Reducing the Sampling Complexity of Topic Models

Aaron Li

joint work with Amr Ahmed, Sujith Ravi, Alex Smola

CMU and Google
Outline

• Topic Models
  • Inference algorithms
  • Losing sparsity at scale
• Inference algorithm
  • Metropolis Hastings proposal
  • Walker’s Alias method for $O(k_d)$ draws
• Experiments
  • LDA, Pitman-Yor topic models, HPYM
  • Distributed inference
Models
Clustering & Topic Models

Latent Dirichlet Allocation

- \( \alpha \) (language prior)
- \( \Theta_i \) (topic label)
- \( Z_{ij} \) (topic probability)
- \( \psi_k \) (instance)
- \( \beta \) (topic probability)
- \( W_{ij} \) (instance)
The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. “Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services,” Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center’s share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.
Collapsed Gibbs Sampler (Griffiths & Steyvers, 2005)

\[ p(z_{ij} = t \mid \text{rest}) = \]
\[ \frac{n^{-ij}(t, d) + \alpha_t}{n^{-ij}(d) + \sum_t \alpha_t} \times \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w} \]
For each document do
- For each word in the document do
  - Resample topic for the word
- Update (document, topic) table
- Update (word, topic) table

\[
(n^{-ij}(t, d) + \alpha_t) \times \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \beta}
\]
Exploiting Sparsity
(Yao, Mimno, Mccallum, 2009)

• For each document do
  • For each word in the document do
    • Resample topic for the word

"constant"

\[
\frac{\alpha_t \beta_w}{n^{-ij}(t) + \beta} + n^{-ij}(t, d) \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \beta} + n^{-ij}(t, w) \frac{\alpha_t}{n^{-ij}(t) + \beta}
\]

sparse for most documents

sparse for small collections

• Update (document, topic) table
• Update (word,topic) table

amortized \(O(k_d + k_w)\) time
Exploiting Sparsity (Yao, Mimno, Mccallum, 2009)

- For each document do
  - For each word in the document do
    - Resample topic for the word

\[
\frac{\alpha_t \beta_w}{n^{-ij}(t) + \beta} + \frac{n^{-ij}(t, d) \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \beta}}{n^{-ij}(t, w) \frac{\alpha_t}{n^{-ij}(t) + \beta}} + \frac{n^{-ij}(t, w) \frac{\alpha_t}{n^{-ij}(t) + \beta}}{n^{-ij}(t, w) \frac{\alpha_t}{n^{-ij}(t) + \beta}}
\]

- Update (document, topic) table
- Update (word,topic) table

"constant" sparse for most documents dense for large collections

we solve this problem

Carnegie Mellon University
• **LDA**

\[ p(z_{di} = t, r_{di} = 0 | \text{rest}) \propto \frac{\alpha_t + n_{dt}}{b_t + m_t} \frac{m_{tw} + 1 - s_{tw}}{m_{tw} + 1} \frac{S_{s_{tw} + 1}^{m_{tw}}}{S_{s_{tw}, a_t}^{m_{tw}}} \]

\[ p(z_{di} = t, r_{di} = 1 | \text{rest}) \propto (\alpha_t + n_{dt}) \frac{b_t + a_t s_t}{b_t + m_t} \frac{s_{tw} + 1}{m_{tw} + 1} \frac{\gamma + s_{tw}}{\gamma + s_t} \frac{S_{s_{tw} + 1}^{m_{tw}}}{S_{s_{tw}, a_t}^{m_{tw}}} \]

• **Poisson-Dirichlet Process**

More Models

- **LDA**

- **Poisson-Dirichlet Process**
More Models

• LDA

\[ \alpha \rightarrow \theta_d \rightarrow z_{di} \rightarrow w_{di} \rightarrow \psi_k \rightarrow \beta \]

for all \( d \)

for all \( k \)

for all \( i \)

• Hierarchical-Dirichlet Process

\[ H \rightarrow \theta_0 \rightarrow \theta_d \rightarrow z_{di} \rightarrow w_{di} \rightarrow \psi_k \rightarrow \beta \]

for all \( k \)

for all \( d \)

for all \( i \)

... even more mess for topic distribution
Key Idea of the Paper

- LDA

Approximate slowly changing distribution by fixed distribution. Use Metropolis Hastings

Amortized O(1) time proposals
Lazy decomposition

- Exploiting topic sparsity in documents

\[
\begin{align*}
(n^{-ij}(t, d) + \alpha_t) \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w} \\
= n^{-ij}(t, d) \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w} + \alpha_t \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w}
\end{align*}
\]

- Normalization costs \(O(k)\) operations!

