Web-scale Topic Models in Spark: An Asynchronous Parameter Server
Rolf Jagerman
Outline

Introduction

Parameter Server

Distributed LDA

Experiments

Conclusion
Introduction

Parameter Server

Distributed LDA

Experiments

Conclusion
Introduction

Problem statement

- Compute a topic model on the ClueWeb12 data set
  - ClueWeb12 is a 27-terabyte Web crawl
  - Contains 733 million web pages
- Be able to infer many topics (100s or 1000s)
- Using Spark
  - Spark provides existing tools and infrastructure to distribute and process the ClueWeb12 data set.
Introduction

Spark

Distributed computing framework using map-reduce
Introduction

Spark

- **Fault-tolerant**
  Lineage-graph allows recomputation in case of failure
  Checkpoint system for long tasks

- **Fast**
  Attempts to cache and keep computed data in memory
  Locality-aware processing

- **Strong community**
  Many open-source projects, tools, research, ...
Introduction
LDA Topic Modeling

Introduction

LDA Topic Modeling

Topic indicators $z$ are initialized uniformly at random at the start. Collapsed Gibbs Sampling\(^1\) repeatedly draws new topics $z$:

$$z_{new} \sim P(z = k) \propto (n_{dk}^{-dw} + \alpha) \frac{n_{wk}^{-dw} + \beta}{n_{k}^{-dw} + V \beta}$$

- $n_{wk}$: number of times topic $k$ is assigned to word $w$
- $n_k$: number of times topic $k$ is assigned to any word
- $n_{dk}$: number of times topic $k$ is assigned to a token in document $d$

Introduction

LDA Topic Modeling

\[
\begin{align*}
\text{for all } d \in \mathcal{D} \text{ do} \\
\text{for all } (\text{word } w, \text{ topic assignment } z_{\text{old}}) \in d \text{ do} \\
\quad z_{\text{new}} \sim P(z = k) \propto (n_{dk}^{-dw} + \alpha) \frac{n_{wk}^{-dw} + \beta}{n_k^{-dw} + \nu \beta} \\
\quad n_k \leftarrow n_k - 1 \\
\quad n_{wk} \leftarrow n_{wk} - 1 \\n\quad n_{dk} \leftarrow n_{dk} - 1 \\
\quad n_k \leftarrow n_k + 1 \\
\quad n_{wk} \leftarrow n_{wk} + 1 \\
\quad n_{dk} \leftarrow n_{dk} + 1
\end{align*}
\]

for } k = z_{\text{old}}

for } k = z_{\text{new}}
Introduction

Parameter Server

Distributed LDA

Experiments

Conclusion
A parameter server provides
- An architecture for large distributed matrices and vectors
- A shared interface to the values of the matrix/vector

Used for storing $n_{wk}$ and $n_k$:
- Distributed matrix $n_{wk}$: the word-topic counts
- Distributed vector $n_k$: the global topic counts
Parameter Server

Client Actions
- Pull(rows, columns)
- Push(rows, columns, values)

Virtual Matrix View

Physical Data Storage
- Parameter Server 1
- Parameter Server 2
- Parameter Server 3
Parameter Server

A **pull** is an asynchronous read operation
Parameter Server

A **push** is an asynchronous write operation

- How to handle many simultaneous writes?
- Locks are very slow over the network
- Avoid conflicts by aggregating writes via **addition**
  - Gibbs sampling increments/decrements counters $n_{wk}$ and $n_k$.
  - Commutative: $a + b = b + a$
  - Associative: $(a + b) + c = a + (b + c)$
Introduction

Parameter Server

Distributed LDA

Experiments

Conclusion
Distributed LDA

Regular collapsed Gibbs sampler:

- Sample from $P(z = k) \propto (n_{dk}^{-d_w} + \alpha) \frac{n_{wk}^{-d_w} + \beta}{n_k^{-d_w} + V \beta}$

- Costly to sample for every token when number of topics $K$ is large

Instead, use Metropolis-Hastings sampler (LightLDA):

- Sample from cheap proposal distributions
- Either accept or reject the proposal with a certain probability.
Distributed LDA

LightLDA uses a factorized strategy
It alternates between a doc-proposal and word-proposal

\[
P(z = k) \propto \left( n_{dk}^{-dw} + \alpha \right) \frac{n_{wk}^{-dw} + \beta}{n_k^{-dw} + V\beta}
\]

\[
\text{doc-proposal } P_d
\]

\[
\text{word-proposal } P_w
\]
Distributed LDA

Sampling document distribution \( (n_{dk}^{dw} + \alpha) \)
Distributed LDA

Sampling document distribution \((n_d^{dw} - n_{dk} + \alpha)\)

