbol{p}(0) := \begin{bmatrix} 1/n & 1/n & \cdots & 1/n \end{bmatrix}^{\mathsf{T}}$$

Matrix Algebra III

Definition

The **Google matrix** $\widehat{\mathscr{M}}$ is defined by

$$\widehat{\mathscr{M}} := d\mathscr{M} + \frac{1-d}{n}E$$

where E is $n \times n$ matrix of all ones (so that $E\boldsymbol{p} = \mathbf{1}$), and $0 < d < 1$ is the damping factor.

The Power Method

Google PageRank can be computed by

$$\boldsymbol{p}(k+1) = \widehat{\mathscr{M}}\boldsymbol{p}(k)$$

One may assume PageRank converges after $|\boldsymbol{p}(t) - \boldsymbol{p}(t-1)| < \epsilon$.

PageRank and Eigenvectors

Recall the equation

$$\boldsymbol{p}(k+1) = \widehat{\mathscr{M}}\boldsymbol{p}(k)$$

PageRank and Eigenvectors

Recall the equation

$$\begin{aligned} \boldsymbol{p}(k+1) &= \widehat{\mathscr{M}}\boldsymbol{p}(k) \\ \Rightarrow \boldsymbol{p} &= \widehat{\mathscr{M}}\boldsymbol{p} \quad \text{(when converged)} \end{aligned}$$

PageRank and Eigenvectors

Recall the equation

$$\begin{aligned} \boldsymbol{p}(k+1) &= \widehat{\mathscr{M}}\boldsymbol{p}(k) \\ \Rightarrow \boldsymbol{p} &= \widehat{\mathscr{M}}\boldsymbol{p} \quad \text{(when converged)} \end{aligned}$$

Iteration is "finding a vector that remains unchanged by linear transformation $\widehat{\mathscr{M}}$".

PageRank and Eigenvectors

Recall the equation

$$\begin{aligned} \boldsymbol{p}(k+1) &= \widehat{\mathscr{M}}\boldsymbol{p}(k) \\ \Rightarrow \boldsymbol{p} &= \widehat{\mathscr{M}}\boldsymbol{p} \quad \text{(when converged)} \end{aligned}$$

Iteration is "finding a vector that remains unchanged by linear transformation $\widehat{\mathscr{M}}$".

Definition

$\boldsymbol{v}$ is the **eigenvector**, and λ is the **eigenvalue** of matrix A, if

$$A\boldsymbol{v} = \lambda\boldsymbol{v}$$

PageRank and Eigenvectors

Recall the equation

$$\begin{aligned} \boldsymbol{p}(k+1) &= \widehat{\mathscr{M}}\boldsymbol{p}(k) \\ \Rightarrow \boldsymbol{p} &= \widehat{\mathscr{M}}\boldsymbol{p} \quad \text{(when converged)} \end{aligned}$$

Iteration is “finding a vector that remains unchanged by linear transformation $\widehat{\mathscr{M}}$”.

Definition

$\boldsymbol{v}$ is the **eigenvector**, and λ is the **eigenvalue** of matrix A, if

$$A\boldsymbol{v} = \lambda\boldsymbol{v}$$

In fact, $\boldsymbol{p}$ is the **principle eigenvector** of the Google matrix $\widehat{\mathscr{M}}$, and 1 is the corresponding eigenvalue (i.e. the eigenvalue with the largest magnitude).

Properties

Corollary

1. $\boldsymbol{p}$ *is the principle eigenvector of* $\widehat{\mathscr{M}}$, *and* 1 *is the corresponding eigenvalue (i.e. the eigenvalue with the largest magnitude).*
   By the Perron–Frobenius Theorem,
   - $\boldsymbol{p}$ *is unique, strictly positive*
   - ***iteration converges.***
2. *Both* $\mathscr{M}$ *and* $\widehat{\mathscr{M}}$ *are* ***column-stochastic*** *(sums of each column are 1), positive.*

Therefore, PageRank algorithm converges to steady probability distribution.

Properties

Corollary

1. *$\boldsymbol{p}$ is the principle eigenvector of $\widehat{\mathscr{M}}$, and 1 is the corresponding eigenvalue (i.e. the eigenvalue with the largest magnitude).*
   By the Perron–Frobenius Theorem,
   - *$\boldsymbol{p}$ is unique, strictly positive*
   - ***iteration converges.***
2. *Both $\mathscr{M}$ and $\widehat{\mathscr{M}}$ are* ***column-stochastic*** *(sums of each column are 1), positive.*

Therefore, PageRank algorithm converges to steady probability distribution.
Question: will the simplified model $\boldsymbol{p}(k+1) = \mathscr{M}\boldsymbol{p}(k)$ we define before always converge?

A View on Probability

Matrix multiplication → updating probability.

A View on Probability

Matrix multiplication $\rightarrow$ updating probability.
Reason to use the damping factor?

A View on Probability

Matrix multiplication $\rightarrow$ updating probability.
Reason to use the damping factor? To avoid trapping.

A View on Probability

Matrix multiplication → updating probability.
Reason to use the damping factor? To avoid trapping.

- With a probability of d, “random surfing” continues;
- with a probability of $1 - d$, surfer redirects to another random page.

A View on Probability

Matrix multiplication → updating probability.
Reason to use the damping factor? To avoid trapping.

- With a probability of d, "random surfing" continues;
- with a probability of $1 - d$, surfer redirects to another random page.

Applications

Generalizing *importance*.

Applications

Generalizing *importance*.

- Evaluating academic papers based on citations.
- Determine species that are essential to the continuing health of ecosystems.
- Ranking performance of sports teams.
- ...

Further Reading

Suggested topics for further reading:

- PageRank is actually a variation of **the Markov Chains**.

Further Reading

Suggested topics for further reading:

- PageRank is actually a variation of **the Markov Chains**.
- Some Easy Ways to Briefly Analyze "Relevance":
    - Word frequency (TF / TF-IDF)
    - Proximity scoring (word vectors)
    - Matching phrases (bags-of-words)
    - ...

Thanks!