Welcome
2017 Faculty Summit
Attendees

Faculty Summit 2017
microsoftfacultysummit.com

Microsoft Research
Microsoft.com/research

Facebook
@microsoftresearch

Twitter
@MSFTResearch
#FacSumm
#EdgeofAI
Topic Modeling: Proof to Practice

MSR India . . . . . . . .
Bing Ads . . . . . . . .
Indian Institute of Science . . . . .

Ravi Kannan, Harsha Simhadri
Kushal Dave, Shrutendra Harsola
Chiranjib Bhattacharyya
Non-Negative Matrix Factorization

\[
\begin{pmatrix}
A \\
\geq 0
\end{pmatrix}
\begin{pmatrix}
B \\
\geq 0
\end{pmatrix}
\begin{pmatrix}
C \\
\geq 0
\end{pmatrix}
\]

Input

Output
Topic Modeling

- $n \ (10^6 \ +)$ documents. Doc is a $d(5K+) \ vector$ of word frequencies.
  - Assume there are $k \ (100s) \ unknown \ topics$ (topic is also a $d$-vector) so that each doc is \textit{approximately} a convex combination of topics.

- Find Topics. Too hard in general. Assume Generative Model.

- Latent Dirichlet Allocation (LDA) Model [Blei, Ng, Jordan]. For each doc:
  - (Randomly) Generate $k$-vector of topic weights; take weighted combination of topics as word-probabilities-vector.
  - Generate words independently with these probabilities

- Nice theory. Many applications. \textit{But..}
**INPUT: Term-Doc Matrix**

<table>
<thead>
<tr>
<th>Term</th>
<th>Doc 1</th>
<th>Doc 2</th>
<th>...</th>
<th>...</th>
<th>...</th>
<th>...</th>
<th>Doc. 10^6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Election</td>
<td>.09</td>
<td>...</td>
<td>H</td>
<td>H</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Putin</td>
<td>.11</td>
<td>...</td>
<td>H</td>
<td>H</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Debate</td>
<td>.07</td>
<td>...</td>
<td>H</td>
<td>H</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Batter</td>
<td>.18</td>
<td>...</td>
<td>H</td>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Match</td>
<td>.2</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>Rain</td>
<td>.16</td>
<td>...</td>
<td></td>
<td>H</td>
<td>H</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Cloudy</td>
<td>.09</td>
<td>...</td>
<td></td>
<td>H</td>
<td>H</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Happen</td>
<td>.03</td>
<td>...</td>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>while</td>
<td>.03</td>
<td>...</td>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Sunday</td>
<td>.04</td>
<td>...</td>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
<td>...</td>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**OUTPUT: Topics Matrix**

<table>
<thead>
<tr>
<th>Topic1</th>
<th>Topic2</th>
<th>Topic3</th>
</tr>
</thead>
<tbody>
<tr>
<td>.38</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>.28</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>.27</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>.38</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>.6</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>.68</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>.3</td>
</tr>
<tr>
<td>.03</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>.01</td>
<td>.02</td>
<td>0</td>
</tr>
<tr>
<td>.02</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>.01</td>
<td>0</td>
<td>.02</td>
</tr>
</tbody>
</table>

**Topic 1:** Identified with "Election", "Putin" and "Debate". We can call it "Politics" (ex post facto).

**Topic 3:** "Weather" or "Seattle".

**Doc 1** ≈ 0.3 (Topic 1) + 0.4 (Topic 2) + 0.3 (Topic 3) = (0.114, 0.084, 0.081, ...)

**Latent Stochastic Model:** Has Topic Matrix.
Generates topic weights (.3,.4,.3).
Generates say 100 words as per (0.114, 0.084, 0.081, ...).
Out: Only freq of words generated.
ML and Theory

- Machine Learning has long (since 1990’s) studied these problems.
- Theory: developed provable algorithms. Here- ``best of both’’:
  - Empirically verifiable assumptions.
  - Good provable time and error bounds.
  - Scales up on real data (to 100M docs, 6B tokens, 1000 topics on a single box)
- Provable:
  - If data was generated (by unknown model), alg should provably (approx) find the generating model (Reconstruction).
  - Prove (good) poly time bounds (for every model satisfying assumptions)
Provable Algorithms for Topic Models, NMF

