SVD Applications: LSI and Link Analysis

Advanced Machine Learning Course

Shaghayegh Sahebi (Sherry)

Outline

- QR Factorization
- Latent Semantic Indexing (LSI)
- Kleinberg's Algorithm (HITS)
- PageRank Algorithm (Google)

Vector Space Model

Documents

D1:How to bake bread without recipes	
D2:The classic art of Viennese pastry	
D3:Numerical recipes: The art of scientific computing	
D4:Breads, pastries, pies and cakes: quantity baking recipes	
D5:Pastry: A book of best french recipes	

Terms

T1:bak(e,ing)	**T2:recipes**	**T3:bread**	**T4:cake**	**T5:pastr(y,ies)**	**T6:pie**

Vector Space Model

- Vector space model represents database as a vector space
  - In indexing terms space
  - Each document represented as a vector
    - weight of the vector: semantic importance of indexing term in the document

$$\underbrace{\begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}}_{\text{terms}}$$

  - queries are modeled as vectors

$$q^{(1)} = \underbrace{(1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0)^T}_{\text{terms}}$$

Vector Space Model

- Whole database: *d* documents described by *t* terms
  - *t x d* term-by-document matrix

Normalized with unit columns

$$A = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$\hat{A} = \begin{pmatrix} 0.5774 & 0 & 0 & 0.4082 & 0 \\ 0.5774 & 0 & 1 & 0.4082 & 0.7071 \\ 0.5774 & 0 & 0 & 0.4082 & 0 \\ 0 & 0 & 0 & 0.4082 & 0 \\ 0 & 1 & 0 & 0.4082 & 0.7071 \\ 0 & 0 & 0 & 0.4082 & 0 \end{pmatrix}$$

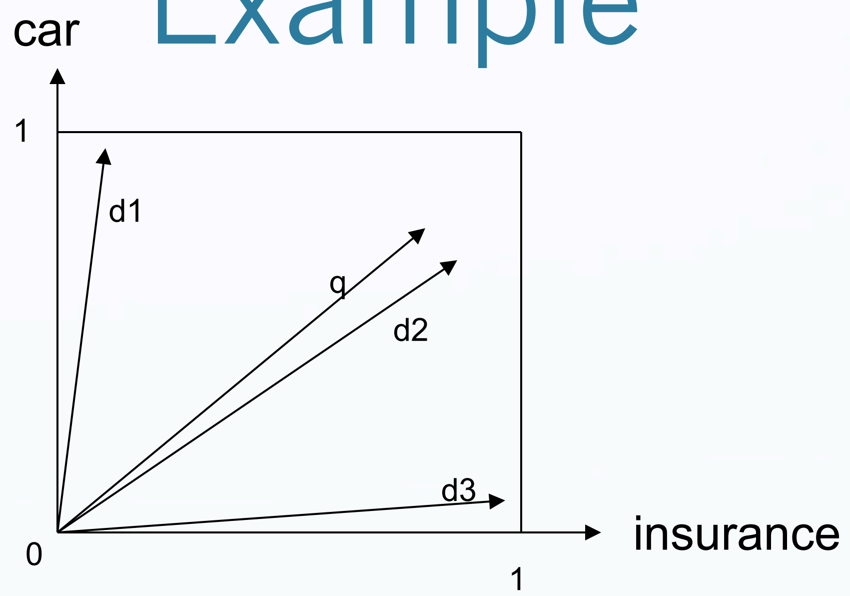
- the semantic content of the database is wholly contained in the column space of A

Similarity Measure

- How to identify relevant documents?
- Using spatial proximity for semantic proximity
  - Most relevant documents for a query ≈ those with vectors closest to the query
- **Cosine measure:** the most widespread similarity measure
  - the cosine of the angle between two vectors.
  - Unit vectors ➔ cosine measure = a simple dot product

$$\cos(\vec{x},\vec{y}) = \frac{\vec{x}.\vec{y}}{|\vec{x}||\vec{y}|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$$

Example



- A vector space with two dimensions
- Three documents and one query (unit vectors)
- D2 is the most similar document to query q

Term weighting

- Simplest term (vector component) weightings:
  - count of number of times word occurs in document
  - binary: word does or doesn't occur in document
- A document is a better match if a word occurs three times than once, but not a three times better match
  - ➔ a series of weighting functions e.g., $1+\log(x)$ *if* $x > 0$
- Significance of a term:
  - occurrence of a term in a document is more important if that term does not occur in many other documents
  - Solution: weight=global weight x local weight