Sparse \(O(k_d)\) time samples

Often dense but slowly varying
Lazy decomposition

- Exploiting topic sparsity in documents

\[
(n^{-ij}(t, d) + \alpha_t) \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w} = n^{-ij}(t, d) \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w} + \alpha_t \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w}
\]

- Normalization costs \(O(k_d + 1)\) operations!
Lazy decomposition

- Exploiting topic sparsity in documents

\[
(n^{-ij}(t, d) + \alpha_t) \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w}
= n^{-ij}(t, d) \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w} + \alpha_t \frac{n^{-ij}(t, w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w}
\approx q(t|d) + q(t|w)
\]

- Normalization costs \(O(k_d + 1)\) operations!
Metropolis Hastings with stationary proposal distribution

• We want to sample from $p$ but only have $q$

• **Metropolis Hastings**
  • Draw $x$ from $q(x)$ and accept *move* from $x'$
  
  \[
  \min \left(1, \frac{p(x)}{p(x')} \frac{q(x')}{q(x)} \right)
  \]

• We only need to evaluate ratios of $p$ and $q$

• **This is a chain. It mixes rapidly in experiments.**
Application to Topic Models

• Recall - we split topic probability

\[ q(t) \propto q(t|d) + q(t|w) \]

- Sparse
- Dense but static

• Dense part has normalization precomputed
• Sparse part can easily be normalized
• Sample from \( q(t) \) and evaluate \( p(t|w,d) \) only for the draws
In a nutshell

\[ q(t) \propto q(t|d) + q(t|w) \]

- Sparse part for document (topics, topic hierarchy, etc.)
  Evaluate this exactly
- Dense part for generative model (language, images, ...)
  Approximate this by stale model
- Metropolis Hastings sampler to correct
- Need fast way to draw from stale model
ALIAS

Sampling
Walker’s Alias Method

- Draw from discrete distribution in $O(1)$ time
- Requires $O(n)$ preprocessing
  - Group all $x$ with $n \cdot p(x) < 1$ into $L$ (rest in $H$)
  - Fill each of the small ones up by stealing from $H$. This yields $(i,j, p(i))$ triples.
- Draw from uniform over $n$, then from $p(i)$
Probability distribution

2

\( \frac{4}{3} \)

\( \frac{1}{3} \)

\( \frac{1}{3} \)

Courtesy of keithschwartz.com
Probability distribution

Splitting

Courtesy of keithschwartz.com
Probability distribution

Filling up (4) with (1)

Courtesy of keithschwartz.com
Probability distribution

Filling up (3) with (1)

Courtesy of keithschwartz.com
Probability distribution

Filling up (1) with (2)

Courtesy of keithschwartz.com
• Conditional topic probability

\[ q(t) \propto q(t|d) + q(t|w) \]

\( k_d \) Sparse \hspace{1cm} Dense but static

• Use Walker’s method to draw from \( q(t|w) \)
• After \( k \) draws from \( q(t|w) \) recompute with current value
• Amortized \( O(1 + k_d) \) sampler
Experiments
LDA: Varying the number of topics (4k)

Figure 4: Comparison of SparseLDA and AliasLDA on GPOL when varying the number of topics for $k = \{256, 1024, 2048, 4096\}$.

Percentage of full PubMedSmall collection

Seconds per iteration

Figure 5: Average runtime per iteration when compared on $\{10\%, 20\%, 40\%, 75\%, 100\%\}$ of the PubMedSmall dataset for SparseLDA and AliasLDA.

The gap in performance is especially large for more sophisticated language models such as PDP and HDP. The running time for each Gibbs iteration is reduced by 60% to 80% for PDP, and 80% to 95% for HDP, an order of magnitude on improvement.

5.5 Varying the number of topics

When the number of topics $k$ increases, the running time for an iteration of AliasLDA increases at a much lower rate than SparseLDA, as seen from Figure 4 on dataset GPOL since $k_d$ is almost constant. Even though the gap between SparseLDA and AliasLDA may seem insignificant at $k = 1024$, it becomes very pronounced at $k = 2048$ (45% improvement) and at $k = 4096$ (over 100%). This confirms the observation above that shorter documents benefit more from AliasLDA in the sense that the average document length $L / D$ relative to the number of topics $k$ becomes "shorter" as $k$ increases. This yields a more sparse $n$ and lower $k_d$ for a document $d$ on average.

