HITS algorithm

From Wikipedia, the free encyclopedia

Hyperlink-Induced Topic Search (**HITS**; also known as **hubs and authorities**) is a link analysis algorithm that rates Web pages, developed by Jon Kleinberg. The idea behind Hubs and Authorities stemmed from a particular insight into the creation of web pages when the Internet was originally forming; that is, certain web pages, known as hubs, served as large directories that were not actually authoritative in the information that it held, but were used as compilations of a broad catalog of information that led users directly to other authoritative pages. In other words, a good hub represented a page that pointed to many other pages, and a good authority represented a page that was linked by many different hubs.[1]

The scheme therefore assigns two scores for each page: its authority, which estimates the value of the content of the page, and its hub value, which estimates the value of its links to other pages.

Contents

History

In journals

Formerly, many methods were used for ranking the importance of scientific journals. One such method was Garfield's impact factor. However, many journals such as Science and Nature are filled with numerous citations, making these magazines have very high impact factors. Thus, when comparing two more obscure journals which have received roughly the same number of citations but one of these journals has received many citations from Science and Nature, this journal needs be ranked higher. In other words, it is better to receive citations from an important journal than from an unimportant one.[2]

On the Web

This phenomenon also occurs in the Internet. Counts the number of links to a page can give us a general estimate of its prominence on the Web, but a page with very few incoming links may also be prominent, if two of these links come from the home pages of Yahoo! or Google or MSN. Thus, because these sites are of very high importance but are also Search Engines, there can be very irrelevant results. The Twitter Social Network uses a HITS style algorithm to suggest user accounts to follow.[3]

Algorithm

In the HITS algorithm, the first step is to retrieve the most relevant pages to the search query. This set is called the *root set* and can be obtained by taking the top n pages returned by a text-based search algorithm. A *base set* is generated by augmenting the root set with all the web pages that are linked from it and some of the pages that link to it. The web pages in the base set and all hyperlinks among those pages form a focused subgraph. The HITS computation is performed only on this *focused subgraph*. According to Kleinberg the reason for constructing a base set is to ensure that most (or many) of the strongest authorities are included.

Authority and hub values are defined in terms of one another in a mutual recursion. An authority value is computed as the sum of the scaled hub values that point to that page. A hub value is the sum of the scaled authority values of the pages it points to. Some implementations also consider the relevance of the linked pages.

The algorithm performs a series of iterations, each consisting of two basic steps:

- **Authority Update**: Update each node's *Authority score* to be equal to the sum of the *Hub Scores* of each node that points to it. That is, a node is given a high authority score by being linked from pages that are recognized as Hubs for information.
- **Hub Update**: Update each node's *Hub Score* to be equal to the sum of the *Authority Scores* of each node that it points to. That is, a node is given a high hub score by linking to nodes that are considered to be authorities on the subject.

The Hub score and Authority score for a node is calculated with the following algorithm:

- Start with each node having a hub score and authority score of 1.
- Run the Authority Update Rule
- Run the Hub Update Rule
- Normalize the values by dividing each Hub score by square root of the sum of the squares of all Hub scores, and dividing each Authority score by square root of the sum of the squares of all Authority scores.

- Repeat from the second step as necessary.

HITS, like Page and Brin's PageRank, is an iterative algorithm based on the linkage of the documents on the web. However it does have some major differences:

- It is query dependent, that is, the (Hubs and Authority) scores resulting from the link analysis are influenced by the search terms;
- As a corollary, it is executed at query time, not at indexing time, with the associated hit on performance that accompanies query-time processing.
- It is not commonly used by search engines. (Though a similar algorithm was said to be used by Teoma, which was acquired by Ask Jeeves/Ask.com.)
- It computes two scores per document, hub and authority, as opposed to a single score;
- It is processed on a small subset of ‘relevant’ documents (a 'focused subgraph' or base set), not all documents as was the case with PageRank.

In detail

To begin the ranking, $\forall p$, $\mathrm{auth}(p) = 1$ and $\mathrm{hub}(p) = 1$. We consider two types of updates: Authority Update Rule and Hub Update Rule. In order to calculate the hub/authority scores of each node, repeated iterations of the Authority Update Rule and the Hub Update Rule are applied. A k-step application of the Hub-Authority algorithm entails applying for k times first the Authority Update Rule and then the Hub Update Rule.

Authority Update Rule

$\forall p$, we update $\mathrm{auth}(p)$ to be the summation:

$$\mathrm{auth}(p) = \sum_{i=1}^{n} \mathrm{hub}(i)$$

where n is the total number of pages connected to p and i is a page connected to p. That is, the Authority score of a page is the sum of all the Hub scores of pages that point to it.