QR-Factorization

- Some information are redundant in vector space model ➔ QR factorization
- How it works?
  - Identify a basis for the column space
  - Low rank approximation

Identify a Basis for Column Space

- For a rank r_A matrix A:
  - R: t x d upper triangular matrix
  - Q: t x t orthogonal matrix

$$A = QR$$

$$A = (Q_A Q_A^{\perp}) \begin{pmatrix} R_A \\ 0 \end{pmatrix} = Q_A R_A$$

$$\cos\theta_j = \frac{a_j^T q}{\| a_j \|_2 \| q \|_2} = \frac{(Q_A r_j)^T q}{\| Q_A r_j \|_2 \| q \|_2} = \frac{r_j^T (Q_A^T q)}{\| r_j \|_2 \| q \|_2}$$

Example

$$Q = \left(\begin{array}{cccc|cc} -0.5774 & 0 & -0.4082 & 0 & -0.7071 & 0 \\ -0.5774 & 0 & 0.8165 & 0 & 0.0000 & 0 \\ -0.5774 & 0 & -0.4082 & 0 & 0.7071 & 0 \\ 0 & 0 & 0 & -0.7071 & 0 & -0.7071 \\ 0 & -1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.7071 & 0 & 0.7071 \end{array}\right),$$

$$R = \left(\begin{array}{ccccc} -1.0001 & 0 & -0.5774 & -0.7070 & -0.4082 \\ 0 & -1.0000 & 0 & -0.4082 & -0.7071 \\ 0 & 0 & 0.8165 & 0 & 0.5774 \\ 0 & 0 & 0 & -0.5774 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right).$$

Query Matching

$$\begin{aligned} q = Iq = QQ^T \\ &= [Q_A Q_A^T + Q_A^{\perp}(Q_A^{\perp})^T]q \\ &= Q_A Q_A^T q + Q_A^{\perp}(Q_A^{\perp})^T q \\ &= q_A + q_A^{\perp}. \end{aligned}$$

$$\cos\theta_j = \frac{a_j^T q_A + a_j^T q_A^{\perp}}{\| a_j \|_2 \| q \|_2} = \frac{a_j^T q_A + a_j^T Q_A^{\perp}(Q_A^{\perp})^T q}{\| a_j \|_2 \| q \|_2}.$$

$$\cos\theta_j = \frac{a_j^T q_A + 0 \cdot (Q_A^{\perp})^T q}{\| a_j \|_2 \| q \|_2} = \frac{a_j^T q_A}{\| a_j \|_2 \| q \|_2}.$$

$$\cos\theta_j{}' = \frac{a_j^T q_A}{\| a_j \|_2 \| q_A \|_2}.$$

Low Rank Approximation

- Change in the DB or not being precise:
  - Use approximation of A: $A + E$ (Uncertainty Matrix)
- What if adding E reduces the rank of A
  - A can be partitioned to isolate smaller parts of the entries

$$R = \left(\begin{array}{rrr|rr} -1.0001 & 0 & -0.5774 & -0.7070 & -0.4082 \\ 0 & -1.0000 & 0 & -0.4082 & -0.7071 \\ 0 & 0 & 0.8165 & 0 & 0.5774 \\ \hline 0 & 0 & 0 & -0.5774 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right) = \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}.$$

QR- Factorization Problem

- Gives no information of row space
- Doesn't choose the smallest values ➔ Could be more precise
- Inability to address two problems
  - Synonymy: two different words (say car and automobile) have the same meaning
  - Polysemy: a term such as charge has multiple meanings
- Synonymy ➔ underestimate true similarity
- Polysemy ➔ overestimate true similarity
- Solution: LSI
  - Use the co-occurrences of terms to capture the latent semantic associations of terms?

Latent Semantic Indexing (LSI)

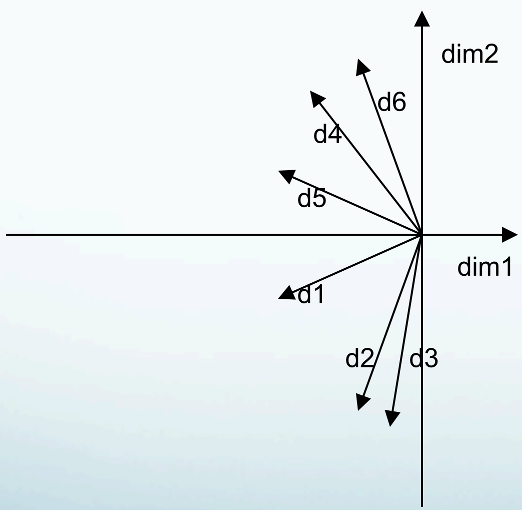
- Approach: Employing a low rank approximation to the vector space representation
- Goal: Cluster similar documents which may share no terms in the latent semantic space, which is a low-dimensional subspace. (improves recall)
- LSI projects queries and documents into a space with latent semantic dimensions.
  - co-occurring words are projected on the same dimensions
  - non-co-occurring words are projected onto different dimensions
- Thus, LSI can be described as a method for dimensionality reduction

