Scalable Topic Models

CS 572: Information Retrieval

Acknowledgements: Slides adapted from Gunnar Martinsson (NIPS 2009 tutorial) and Ankur Moitra (NMF)
Topic Modeling (Blei, etc.)

- Discover hidden topics
- Annotate documents according to these topics
- Organize and summarize the collection

*Parceling Out a Nest Egg, Without Emptying It*

By PAUL SULLIVAN

What clients often forget are fixed costs — homes, cars, insurance — that must come down but take time to reduce, she said. Beyond that is her clients’ skittish approach to risk; putting all of their money in cash may make them feel safe, she said, but it probably will not support the lifestyle they want for decades.

A generational disconnect is at work here: most people plan to retire at 65, the retirement age established for Social Security in 1935, when the average life expectancy was 61. Today the average is over 80 for men and women with a college degree.

So the $5.12 million gift exemption — created in a compromise between President Obama and Congress in 2010 — presents the well-off with a decision laden with short- and long-term consequences. How much should they give heirs now — and thus avoid giving the government in estate taxes later — while maintaining their lifestyle over a probably longer but still unpredictable remaining life span?
Term-by-Document Matrix

- words (m)
- documents (n)

relative frequency of word i in document j
Latent Semantic Indexing: Singular Value Decomposition

- Map terms and documents to a “latent semantic space”
- Similar terms map to similar locations in low dimensional space
- Noise reduction by dimensionality reduction
- Use singular value decomposition (SVD) to decompose term-by-document matrix

\[ A = U \Sigma V^T \]

- term vectors
- document vectors
SVD: A “Master” Algorithm

- Solve a linear system or any least squares problem
- Compute other factorizations: LU, QR, eigenvectors, etc.
- Data analysis: PCA, LSI, etc
- Standard algorithms are very stable, have only $O(n^3)$ asymptotic complexity and provide double precision accuracy

But what about large matrices, communication constraints (data stored in slow memory, streaming data), and also lots of noises?
Krylov Subspace Methods

- Method that often yields excellent accuracy for large, sparse problems
- “Everybody” knows Krylov subspace methods are great
- Basis for various algorithms to solve linear systems and eigenvalue problems

- Krylov subspace is a space generated by nonsingular matrix $A$

\[ K_k(A, r_0) = \text{Span}(r_0, Ar_0, \ldots, A^{k-1}r_0) \]
Power Method

- Simple algorithm that finds a single eigenvalue (one with greatest absolute value)

- Pick a random unit vector $x$

- Then repeat the following a few times: $x = Ax$ and normalize the vector

- Project all vectors to the subspace perpendicular to the estimated eigenvector and find the remaining ones
Lanczos Method

• Uses the power method except stores the series of vectors $A^{n-1}v$ and uses Gram-Schmidt process to reorthogonalize them

• Popular algorithm used to find top K eigenvectors and is implemented in many packages

• Iterative method to determine the next orthonormal vector and a tridiagonal matrix

• Eigenvalues of the tridiagonal matrix are approximate eigenvalues of original matrix $A$
Standard Deterministic Method: Golub-Businger Algorithm

- Determine an orthonormal basis \( \{ q_j \}_{j=1}^k \) for the range of \( A \) (simple Gram-Schmidt process can do this)
  
  Set \( Q = [q_1, q_2, \ldots, q_k] \)

- Form \( B = Q^T A \)

- Factor the small matrix \( B = \hat{U} \Sigma V^\top \)

- Calculate \( U = Q\hat{U} \)
Randomized SVD Algorithm

- Draw random vectors $w_1, w_2, \ldots, w_k$ from some distribution (e.g., Gaussian distribution)

- Compute samples $y_j = A w_j$ from Ran($A$). Vectors \{y_1, y_2, \ldots, y_k\} are linearly independent with probability 1

- Form an $n \times k$ matrix $Q$ whose columns form an orthonormal basis for the columns of the matrix $Y = [y_1, y_2, \ldots, y_k]$ => $A = QQ^T A$

- Form $k \times n$ “small” matrix $B = Q^T A$ => $A = QB$
Randomized Algorithm (2)

- Find the SVD of $B$ (cheap since $B$ is “small”):
  \[ B = \hat{U} \Sigma V^\top \]

- Calculate the original $U$:
  \[ U = Q\hat{U} \]

Similar to Golub-Businger with the difference in how we form the basis for range of $A$
Comparison to Golub-Businger

- Golub-Businger restricts $A$ to a basis that assuredly spans the range of the matrix.