5.6 Varying the corpus size

Figure 5 demonstrates how the gap in running time speeds scales with growing number of documents in the same domain. We measure the average runtime for the first 50 Gibbs iterations on 10%, 20%, 40%, 75%, and 100% of PubMedSmall dataset. The speedup ratio for each subset is at 31%, 34%, 37%, 41%, 43% respectively. In other words, it increases with the amount of data, which conforms our intuition that adding new documents increases the density of $n_{tw}$, thus slowing down the sparse sampler much more than the alias sampler, since the latter only depends on $k_d$ rather than $k_d + k_w$.

Perplexity vs. Runtime

Perplexity vs. Iterations

Figure 6: Perplexity as a function of runtime (left) and number of iterations (right) for LDA, SparseLDA, and LDA, PDP and HDP, both with and without using the Alias method. We see considerable acceleration at unchanged perplexity.

6. CONCLUSION

In this paper, we described an approach that effectively reduces sampling complexity of topic models from $O(k_t)$ to $O(k_d)$ in general, and from $O(k_d + k_w)$ (SparseLDA) to $O(k_d)$ (AliasLDA) for LDA topic model. Empirically, we showed that our approach scales better than existing state-of-the-art method when the number of topics and the number of documents become large. This enables many large scale applications, and many existing applications which require a
LDA: Varying data size

![Graph showing speed of SparseLDA and AliasLDA as a function of percentage of full PubMedSmall collection]
HDP & PDP

RS (321K tokens)  GPOL (2.6M tokens)  Enron (6M tokens)
5.5 Varying the number of topics

When the number of topics $k$ increases, the running time for an iteration of AliasLDA increases at a much lower rate than SparseLDA, as seen from Figure 4 on dataset GPOL since $k_d$ is almost constant. Even though the gap between SparseLDA and AliasLDA may seem insignificant at $k = 1024$, it becomes very pronounced at $k = 2048$ (45% improvement) and at $k = 4096$ (over 100%). This confirms the observation above that shorter documents benefit more from AliasLDA in the sense that the average document length $L/D$ relative to the number of topics $k$ becomes "shorter" as $k$ increases. This yields a more sparse $n_d$ and lower $k_d$ for a document $d$ on average.

5.6 Varying the corpus size

Figure 5 demonstrates how the gap in running time speeds scales with growing number of documents in the same domain. We measure the average runtime for the first 50 Gibbs iterations on 10%, 20%, 40%, 75%, and 100% of PubMedSmall dataset. The speedup ratio for each subset is at 31%, 34%, 37%, 41%, 43% respectively. In other words, it increases with the amount of data, which conforms our intuition that adding new documents increases the density of $n_w$, thus slowing down the sparse sampler much more than the alias sampler, since the latter only depends on $k_d$ rather than $k_d + k_w$.

6. CONCLUSION

In this paper, we described an approach that effectively reduces sampling complexity of topic models from $O(k)$ to $O(k_d)$ in general, and from $O(k_d + k_w)$ (SparseLDA) to $O(k_d)$ (AliasLDA) for LDA topic model. Empirically, we showed that our approach scales better than existing state-of-the-art method when the number of topics and the number of documents become large. This enables many large scale applications, and many existing applications which require a quality
Summary

• Extends Sparse LDA concept of Yao et al.’09
• Works for any sparse document model
• Useful for many emissions models (Pitman Yor, Gaussians, etc.)

• Metropolis-Hastings-Walker
  • MH proposals on stale distribution
  • Recompute proposal after k draws for O(1)
• Fastest LDA sampler by a large margin
And now in parallel

Sparse LDA on 60k cores (0.1% of a nuclear reactor)
Mu Li et al, 2014, OSDI
Saving Nuclear Power Plants

AliasLDA vs SparseLDA Convergence
1000 Clients, 400 Servers, 1000 Shards
(49 Billion Tokens)

Aaron Li et al, submitted
Saving Nuclear Power Plants

Speed does not improve over time. It is machines are shutting down because algorithm is too slow...

1 machine alive
Min = Avg