Latent Semantic Indexing (LSI)

- Dimensions of the reduced semantic space correspond to the axes of greatest variation in the original space (closely related to PCA)
- LSI is accomplished by applying SVD to term-by-document matrix
- Steps:
  - Preprocessing: Compute optimal low-rank approximation (latent semantic space) to the original term-by-document matrix with help of SVD
  - Evaluation: Rank similarity of terms and docs to query in the latent semantic space via a usual similarity measure
- Optimality dictates that the projection into the latent semantic space should be changed as little as possible measured by the sum of the squares of differences

Example

$$A = \begin{array}{c} \\ \mathbf{cos}\,\boldsymbol{monaut} \\ \boldsymbol{astronaut} \\ \boldsymbol{moon} \\ \boldsymbol{car} \\ \boldsymbol{truck} \end{array} \begin{array}{cccccc} \boldsymbol{d1} & \boldsymbol{d2} & \boldsymbol{d3} & \boldsymbol{d4} & \boldsymbol{d5} & \boldsymbol{d6} \end{array} \atop \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$



- A: term-by- document matrix with rank 5
- Reduced to two dimensions (latent dimensions, concepts)
- In the original space the relation between d2 and d3 is not clear

Singular Value Decomposition (SVD)

- Decomposes A_{txd} into the product of three matrices T_{txd} , S_{nxn} and D_{dxn}

$$\begin{array}{cccccccc} A_{txd} & & T_{txn} & & S_{nxn} & & (D_{dxn})^T \\ \begin{pmatrix} o & o & o \\ o & o & o \\ o & o & o \\ o & o & o \end{pmatrix} & = & \begin{pmatrix} o & o \\ o & o \\ o & o \\ o & o \end{pmatrix} & x & \begin{pmatrix} o & o \\ o & o \end{pmatrix} & x & \begin{pmatrix} o & o & o \\ o & o & o \end{pmatrix} \end{array}$$

- T and D: have orthonormal columns
- S: diagonal matrix containing singular values of A in descending order. number of non-zero singular values = rank of A

Singular Value Decomposition (SVD)

- *Columns of T: orthogonal eigenvectors of AA^T*
- *Columns of D: orthogonal eigenvectors of A^TA*
- *LSI defines:*
  - *A as term-by-document matrix*
  - *T as term-to-concept similarity matrix*
  - *S as concept strengths*
  - *D as concept-to-doc similarity matrix*
- If rank of A is smaller than term count, we can directly project into a reduced dimensionality space. However, we may also want to reduce the dimensionality of A by setting small singular values of S to zero.

Dimensionality Reduction

- Compute SVD of $A_{txd}=T_{txn}S_{nxn}(D_{dxn})^T$
- Form $A\hat{}_{txk}=T_{txk}S_{kxk}(D_{kxn})^T$ by replacing the $r-k$ smallest singular values on the diagonal by zeros, which is the optimal reduced rank-k approximation of A_{txd}
- $B\hat{}_{txk}=S_{kxk}(D_{kxn})^T$ builds the projection of documents from the original space to the reduced rank-k approximation
  - in the original space, n dimensions correspond to terms
  - in the new reduced space, k dimensions correspond to concepts
- $Q_k=(T_{txk})^T Q_t$ builds the projection of the query from the original space to the reduced rank-k approximation
- Then we can rank similarity of documents to query in the reduced latent semantic space via a usual similarity measure