- **Arora, Ge, Kannan, Moitra**: Provable alg for NMF with assumptions. (STOC 2012).
- **Arora, Ge, Moitra**: Ditto for Topic Modeling. Assumptions. (ICML 2013).
- **Anandakumar, Foster, Hsu, Kakade**: Tensor-based methods (2012)
- **Bansal, Bhattacharyya, Kannan**: Provable Topic Modeling Algorithm under realistic, empirically verified assumptions. (NIPS 2014).
- Proof of the pudding: Scalable. Billions of tokens.
- Metrics for Real Data (generating model not available).
Geometry of Topic Modeling: Basic Topics in space spanned by vocabulary

Given \textit{docs} (o’s), find \( M \)

Helps to find nearly \textit{pure documents} (o’s near corners).
Dependent Topics

• Existing Models- assume the topic vectors are linearly (``very’’) independent.

• If true, we want to scale upto 5K topics, (essential) rank of data matrix must be 5K. Lets Check! Rank of Data: Observable.

• Real data has rank <<5K! How then can be scale up number of topics?

• Must have linearly dependent topics.

• Our Model: Small number of lin indep basic topics.
  • Each (actual) topic is a convex combination of two basic topics.
Squared singular values of data matrix

NY Times

PubMed
Geometry of Topic Modeling: Edge Topics

3 basic topics (corners of triangle) and 6 edge topics.

If there aren’t too many points near corners, edge topics provide tighter explanation of data.
Our Model – Assumption I

- Each topic has a set of *Catchwords*.
  - Each Catchword has higher frequency in its topic than in other topics.
  - All catchwords together have frequency at least 0.1.

- Replaces *Anchor words* assumption of earlier provable algorithms.

- Anchor Words Vs Catchwords:
  - *Homerun* occurs only in topic baseball; every 10\textsuperscript{th} word in baseball is *Homerun*.
  - Freq of each of *Batter, Bases, homerun* in baseball is 1.1 times frequency in any other topic and every 10\textsuperscript{th} word is one of these.
Our Model – Assumptions II and III

- Each doc has a dominant topic whose weight is $\geq$ say 0.2, (when $k = 100$) and the weight of each other topic in document is $\leq 0.15$.

- Nearly pure docs: For each topic, there is a fraction (say $1/10k$) of documents whose weight on the topic is $\geq 0.9$.

- All three assumptions are
  - Empirically verified
  - Provably hold if we assume LDA model.
  - Like previous models, we also need some technical assumptions.

- Assumption 0: Documents are independent random.
SVD gets a bad rap

• **Latent Semantic Indexing** *(Susan Dumais – 1990-state of the art): Pre-Topic Modeling (uses SVD)*

• **Papadimitriou, Vempala**: LSI provably does Topic Modeling when each doc has a single topic (1998).

• **Folklore**: *SVD does not help when multiple topics per doc.*


• **Arora, Ge, Moitra “Beyond SVD...”** - Optimization based algorithm for topic models.
Importance Sampled Learning for Edge Topics

Three simple steps for basic topics:
- Threshold
- SVD and Cluster by dominant topic
- Find Catchwords and Topics.

+ One step for edge topics

Why ISLE?
- Provable bounds on reconstruction error and time.
- Performs better on many quality metrics.
- Fast in practice. Highly parallelizable.
- Can scale to terabytes of data via $l^2$-sampling: Can run on sampled data with two passes over disk resident data.
Step I: Thresholding to the rescue

Diagonal blue blocks are Catchwords for each topic. 
Black: Non-Catchwords.

We develop an algorithm to find the right threshold for each word and prove its efficacy.
Step 2: SVD and Cluster

- Idea: Use k-means clustering to the thresholded data to get dominant topics.
- But, no proof of convergence in general
- Leveraging earlier theory [Kumar, Kannan]
  - We prove clustering in SVD projection gives a good start, and convergence.

- Can reduce SVD and cluster compute time:
- $L^2_2$ sampling of thresholded documents
What careful clustering gave us

Given *documents* (o’s), find $\mu_l$.

Helps to find nearly *pure documents* (o’s near corners).

Cannot take average of points in each cluster. Need Corners.

$\mu_1 = (0.38, 0.28, 0.27, 0, 0, 0, 0.03, 0.01, 0.02, 0.01)$

$\mu_2 = (0, 0, 0, 0.38, 0.6, 0, 0, 0, 0.02, 0, 0)$

$\mathbf{x}$ : Probability Vector

= Weighted combination of $\mu_l$

$\mathbf{o}$ : Observations, i.e., documents
Step 3: Using Clustering to find basic topics

- Find *catchwords* for each topic
  - Words with $\left(1 - \frac{1}{k}\right)$ fractile freq in this cluster $> 1.1$ times other clusters.
  - We prove that these are (approx) the set of catchwords.