- Randomized algorithm restrict $A$ to the range of the random sample matrix.

- Appeal of randomized method is that $Y$ can be evaluated with a single sweep and is amendable to BLAS3, etc.

- Some issues arise in theory and practice when performing algorithms with finite-precision arithmetic.
Power Method for Improving Accuracy

- Idea: Apply randomized algorithm to auxiliary matrix
  \[ B = (AA^\top)^q A \]

- Matrices A and B have same left singular vectors and the singular values of B decay much faster => better algorithmic performance
  \[ \sigma_j(B) = (\sigma_j(A))^{2q+1} \]

- Use sample matrix instead of the original Y
  \[ B\Omega = (AA^\top)^q A\Omega \]
Updated Randomized Algorithm

• Pick a parameter $p$ and set $l = k + p$

• Draw an $n \times l$ random matrix $\Omega$

• Compute the sample matrix $Y = (AA^\top)^q A\Omega$

• Compute an orthonormal matrix $Q$, such that $Y = QQ^\top Y$

• Form $B = Q^\top A$

• Factorize $B = \hat{U}\Sigma V^\top$

• Form $U = Q\hat{U}$

• If desired, truncate the $l$-term SVD to its leading $k$ terms
Example: 9025 x 9025 matrix

- Pink line: performance of random sampling is quite large
- Power method helps quite a bit towards recovering the exact eigenvalues
Example: Synthesized Data

- A 500,000 x 80,000 matrix (160 GB of storage) was synthesized with singular values having some specific form.

- Algorithm was implemented with q=3 in Matlab.

- Computer was a single-core 32-bit 2GHz Pentium M laptop with 1.5GB of Ram.

- External hard drive connected via USB 2.0.

- Matrix was processed in 18 hours with estimated accuracy of 0.01 ± 0.001.
LDA: Overview

- Each document is a distribution on topics
- Each topic is a distribution of words
LDA: Inference

- Learning the various distributions (set of topics, associated word probabilities, topic of each word, and particular topic mixture of each document) is a problem of Bayesian inference.

- Original paper [Blei et al, 2003] used variational Bayes approximation (approximate posterior distribution to get around intractable integrals).
  
  - Gibbs sampling [Pritchard et al, 2000; Griffith & Steevers, 2004]
  
  - Expectation propagation [Minka & Lafferty, 2002]

Are there efficient algorithms to find topics?
Matrix Factorization

• Low rank approximation to original matrix

• Generalization of many methods (e.g., SVD, QR, CUR, Truncated SVD, etc.)

• Basic Idea: Find two (or more) matrices whose product best approximate the original matrix

\[ X \approx \underbrace{W \ H^\top}_M \underbrace{N \times R \times R}_R, \quad R << N \]
Matrix Factorization (Pictorially)

Data matrix $X$ is approximated as $X \approx WH$.

- $X$: Data matrix
- $W$: "regressors", "activation coefficients", "expansion coefficients"
- $H$: "dictionary", "patterns", "topics", "basis", "explanatory variables"
Nonnegative Matrix Factorization (NMF)

• Popularized by Lee and Seung (1999) for “learning the parts of objects”

• Both $W$ and $H$ are nonnegative

• Empirically induces sparsity

• Improved interpretability (sum of parts representation)

• Applications to text classification, information retrieval, collaborative filtering, etc.
NMF (Example)

WLOG, assume columns of \( W \) and \( H \) sum to 1
NMF (Example)

WLOG, assume columns of $W$ and $H$ sum to 1

<table>
<thead>
<tr>
<th>topic</th>
<th>loading</th>
</tr>
</thead>
<tbody>
<tr>
<td>personal finance</td>
<td>0.55</td>
</tr>
<tr>
<td>bond-fund</td>
<td>0.30</td>
</tr>
<tr>
<td>advice</td>
<td>0.15</td>
</tr>
</tbody>
</table>

article from wall street journal
Algorithms for NMF

Local Search: Given $W$, compute $H$, compute $W$, …

- Known to fail on worst-case inputs (stuck in local optima)

- Highly sensitive to:
  - Cost function
  - Update procedure
  - Regularization

Can there be an efficient algorithm that works on all inputs?
NMF: Worst-case Complexity

- Theorem [Vavasis 2009]: It is NP-hard to compute NMF

- Theorem [Cohen & Rothblum 1993]: Can solve NMF in time $(nm)^{O(nr+mr)}$

- Theorem [Arora, Ge, Kanna, Moitra 2012]: Can solve NMF in time $(nm)^{O(r^2)}$ yet any algorithm that runs in time $(nm)^{o(r)}$ would yield a $2^{o(n)}$ algorithm for 3-SAT

$$X \approx WH$$

system of polynomial inequalities

variables
NMF: Separability and Anchor Words

If an anchor word occurs then the document is at least partially about the topic.
NMF: Separability and Anchor Words

If an anchor word occurs then the document is at least partially about the topic.
NMF: Separability and Anchor Words

If an anchor word occurs then the document is at least partially about the topic.