Example-SVD

$$A = \begin{array}{c} \\ \cos monaut \\ astronaut \\ moon \\ car \\ truck \end{array} \begin{pmatrix} d1 & d2 & d3 & d4 & d5 & d6 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

$$T = \begin{array}{c} \\ \cos monaut \\ astronaut \\ moon \\ car \\ truck \end{array} \begin{pmatrix} \dim 1 & \dim 2 & \dim 3 & \dim 4 & \dim 5 \\ -0.44 & -0.30 & 0.57 & 0.58 & 0.25 \\ -0.13 & -0.33 & -0.59 & 0.00 & 0.73 \\ -0.48 & -0.51 & -0.37 & 0.00 & -0.61 \\ -0.70 & 0.35 & 0.15 & -0.58 & 0.16 \\ -0.26 & 0.65 & -0.41 & 0.58 & -0.09 \end{pmatrix}$$

$$S = \begin{pmatrix} 2.16 & 0 & 0 & 0 & 0 \\ 0 & 1.59 & 0 & 0 & 0 \\ 0 & 0 & 1.28 & 0 & 0 \\ 0 & 0 & 0 & 1.00 & 0 \\ 0 & 0 & 0 & 0 & 0.39 \end{pmatrix}$$

$$D^T = \begin{array}{c} \\ \dim 1 \\ \dim 2 \\ \dim 3 \\ \dim 4 \\ \dim 5 \end{array} \begin{pmatrix} d1 & d2 & d3 & d4 & d5 & d6 \\ -0.75 & -0.28 & -0.20 & -0.45 & -0.33 & -0.12 \\ -0.29 & -0.53 & -0.19 & 0.63 & 0.22 & 0.41 \\ 0.28 & -0.75 & 0.45 & -0.20 & 0.12 & -0.33 \\ 0 & 0 & 0.58 & 0 & -0.58 & 0.58 \\ -0.53 & 0.29 & -0.63 & 0.19 & 0.41 & -0.22 \end{pmatrix}$$

Example-Reduction (rank-2 approx.)

$$S^r = \begin{pmatrix} 2.16 & 0 & 0 & 0 & 0 \\ 0 & 1.59 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

We can get rid of zero valued columns and rows
And have a 2 x 2 concept strength matrix

$$T^r = \begin{pmatrix} & \text{dim}1 & \text{dim}2 & \text{dim}3 & \text{dim}4 & \text{dim}5 \\ cosmonaut & -0.44 & -0.30 & 0 & 0 & 0 \\ astronaut & -0.13 & -0.33 & 0 & 0 & 0 \\ moon & -0.48 & -0.51 & 0 & 0 & 0 \\ car & -0.70 & 0.35 & 0 & 0 & 0 \\ truck & -0.26 & 0.65 & 0 & 0 & 0 \end{pmatrix}$$

We can get rid of zero valued columns
And have a 5 x 2 *term-to-concept similarity* matrix

$$D^{rT} = \begin{pmatrix} & d1 & d2 & d3 & d4 & d5 & d6 \\ \text{dim}1 & -0.75 & -0.28 & -0.20 & -0.44 & -0.33 & -0.12 \\ \text{dim}2 & -0.29 & -0.53 & -0.19 & 0.65 & 0.22 & 0.41 \\ \text{dim}3 & 0 & 0 & 0 & 0 & 0 & 0 \\ \text{dim}4 & 0 & 0 & 0 & 0 & 0 & 0 \\ \text{dim}5 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

We can get rid of zero valued columns
And have a 2 x 6 *concept-to-doc similarity matrix*

dim1 and dim2 are the new concepts

Example-Projection

Original space | Reduced latent semantic space

$$Q = \begin{pmatrix} \cos monaut & 1 \\ astronaut & 0 \\ moon & 0 \\ car & 0 \\ truck & 0 \end{pmatrix} \longrightarrow Q^r = \begin{pmatrix} \dim 1 & -0.44 \\ \dim 2 & -0.30 \end{pmatrix}$$

$$A = \begin{pmatrix} & d1 & d2 & d3 & d4 & d5 & d6 \\ \cos monaut & 1 & 0 & 1 & 0 & 0 & 0 \\ astronaut & 0 & 1 & 0 & 0 & 0 & 0 \\ moon & 1 & 1 & 0 & 0 & 0 & 0 \\ car & 1 & 0 & 0 & 1 & 1 & 0 \\ truck & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \longrightarrow B = \begin{pmatrix} & d1 & d2 & d3 & d4 & d5 & d6 \\ \dim 1 & -1.62 & -0.60 & -0.44 & -0.97 & -0.70 & -0.26 \\ \dim 2 & -0.46 & -0.84 & -0.30 & 1.00 & 0.35 & 0.65 \end{pmatrix}$$

$$\cos(Q, d2) = 0 \longrightarrow \cos(Q^r, d2) = 0.88$$

We see that query is not related to the d2 in the original space but in the latent semantic space they become highly related, which is true
Max [cos(x,y)]=1

