PageRank

Info 427

Announcements

• Remaining classwork:
  – Assignment 3, due Wednesday November 13
  – Assignment 4, due Friday November 22
  – Final project, due Sunday December 15
  – Final exam, Tuesday December 17

Final project

• Choose one of the following options:
  
  – **Paper option**: Research a topic related to web search (of your choice). Write a research paper (3000-5000 words ~15 pages). Individual work (not in pairs).

Vector Space Model

• Introduced by Salton in the 70’s (SMART system)

• Documents & queries represented as vectors in a high-dimensional space where each dimension corresponds to a term

• Sometimes referred to as "bag of words" approach
  – We do **not** worry about the positions of words

tf idf

• This scoring function is called “tf-idf”
  – “Term frequency – inverse document frequency”
  – Very commonly used!

\[
\text{score}(q,d) = \sum_{t \in q} \frac{\text{ntf}(t,d) \times \text{idf}(t)}{\log \left( \sum_{t \in d} \text{ntf}(t,d) \right)}
\]

\[
\text{ntf}(t,d) = \frac{f(t,d)}{\sum_{t \in d} f(t,d)}
\]

\[
\text{idf}(t) = \frac{1}{1 + \log (df(t))}
\]

Vector space model

• Represent a document as a histogram over word frequency

When in the Course of human events, it becomes necessary for one people to dissolve the political bands which have connected them with another, and to assume among the powers of the earth, the separate and equal station to which the Laws of Nature and of Nature’s God entitle them, a decent respect to the…
**Vectors like recipes**

- For each ingredient (term), specify amount (weight)

**Comparing two documents**

- We can now represent each document as a vector in a very high dimensional space
  - Let’s say we have a document D and a query Q. We can represent these as vectors d and q in the VSM.
  - Before we saw a simple scoring function:
    \[ \text{score}(Q, D) = \sum_{i \in Q} \text{tf}_i \cdot \text{df}_i \]
  - How would you write this in terms of vector notation?
    \[ \text{score}(q, d) = q \cdot d \]

**Common approach:** measure similarity using angle between two vectors (documents):

\[ \sigma(p_1, p_2) = \arccos \left( \frac{p_1 \cdot p_2}{\|p_1\| \cdot \|p_2\|} \right) \]

For speed, usually instead rank using cosine of angle between vectors:

\[ \sigma(p_1, p_2) = \frac{p_1 \cdot p_2}{\|p_1\| \cdot \|p_2\|} \]

**Q:** At any moment in time, what’s the probability that the frog is on pad 1?
Q: At any moment in time, what’s the probability that the frog is on pad 1?
A: \( P(\text{Pad}=1) = 1/5 \). Same for 2, 3, 4, 5.

Q: At any moment in time, what’s the probability that the frog is on pad 1?
A: \( P(\text{Pad}=1) = ? \)

\[ \begin{align*}
\text{Q: At any moment in time, what’s the probability that the frog is on pad 1?} \\
\text{A: } P(\text{Pad}=1) = ?
\end{align*} \]

• Let \( r_i \) denote \( P(\text{Pad}=i) \)
  - What is the value of \( r_1 \)?
    - If we knew \( r_4 \) and \( r_5 \), then it would be \( r_1 = r_5 + 0.5r_4 \)
    - \( r_2 = 0.5r_3 \)
    - \( r_3 = r_1 = r_2 = r_4 = r_5 \)
  - The sum of all \( r \)'s has to equal 1
  - So \( r_1 = ? \)

A little notation
• Recall that a directed graph has nodes (vertices) and edges (arcs or arrows)
  - The outdegree of a vertex is the number of edges leaving it
  - The indegree of a vertex is the number of edges entering it

Computing lily pad probabilities
• How do we compute the probability for a pad?
  - Sum up the probabilities from incoming edges

\[ r_j = \sum_{i \rightarrow j} r_i d_i \]