W is p-separable if each topic has an anchor word that occurs with probability at least p.
NMF: Complexity for Separability

- Theorem [Arora, Ge, Kanna, Moitra 2012]: There is an $O(nmr + mr^{3.5})$ time algorithm for NMF when the topic matrix $W$ is separable.

  Topic Models: documents are stochastically generated as convex combination of topics.

- Theorem [Arora, Ge, Moitra 2012]: There is a polynomial time algorithm that learns the parameters of any topic model provided that the topic matrix $W$ is $p$-separable.

  Algorithm is highly practice and runs orders of magnitude faster with nearly-identical performance as the current best (Gibbs sampling).
How Anchor Words Help

Observation: If $W$ is separable, the rows of $H$ appear as rows of $M$, and we just need to find the anchor words

How to find anchor words?
Think of Anchor Words as Vertices

Deleting a word changes the convex hull

Anchor word!
NMF Algorithm

- Find anchor words: can be found by linear programming or a combinatorial distance-based algorithm
- Paste these vectors in as rows in $H$
- Find nonnegative $W$ so that $WH = X$ (convex programming)
Topic Models

$X \approx WH$
Latent Dirichlet Allocation (Blei, Ng, Jordan)

- assumes independence between topics
- representation drawn from Dirichlet distribution
Topic Models

Correlated Topic Model (Blei, Lafferty)

- allows correlation between topics
- covariance matrix of logistic normal models
  topic correlations
Topic Models

Pachinko Allocation Model (Li, McCallum)

- models correlations between topics by uncovering hidden thematic structure
- extension of LDA

Models differ only in how $H$ is generated!
Algorithms for Topic Models: Gram Matrix

\[ E[XX^\top] = WE[HH^\top]W^\top \rightarrow WRW^\top \]

Gram Matrix

\[ \hat{X} \hat{X}^\top \]

Anchor words are extreme rows of Gram matrix
Bayes Rule: Using Anchor Words

points are rows of Gram matrix $W$

$$
Pr[\text{word} \mid \text{topic}] = \frac{Pr[\text{topic} \mid \text{word}] \ Pr[\text{word}]}{\sum_{\text{word}'} Pr[\text{topic} \mid \text{word}'] \ Pr[\text{word}']}
$$
Topic Model Algorithm

• Form Gram matrix and find anchor words

• Write each word as a convex combination of the anchor words to find $P[\text{topic} \mid \text{word}]$

• Compute $W$ from the formula above

• This provably works on any topic model provided $W$ is separable and $R$ is non-singular
Experiment: Synthetic NIPS documents

![Graph 1: Time vs. Documents](image1)

- Algorithm options: Gibbs, Recover, RecoverL2, RecoverKL

![Graph 2: L1 error vs. Documents](image2)

- Algorithm options: Gibbs, Recover, RecoverL2, RecoverKL
Experiment: UCI Collection of NYT

• 300,000 New York Times articles with 30,000 distinct words

• Run time: 12 minutes (compared to 10 hours for MALLET and other state-of-the-art topic models)

• Topics are high quality

<table>
<thead>
<tr>
<th>RecoverL2</th>
<th>run inning game hit season zzz_anaheim_angel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gibbs</td>
<td>run inning hit game ball pitch</td>
</tr>
<tr>
<td>RecoverL2</td>
<td>father family zzz_elian boy court zzz_miami</td>
</tr>
<tr>
<td>Gibbs</td>
<td>zzz_cuba zzz_miami cuban zzz_elian boy protest</td>
</tr>
<tr>
<td>RecoverL2</td>
<td>file sport read internet email zzz_los_angeles</td>
</tr>
<tr>
<td>Gibbs</td>
<td>web site com www mail zzz_internet</td>
</tr>
</tbody>
</table>
Only Tip of the Iceberg

- Distributed stochastic gradient descent with MapReduce
- Distributed stochastic gradient descent with Spark
- Divide-and-conquer approach for solving SVD based on MapReduce
- Divide-and-conquer for matrix factorization
- Coordinate descent for matrix factorization
- ...

42