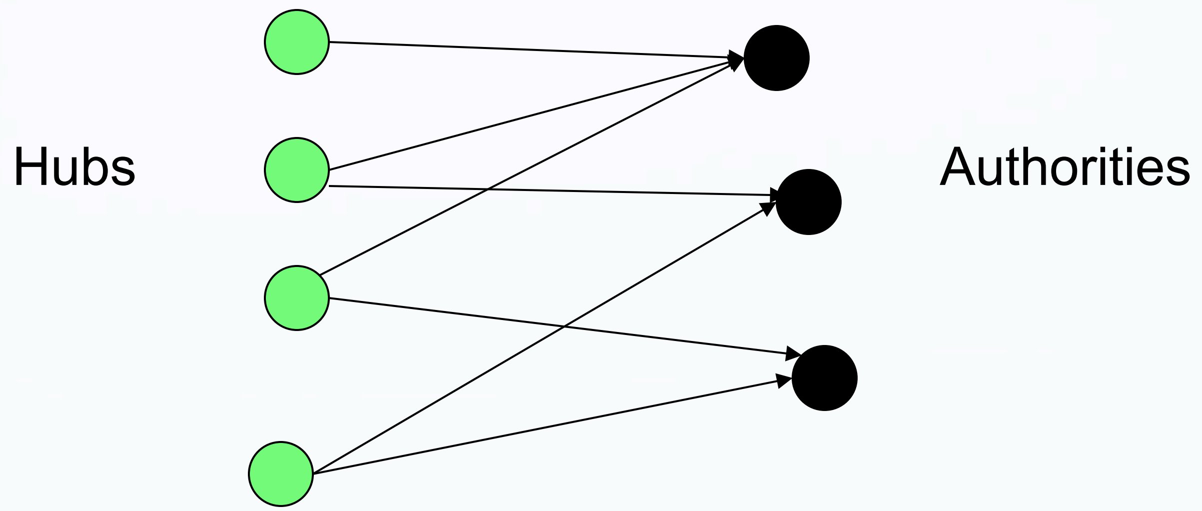
Database as in Graph Model

- Building citation graph and its adjacency matrix
- Represent documents and terms as nodes of the graph
- There is a link from each document to each term if the term appears in that document
- An authoritive word: a commonly used word
- connected components in the linkage graph: distinct document topics

Kleinberg's Algorithm

- Extracting information from link structures of a hyperlinked environment
- Basic essentials
  - Authorities
  - Hubs
- For a topic, authorities are relevant nodes which are referred by many hubs
- For a topic, hubs are nodes which connect many related authorities for that topic
- Authorities are defined in terms of hubs and hubs defined in terms of authorities
  - Mutually enforcing relationship (global nature)

Authorities and Hubs



- The algorithm can be applied to arbitrary hyperlinked environments
  - World Wide Web (nodes correspond to web pages with links)
  - Publications Database (nodes correspond to publications and links to co-citation relationship)

Kleinberg's Algorithm (WWW)