\(n_d + K\alpha\)

Draw from current topic assignments array

\(z\)

Draw uniformly at random
Distributed LDA

Sampling word distribution $P_w \propto \frac{n_{w|k}^{-d_w} + \beta}{n_k^{-d_w} + V \beta}$:

- Can be reused for many tokens in the corpus
- Can achieve amortized $O(1)$ time using Alias Tables.
Distributed LDA

Sampling word distribution \( \frac{n_{w_k}^{-d_w} + \beta}{n_k^{-d_w} + V \beta} \) using Alias Tables\(^2\)

Distributed LDA
Distributed LDA
Distributed LDA
Distributed LDA

- Parameter Server 1
- Parameter Server 2
- Parameter Server 3

Worker

Dataset RDD

Resample tokens using Metropolis-Hastings
Distributed LDA

- Parameter Server 1
- Parameter Server 2
- Parameter Server 3

Dataset RDD

Worker

Resample tokens using Metropolis-Hastings
Push changes asynchronously to parameter servers
Experiments

Performed two types of experiments:

- Comparison to current Spark implementations on 200GB
- Running the system on the full ClueWeb12 corpus (27TB)

Using a computing cluster with 30 nodes:

- $30 \times 2\text{TB HDD} = 60\text{TB HDD}$
- $30 \times 128\text{GB RAM} = 3.84\text{TB RAM}$
- $30 \times 16\text{ cores} = 480\text{ cores}$
- Connected over 10Gbps network
Experiments

Comparison to Spark implementations (Spark EM LDA, Spark Online LDA)

- At most 10% of ClueWeb12 B13 (200GB)
- Vary number of topics $K$
- Vary data set size 2.5% ... 10%
- $\alpha = 0.05$
- $\beta = 0.001$
- Measure perplexity and runtime
Experiments
Comparison to Spark

![Graph showing comparison between different implementations of LDA with respect to Perplexity and Runtime. The x-axis represents the number of topics K, and the y-axis represents Perplexity and Runtime in minutes. The graph compares our implementation, Spark EM LDA, and Spark online LDA.]
Experiments
Comparison to Spark

Perplexity

Runtime (minutes)

Fraction of ClueWeb12 B13 data set
Experiments

Compute a 1000-topic LDA model on the full ClueWeb12 corpus

- About 95% of ClueWeb12 data set
- $K = 1000$
- $\alpha = 0.05$
- $\beta = 0.01$
- Measure system metrics
- Measure perplexity on training set (convergence curve)
Experiments

![CPU Usage Chart]

![Gigabytes/second Chart]
Experiments

- Network usage
- Perplexity
Experiments

$\beta = 0.001$
Experiments

<table>
<thead>
<tr>
<th>Topic #984</th>
<th>Topic #405</th>
<th>Topic #941</th>
<th>Topic #741</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘Tourism’</td>
<td>‘Video games’</td>
<td>‘Programming’</td>
<td>‘Biology’</td>
</tr>
<tr>
<td>hotel</td>
<td>allianc</td>
<td>write</td>
<td>tag</td>
</tr>
<tr>
<td>locat</td>
<td>hord</td>
<td>docum</td>
<td>protein</td>
</tr>
<tr>
<td>car</td>
<td>euro</td>
<td>var</td>
<td>hypothet</td>
</tr>
<tr>
<td>holidai</td>
<td>warcraft</td>
<td>return</td>
<td>gene</td>
</tr>
<tr>
<td>area</td>
<td>wow</td>
<td>prop</td>
<td>cytoplasm</td>
</tr>
<tr>
<td>golf</td>
<td>gold</td>
<td>function</td>
<td>prk</td>
</tr>
<tr>
<td>wed</td>
<td>warhamm</td>
<td>substr</td>
<td>locu</td>
</tr>
<tr>
<td>room</td>
<td>starcraft</td>
<td>indexof</td>
<td>note</td>
</tr>
</tbody>
</table>
Conclusion
Conclusion

- The system is much more scalable than the default Spark LDA implementations without sacrificing model quality
  - Using a parameter server to store $n_{wk}$ and $n_k$
  - Using the LightLDA algorithm for an $O(1)$ sampling complexity
- Computed a 1000-topic LDA model on the ClueWeb12 corpus
Conclusion

Future work:

- Apply the 1000-topic topic model to extrinsic evaluation tasks such as search, information retrieval or word sense disambiguation
- Improve the parameter server for better reliability (e.g. distributed hash tables for instantaneous failover)
- Improve scalability by implementing sparse representations