Hub Update Rule

$\forall p$, we update $\mathrm{hub}(p)$ to be the summation:

$$\mathrm{hub}(p) = \sum_{i=1}^{n} \mathrm{auth}(i)$$

where n is the total number of pages p connects to and i is a page which p connects to. Thus a page's Hub score is the sum of the Authority scores of all its linking pages

Normalization

The final hub-authority scores of nodes are determined after infinite repetitions of the algorithm. As directly and iteratively applying the Hub Update Rule and Authority Update Rule leads to diverging values, it is necessary to normalize the matrix after every iteration. Thus the values obtained from this process will eventually converge.[4]

Pseudocode

```
G := set of pages
for each page p in G do
  p.auth = 1 // p.auth is the authority score of the page p
  p.hub = 1 // p.hub is the hub score of the page p
function HubsAndAuthorities(G)
  for step from 1 to k do // run the algorithm for k steps
    norm = 0
    for each page p in G do  // update all authority values first
      p.auth = 0
      for each page q in p.incomingNeighbors do // p.incomingNeighbors is the set of pages that link to p
         p.auth += q.hub
      norm += square(p.auth) // calculate the sum of the squared auth values to normalise
    norm = sqrt(norm)
    for each page p in G do  // update the auth scores 
      p.auth = p.auth / norm  // normalise the auth values
    norm = 0
    for each page p in G do  // then update all hub values
      p.hub = 0
      for each page r in p.outgoingNeighbors do // p.outgoingNeighbors is the set of pages that p links to
        p.hub += r.auth
      norm += square(p.hub) // calculate the sum of the squared hub values to normalise
    norm = sqrt(norm)
    for each page p in G do  // then update all hub values
      p.hub = p.hub / norm   // normalise the hub values
```

The hub and authority values converge in the pseudocode above.

The code below does not converge, because it is necessary to limit the number of steps that the algorithm runs for. One way to get around this, however, would be to normalize the hub and authority values after each "step" by dividing each authority value by the square root of the sum of the squares of all authority values, and dividing each hub value by the square root of the sum of the squares of all hub values. This is what the pseudocode above does.

Non-converging pseudocode

```
G := set of pages
for each page p in G do
  p.auth = 1 // p.auth is the authority score of the page p
  p.hub = 1 // p.hub is the hub score of the page p
function HubsAndAuthorities(G)
  for step from 1 to k do // run the algorithm for k steps
    for each page p in G do  // update all authority values first
      p.auth = 0
      for each page q in p.incomingNeighbors do // p.incomingNeighbors is the set of pages that link to p
        p.auth += q.hub
    for each page p in G do  // then update all hub values
      p.hub = 0
      for each page r in p.outgoingNeighbors do // p.outgoingNeighbors is the set of pages that p links to
        p.hub += r.auth
```

References

1. Christopher D. Manning, Prabhakar Raghavan & Hinrich Schütze (2008). "Introduction to Information Retrieval" (http://nlp.stanford.edu/IR-book/html/htmledition/hubs-and-authorities-1.html). Cambridge University Press. Retrieved 2008-11-09.
2. Kleinberg, Jon (December 1999). "Hubs, Authorities, and Communities" (http://www.cs.brown.edu/memex/ACM_HypertextTestbed/papers/10.html). Cornell University. Retrieved 2008-11-09.
3. Pankaj Gupta, Ashish Goel, Jimmy Lin, Aneesh Sharma, Dong Wang, and Reza Bosagh Zadeh WTF: The who-to-follow system at Twitter (http://dl.acm.org/citation.cfm?id=2488433), Proceedings of the 22nd international conference on World Wide Web
4. von Ahn, Luis (2008-10-19). "Hubs and Authorities" (http://nitch.marketing/science-interwebs/) (PDF). *15-396: Science of the Web Course Notes*. Carnegie Mellon University. Retrieved 2015-01-19.

- Kleinberg, Jon (1999). "Authoritative sources in a hyperlinked environment" (http://www.cs.cornell.edu/home/kleinber/auth.pdf) (PDF). *Journal of the ACM* **46** (5): 604–632. doi:10.1145/324133.324140 (https://dx.doi.org/10.1145%2F324133.324140).
- Li, L.; Shang, Y.; Zhang, W. (2002). "Improvement of HITS-based Algorithms on Web Documents" (http://www2002.org/CDROM/refereed/643/). *Proceedings of the 11th International World Wide Web Conference (WWW 2002)*. Honolulu, HI. ISBN 1-880672-20-0.

External links

- U.S. Patent 6,112,202 (https://www.google.com/patents/US6112202)
- Create a data search engine from a relational database (http://www.dupuis.me/node/25) Search engine in C# based on HITS

Retrieved from "https://en.wikipedia.org/w/index.php?title=HITS_algorithm&oldid=667593744"

Categories: Link analysis