- Is different from clustering
  - Different meanings of query terms

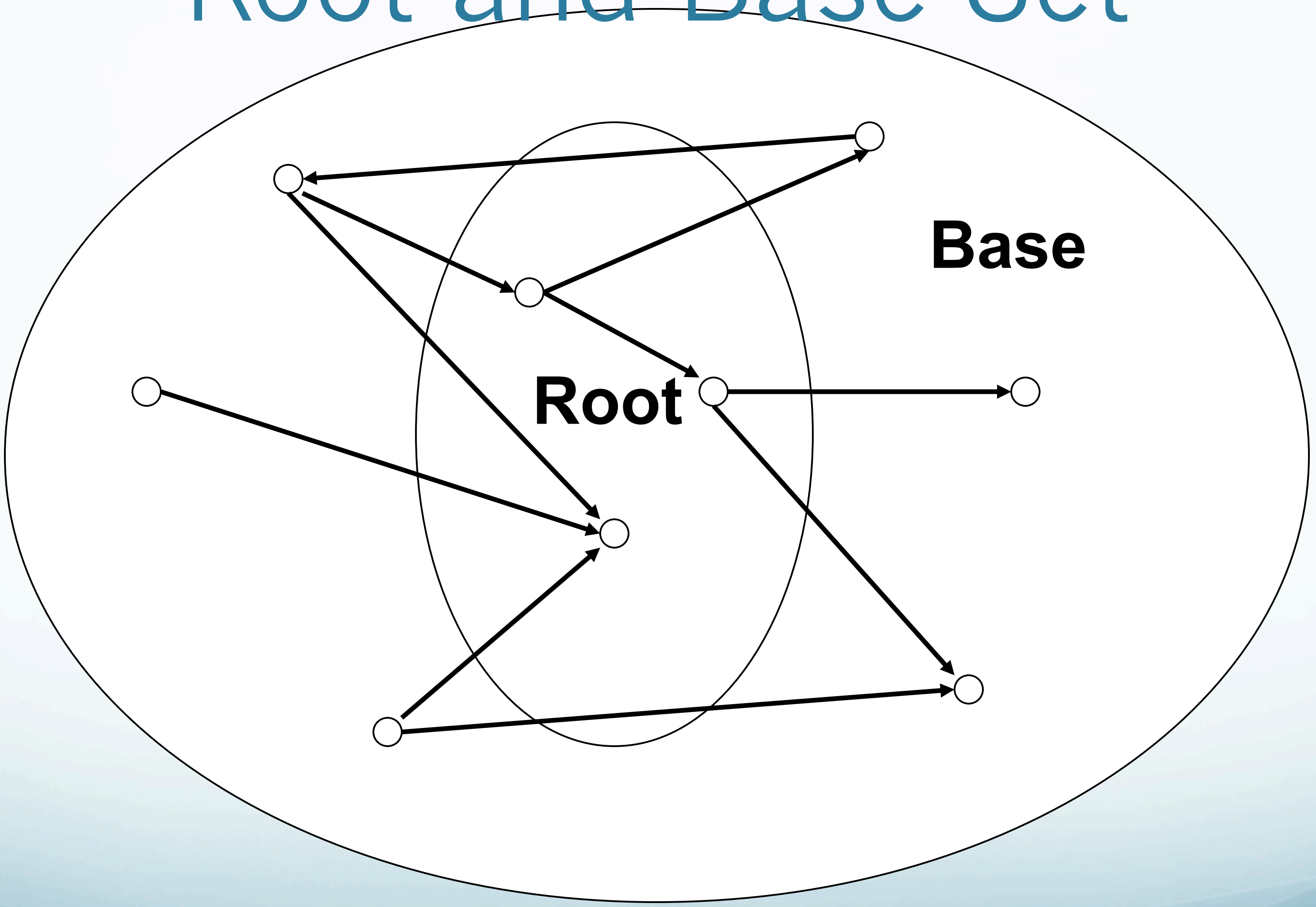
- Addressed problems by the text-based model
  - Self-description of page may not include appropriate keywords
  - Distinguish between general popularity and relevance

- Three steps
  - Create a focused sub-graph of the Web
  - Iteratively compute hub and authority scores
  - Filter out the top hubs and authorities

Root and Base Set

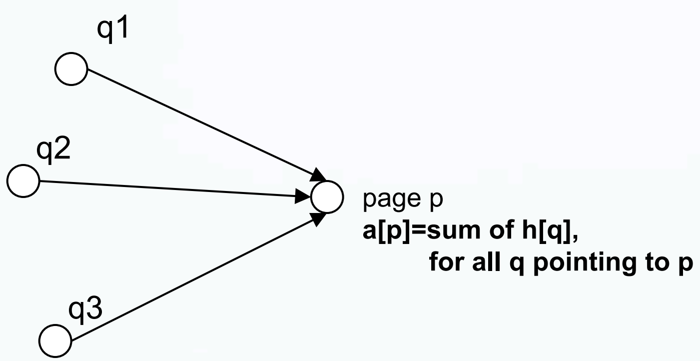
- For the success of the algorithm base set (sub-graph) should be
  - relatively small
  - rich in relevant pages
  - contains most of the strongest authorities
- Start first with a root set
  - obtained from a text-based search engine
  - does not satisfy third condition of a useful subgraph
- Solution: extending root set
  - add any page pointed by a page in the root set to it
  - add any page that points to a page in the root set to it (at most d)
  - the extended root set becomes our base set