Sum over each of \( r \)'s incoming edges
Computing lily pad probabilities

• How do we compute the probability for all pads?
  – One approach: an iterative algorithm
    • Assign all nodes the same probability (e.g., 1/n)
    • Then repeat:
      Compute a new probability for each node, using the probabilities computed during the last iteration for the other nodes

\[ r_j = \sum_{i \neq j} \frac{r_i}{d_i} \]

More terminology

• The system of lily pads just described is an example of a Markov Chain, a very general math model
  – With states instead of pads, and transitions instead of leaps
  – Key property is that the system is memoryless, i.e. the next pad depends only on the current pad
  – The \( r \) values are called the stationary distribution, the probabilities of being on any pad at any moment in time

Relevance vs ranking

• Tf-idf scores pages based on relevance to a query
  – But there may be millions of documents relevant to a query:

• We need a way of ranking pages on the web according to their “quality” (or “importance”)
  – In the late 1990’s, it wasn’t clear at all how to do this

Now, back to web search...

Enter Google

• Larry Page and Sergey Brin, 1997
  – Commercialized a new way of ranking pages, based on the web’s structure

Link analysis

• Which pages look more “important”?
PageRank

- Basic idea: assign a score to every web page
  - The score represents importance or quality, independent of the query

- The structure of the web can help
  - High-quality pages are linked to by many other pages

Page quality score

- Let’s say page i has a quality score $r_i$
  - A link from i to a page is a “vote” for that page
  - If i is high-quality, its opinion should get more weight (be allowed to cast more votes)
  - So, let i have $r_i$ votes, to split equally among its outgoing links

Computing page quality score

- How do we compute the score for a page?
  - Sum up the votes cast by its incoming links
    $$r_j = \sum_{i|j \rightarrow i} \frac{r_i}{d_i}$$
    Sum over each of i’s incoming links

- How do we compute the score for all pages?
  - This is just like the Frog and Lily pad example!

Computing page quality score

- A simple model
  - A user (frog) is randomly clicking on links on webpages (lily pads)
  - At any given moment, there’s some probability that the random user is on page p
  - If p is important, many important pages will link to it, so the probability that the random user is at p will be high
  - If p is unimportant, few important pages will link to it, so the probability of being at the page is low

Page quality score – problems?

- Unfortunately, this doesn’t always work.
The PageRank model

- Problem: some nodes have in-degree 0, so their score is 0 and their votes do not count
  - Simple fix: add a small constant to every rank score

\[ r_j = \frac{p}{n} + (1-p) \sum_{i\in\text{adj}} \frac{r_i}{d_i} \]

Total number of pages on web

- Constant (e.g. 0.8)

What does the model mean?

- Say you’re a user on the web, visiting a page
  - with probability 1-p, follow a random link on the page;
  - otherwise (probability p), visit a random webpage
  - continue doing this forever

A page’s score is equal to the probability that the user is visiting that page at any moment in time

Computing PageRank

- The iterative algorithm from before can be used
  - Algorithm:
    - Assign an initial score to each node (e.g. 1/n, but doesn’t matter)
    - Then repeat:
      Compute a new score for each node, using the scores computed during the last iteration for the other nodes, and the equation:

\[ r_j = \frac{p}{n} + (1-p) \sum_{i\in\text{adj}} \frac{r_i}{d_i} \]

- Useful facts: This algorithm always converges, and
  - All of the scores are non-zero and in the range [0,1]
  - The scores sum to 1

An example

\[ r_j = \frac{p}{n} + (1-p) \sum_{i\in\text{adj}} \frac{r_i}{d_i} \]
An example

\[ r_j = \frac{p}{n} + (1 - p) \sum_{i \neq j} \frac{r_i}{d_i} \]

\[ \begin{align*}
1 & \quad 0.25 \\
2 & \quad 0.25 \\
3 & \quad 0.25 \\
4 & \quad 0.25
\end{align*} \]
An example

\[ r_j = \frac{p}{n} + (1 - p) \sum_{i \neq j} \frac{r_i}{d_i} \]

Diagram: 4 vertices labeled 1, 2, 3, 4. Edges connecting vertices with probabilities indicated. For example, edge between 1 and 2 has a probability of 0.25, and edge between 1 and 3 has a probability of 0.05.