- Find *pure documents* for each topic:
  - Find $\left(1 - \frac{1}{10k}\right)$ fractile of total freq of catchwords. Take docs above this freq.
  - We prove this is the set (approximately) of pure documents for the topic.

- Return the *average of pure docs* found.
  - We prove the errors incurred are small enough.
Step 4: Edge Topics from Basic Topics

• For each document $s$, find $l_1(s)$ and $l_2(s)$, the basic topics whose catchwords have first and second highest count in $s$.

• $X(l, l') = \{ s \mid l = l_1(s), l' = l_2(s), \text{wt of } l' \geq***\}$

• If $|X(l, l')| \geq\star\star$, average of doc’s in $X(l, l')$ is an edge topic.
Empirical Results: Quality

- **Metrics**
  - **Topic Coherence**: $\log$ (pair-wise co-occurrence) of the top 5 words in each topic.
  
  - **Likelihood**: Given $A$, find MLE $M$ and likelihood under $M$.
  
  - **Reconstruction Error**: Does the topic model constructed from finite number of samples converge to a perceived ground truth.

- **Comparison with LightLDA** [Yuan, Gao, Lui, Ma (MSR), Ho, Dai, Wei, Zheng (CMU)]:
  - Compare results on NY Times, Wikipedia and *their biggest open dataset* – Pubmed.
  - Compared with 100 iterations of TLC’s LightLDA.

**ISLE is mostly better than LightLDA**
# Datasets and Sizes

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Docs</th>
<th>Words</th>
<th>NNZs</th>
<th>Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>NY Times</td>
<td>270K</td>
<td>101K</td>
<td>57M</td>
<td>81M</td>
</tr>
<tr>
<td>PubMed</td>
<td>8.1M</td>
<td>141K</td>
<td>429M</td>
<td>650M</td>
</tr>
<tr>
<td>Wikipedia</td>
<td>11.7M</td>
<td>200K</td>
<td>981M</td>
<td>2168M</td>
</tr>
<tr>
<td>ProductAds</td>
<td>100M</td>
<td>300K</td>
<td>4B</td>
<td>6B</td>
</tr>
</tbody>
</table>
ISLE vs LightLDA: Topic Coherence and Likelihood

<table>
<thead>
<tr>
<th>Topics</th>
<th>Average log-likelihood per document</th>
<th>Average topic coherence based on top-5 words</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ISLE</td>
<td>LDA</td>
</tr>
<tr>
<td>100</td>
<td>-1220</td>
<td>-2891</td>
</tr>
<tr>
<td>50K</td>
<td>-2436</td>
<td>-4075</td>
</tr>
</tbody>
</table>

For $k_0 = 100; 1000; 2000$ basic topics and $k = 10K; 20K; 50K$ edge topics. Edge topics were generated using 2000 basic topics. **Bold** is better.
Importance Sampling

<table>
<thead>
<tr>
<th></th>
<th>Average Topic Coherence</th>
<th></th>
<th>Log-likelihood</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>((k_0))</td>
<td>100% 10%</td>
<td>100% 10%</td>
<td>100% 10%</td>
<td>100% 10%</td>
</tr>
<tr>
<td>100</td>
<td>-25.32 -24.94</td>
<td>-15.91 -16.95</td>
<td>-422 -431</td>
<td>-815 -790</td>
</tr>
<tr>
<td>1000</td>
<td>-27.70 -32.70</td>
<td>-19.28 -21.54</td>
<td>-481 -479</td>
<td>-671 -691</td>
</tr>
</tbody>
</table>

Average topic coherence and log-likelihood of \(k_0\) basic topics for ISLE with \(r = s/10\) (10\%) sampling and \(r = s\) (100\%). Sampling does not significantly affect quality.
Reconstruction Error: PubMed

- Take 2 samples from corpus and compare best-matched $L_1$ distance between the two topic matrices returned by the algorithm.
- This obviates knowing ``ground truth.'' Note: we are not given generating matrix.
- Closer is better. $L_1$ distance range: [0, 2].