Root and Base Set

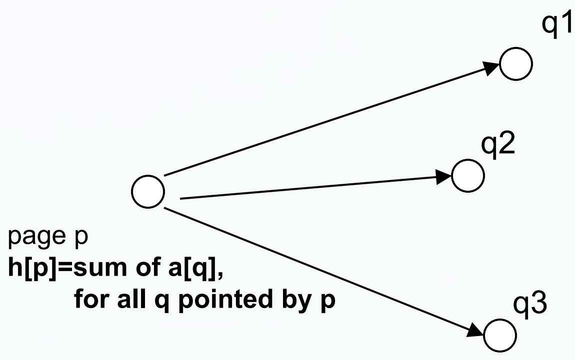


Two Operations

Updating authority weight

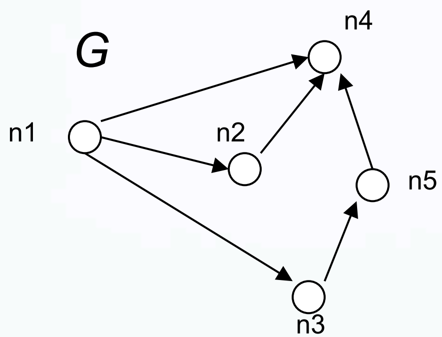


Updating hub weight



- a[p] ... authority weight for page p
- h[p] ... hub weight for page p
- Iterative algorithm
  1. set all weights for each page to 1
  2. apply both operations on each page from the base set and normalize authority and hub weights separately (sum of squares=1)
  3. repeat step 2 until weights converge

Matrix Notation



$$A = \begin{array}{c} \\ n1 \\ n2 \\ n3 \\ n4 \\ n5 \end{array} \begin{array}{c} \begin{array}{ccccc} n1 & n2 & n3 & n4 & n5 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

- G (root set) is a directed graph with web pages as nodes and their links
- G can be presented as a connectivity matrix A
  - A(i,j)=1 only if i-th page points to j-th page
- Authority weights can be represented as a unit vector a
  - a(i) is the authority weight of the i-th page
- Hub weights can be represented as a unit vector h
  - h(i) is the hub weight of the i-th page

Convergence

- Two mentioned basic operations can be written as matrix operations (all values are updated simultaneously)
  - Updating authority weights: $a = A^T h$
  - Updating hub weights: $h = Aa$
- After k iterations:

$$\begin{aligned} a_1 &= A^T h_0 \\ h_1 &= A a_1 \end{aligned} \longrightarrow h_1 = AA^T h_0 \rightarrow h_k = (AA^T)^k h_0$$

- Thus
  - h_k is a unit vector in the direction of $(AA^T)^k h_0$
  - a_k is a unit vector in the direction of $(A^T A)^{k-1} h_0$
- Theorem
  - a_k converges to the principal eigenvector of $A^T A$
  - h_k converges to the principal eigenvector of AA^T

Convergence

- $(A^TA)^k \times v` \approx (\text{const})\ v_1$ where k>>1, v` is a random vector, v_1 is the eigenvector of A^TA
- Proof:

$$(A^TA)^k = (A^TA) \times (A^TA) \times \ldots = (V\Lambda^2V^T) \times (V\Lambda^2V^t) \times \ldots$$

$$= (V\Lambda^2V^T) \times \ldots = (V\Lambda^4V^T) \times \ldots. = (V\Lambda^{2k}V^T)$$

Using spectral decomposition:

$$(A^TA)^k = (V\Lambda^{2k}V^T) = \lambda_1^{2k} v_1 v_1^T + \lambda_2^{2k} v_2 v_2^T + \ldots + \lambda_n^{2k} v_n v_n^T$$

because $\lambda_1 > \lambda_{i\neq1} \rightarrow \lambda_1^{2k} >> \lambda_{i\neq1}^{2k}$

thus $(A^TA)^k \approx \lambda_1^{2k} v_1 v_1^T$

now $(A^TA)^k \times v` = \lambda_1^{2k} v_1 v_1^T \times v` = (\text{const})\ v_1$

because $v_1^T \times v`$ is a scalar.

Sign of Eigenvector

- We know that $(A^TA)^k \approx \lambda_1^{2k} v_1 v_1^T$
- Since A is the adjacency matrix, elements of $(A^TA)^k$ are all positive
- $\rightarrow \lambda_1^{2k} v_1 v_1^T$ should be positive
- λ_1^{2k} is positive $\rightarrow$ $v_1 v_1^T$ is positive $\rightarrow$ all elements of v_1 should have the same sign (either all elements are positive or all are negative)

Sub-communities

- Authority vector converges to the principal eigenvector of A^TA, which lets us choose strong authorities
- Hub vector converges to the principal eigenvector of AA^T which lets us choose strong hubs
- These chosen authorities and hubs build a cluster in our network
- However there can exist different clusters of authorities and hubs for a given topic, which correspond to:
  - different meanings of a term (e.g. jaguar → animal,car,team)
  - different communities for a term (e.g. randomized algorithms)
  - polarized thoughts for a term (e.g. abortion)
- Extension:
  - each eigenvector of A^TA and AA^T represents distinct authority and hub vectors for a sub-community in Graph G, respectively.

PageRank

- PageRank is a link analysis algorithm that assigns weights to nodes of a hyperlinked environment
- It assigns importance scores to every node in the set which is similar to the authority scores in Kleinberg algorithm
- It is an iterative algorithm like Kleinberg algorithm
- Main assumptions:
  - in-degree of nodes are indicators of their importance
  - links from different nodes are not counted equally. They are normalized by the out-degree of its source.

Simplified PageRank (WWW)

$$\Pr(u) = \sum_{v \in B(u)} \frac{\Pr(v)}{L(v)}$$

B(u) is the set of nodes which have a link to u

- PageRank Algorithm simulates a random walk over web pages.
- Pr value is interpreted as probabilities
- In each iteration we update Pr values of each page simultaneously
- After several passes, Pr value converges to a probability distribution used to represent the probability that a person randomly clicking on links will arrive at any particular page

Matrix Notation

$$\underbrace{\mathbf{Pr}_{kx1} = M_{kxk} \times \mathbf{Pr}_{kx1}}_{\text{Update step}}$$

$$M_{ij} = \begin{cases} \dfrac{1}{|B_j|} & \text{, if } i \in B_j \\ 0 & \text{, else} \end{cases}$$

k is the number of total pages
B_i is the set of pages which have a link to i-th page

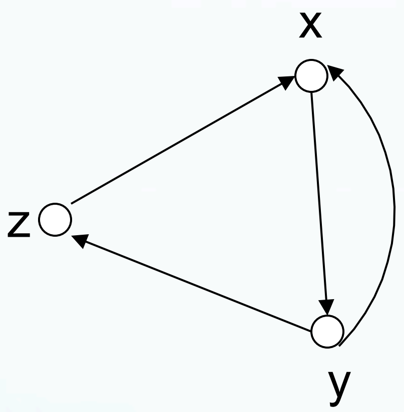
- M(i,j) is the transition matrix and defines fragment of the j-th page's Pr value which contributes to the Pr value of the i-th page

PageRank and Markov Chain

- PageRank defines a Markov Chain on the pages
  - with transition matrix M and stationary distribution Pr
    - states are pages
    - transitions are the links between pages (all equally probable)

- As a result of Markov theory, Pr value of a page is the probability of being at that page after lots of clicks.

Matrix Notation



Update step

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 & 0.5 & 1 \\ 1 & 0 & 0 \\ 0 & 0.5 & 0 \end{pmatrix} \times \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

$$x = 0 \cdot x + 1/2 \cdot y + 1 \cdot z$$

$$y = 1 \cdot x + 0 \cdot y + 0 \cdot z$$

$$z = 0 \cdot x + 1/2 \cdot y + 0 \cdot z$$

Non-Simplified PageRank (WWW)

$$\Pr(u) = \frac{1-d}{k} + d \cdot \sum_{v \in B(u)} \frac{\Pr(v)}{L(v)}$$

Matrix Notation

$$\overbrace{\Pr_{kx1} = M_{kxk} \times \Pr_{kx1}}^{\text{Matrix Notation}}$$

$$M_{ij} = \begin{cases} \dfrac{1-d}{k} + \dfrac{d}{\left|B_j\right|} & \text{, if } i \in B_j \\ \dfrac{1-d}{k} & \text{, else} \end{cases}$$

k is the number of total pages
B_i is the set of pages which have a link to i-th page

- (1-d) defines the probability to jump to a page, to which there is no link from the current page
- Pr converges to the principal eigenvector of the transition matrix M

Randomized HITS

- Random walk on HITS
- Odd time steps: update authority
- Even time steps: update hubs

$$\begin{aligned} a^{(t+1)} &= \epsilon\vec{1} + (1-\epsilon)A_{\text{row}}^{T} h^{(t)} \\ h^{(t+1)} &= \epsilon\vec{1} + (1-\epsilon)A_{\text{col}} a^{(t+1)} \end{aligned}$$

- t: a very large odd number, large enough that the random walk converged ➔ The authority weight of a page = the chance that the surfer visits that page on time step t

Stability of Algorithms

- Being stable to perturbations of the link structure.
- HITS: if the eigengap is big, insensitive to small perturbations; If it's small there may be a small perturbation that can dramatically change its results.
- PageRank: if the perturbed/modified web pages did not have high overall PageRank, then the perturbed PageRank scores will not be far from the original.
- Randomized HITS: insensitive to small perturbations

Thank You!

Special thanks to Cem