<table>
<thead>
<tr>
<th>L1 Distance</th>
<th>ISLE</th>
<th>LightLDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>100K samples</td>
<td>0.63 Avg</td>
<td>1.39 Avg</td>
</tr>
<tr>
<td></td>
<td>1.16 Max</td>
<td>1.98 Max</td>
</tr>
<tr>
<td>200K samples</td>
<td>0.48 Avg</td>
<td>1.31 Avg</td>
</tr>
<tr>
<td></td>
<td>0.99 Max</td>
<td>1.90 Max</td>
</tr>
</tbody>
</table>
Empirical Results: Time

• 16 core workstation, dual Intel® Xeon® E5-2630 v3128 CPUs, 128GB RAM

• ISLE
  • VC++/VS2015, OpenMP for multi-core parallelism.
  • Intel® Math Kernel Library 17.x.y for parallel sparse and dense math calls.
  • SVD using Spectra Eigenvalue solver library, a C header reimplementation of ARPACK with Eigen Matrix Classes and Intel MKL.
  • Distance calculations translated to BLAS calls, linked to MKL.
    e.g. For points P and centers C (matrix columns are coordinates):

\[ L_2^2(P, C) = 1^T \text{colsum}^2(P) + \text{colsum}^2(C)1^T - 2PC \]

• LightLDA: MAML v3.6, 24 threads, 100 iterations
### PubMed Time

<table>
<thead>
<tr>
<th>Time on 16 core machine</th>
<th>ISLE (r=s/10, 10%)</th>
<th>ISLE (r=s, 100%)</th>
<th>LightLDA (100 iter)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 Basic topics</td>
<td>6 min</td>
<td>17 min</td>
<td>74 min</td>
</tr>
<tr>
<td>1000 Basic topics</td>
<td>51 min</td>
<td>118 min</td>
<td>118 min</td>
</tr>
<tr>
<td>2000 Basic topics</td>
<td>147 min</td>
<td>501 min</td>
<td>123 min</td>
</tr>
<tr>
<td>100000 Edge topics</td>
<td>149 min</td>
<td>503 min</td>
<td>237 min</td>
</tr>
</tbody>
</table>

Computing Edge Topics from Basic Topics takes at most 2 minutes. LightLDA in TLC 3.6 is about 2-3 times faster than in TLC 3.2 and TLC 2.8!
Wikipedia Time

| Time on 16 core machine       | ISLE  
\(r=s/10, 10\%\) | ISLE  
\(r=s, 100\%\) | LightLDA  
(100 iter) |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>100 Basic topics</td>
<td>13 min</td>
<td>30 min</td>
<td>142 min</td>
</tr>
<tr>
<td>1000 Basic topics</td>
<td>84 min</td>
<td>233 min</td>
<td>193 min</td>
</tr>
<tr>
<td>2000 Basic topics</td>
<td>172 min</td>
<td>757 min</td>
<td>209 min</td>
</tr>
<tr>
<td>100000 Edge topics (from 2000 Basic Topics)</td>
<td>174 min</td>
<td>759 min</td>
<td>411 min</td>
</tr>
</tbody>
</table>

Our running time for 100M ProductAds was 8.5 hours for 1000 topics and the coherence was -25.3
Potential improvements in time

• SVD solver → Improvements in Spectra library
• K-means++ for initialization → Explore faster seeding algorithms
• K-means, currently Lloyds → Elkans, YingYang (TLC) algorithms

• Goal: Multi-node implementation, work with 1TB scale data.
Conclusion

• New topic models with large sample sets, vocabularies, and topics.

• Polynomial time algorithms with provable bounds on recovery error

• Empirical Validation: better than LightLDA in quality and time

• Scalable Implementations: more to come.
New Projects

• Flash algorithms
• Nested Dataflow
Potential Applications in Microsoft

• Bing Ads: Snippet Generation
• Office Substrate
• Triage for customer complaints.
NMF with Realistic Noise

- Same assumptions on $B, C$ (the factors) as in Topic Modeling, EXCEPT
  - We do not assume the data points are stochastically generated.

- Earlier provable algorithms assumed instead of Stochastic generation a strict noise model:
  - Noise $A - BC$ in each data point $<<$ data.
  - Violated by most points for Topic Models!

- Bhattacharyya, Goyal, Kannan, Pani (ICML 2016) assume:
  - Subset Noise: For any subset of $1/10$ th of the data points,
  - Noise in average of subset in $\leq \varepsilon$ (average of the data). Holds for many situations.
  - Same Algorithm as for Topic Modeling works. Proof quite different (because there is no stochastic model). More efficient than previous algs.
Thank you