Mr. LDA: A Flexible Large Scale Topic Modeling Package using Variational Inference in MapReduce

Ke Zhai, Jordan Boyd-Graber, Nima Asadi, and Mohamad Alkhouja
Mr. LDA: A Flexible Large Scale Topic Modeling Package using Variational Inference in MapReduce

Ke Zhai, Jordan Boyd-Graber, Nima Asadi, and Mohamad Alkhouja
Introductions

MR LDA

- MR = MapReduce
- LDA = latent Dirichlet allocation
- MR LDA = Ke

First author

Immigration issues prevented presentation
Introductions

MR LDA

- MR = MapReduce
- LDA = latent Dirichlet allocation
- MR LDA = Ke

First author

Immigration issues prevented presentation
Roadmap

- Review of topic models
- The need for scalability
- Variational inference vs. Gibbs sampling
- MR LDA
  - A scalable topic modeling package
  - Using variational inference
- Extensions
  - Extending psychologically-inspired word lists
  - Discovering topics consistent across languages
Why topic models?

- Suppose you have a huge number of documents
- Want to know what’s going on
- Can’t read them all (e.g. every New York Times article from the 90’s)
- Topic models offer a way to get a corpus-level view of major themes
Why topic models?

- Suppose you have a huge number of documents
- Want to know what’s going on
- Can’t read them all (e.g. every New York Times article from the 90’s)
- Topic models offer a way to get a corpus-level view of major themes
- Unsupervised
From an input corpus and number of topics $K \rightarrow$ words to topics
Conceptual Approach

From an input corpus and number of topics $K \rightarrow \text{words to topics}$

**TOPIC 1**
- computer, technology, system, service, site, phone, internet, machine

**TOPIC 2**
- sell, sale, store, product, business, advertising, market, consumer

**TOPIC 3**
- play, film, movie, theater, production, star, director, stage
Conceptual Approach

- For each document, what topics are expressed by that document?

- Red Light, Green Light: A 2-Tone L.E.D. to Simplify Screens
- The three big Internet portals begin to distinguish among themselves as shopping malls
- Stock Trades: A Better Deal For Investors Isn't Simple
- Forget the Bootleg, Just Download the Movie Legally
- The Shape of Cinema, Transformed At the Click of a Mouse
- Multiplex Heralded As Linchpin To Growth
- A Peaceful Crew Puts Muppets Where Its Mouth Is
Topic Models: What’s Important

- **Topic models**
  - Topics to words - multinomial distribution
  - Documents to topics - multinomial distribution

- **Statistical structure inferred from data**

- **Have semantic coherence because of language use**

- **We use latent Dirichlet allocation (LDA) [Blei et al. 2003], a fully Bayesian version of pLSI [Hofmann 1999], probabilistic version of LSA [Landauer and Dumais 1997]**
Applications

Computer Vision [Li Fei-Fei and Perona 2005]
Applications

Social Networks [Airoldi et al. 2008]
Applications

Music [Hu and Saul 2009]

Figure 2: The C major and C minor key-profiles learned by our model, as encoded by the matrix. Resulting key-profiles are obtained by transposition.

Figure 3: Key judgments for the first 6 measures of Bach's Prelude in C minor, WTC-II. Annotations for each measure show the top three keys (and relative strengths) chosen for each measure. The top set of three annotations are judgments from our LDA-based model; the bottom set of three are from human expert judgments [3].

We identified with the 24 major and minor modes of classical western music. We note that our approach is still regarded as unsupervised because we do not learn from labeled or annotated data.

2.3 Results & Applications

Our learnt key-profiles are shown in Figure 2. We note that these key-profiles are consistent with music theory principals: In both major and minor modes, weights are given in descending order to degrees of the triad, diatonic, and finally chromatic scales. Intuitively, these key-profiles represent the underlying distributions that are used to characterize all the songs in the corpus.

We also show how to do key-finding and modulation-tracking using the representations learned by our model. The goal of key-finding is to determine the overall key of a musical piece, given the notes of the composition. Since the key weight vector represents the most likely keys present in each song, we classify each song as the key that is given the largest weight in . A related task is modulation-tracking, which identifies where the modulations occur within a piece. We achieve this by determining the key of each segment from the most probable values of its topic latent variable $z$.

We estimated our model from a collection of 235 MIDI files compiled from classicalmusicmidipage.com. The collection included works by Bach, Vivaldi, Mozart, Beethoven, Chopin, and Rachmaninoff. These composers were chosen to span the baroque through romantic periods of western, classical music. Our results for key-finding achieved an accuracy of 86%, out-performing several other key-finding algorithms, including the popular KS model [3]. We also show in Figure 3 that our annotations for modulation-tracking are comparable to those given by music theory experts. More results can be found in our paper [1].
Why large-scale?

- The most interesting datasets are the **big** ones
- These datasets don’t fit on a single machine
- Thus we can’t depend on analysis that sits on a single machine
MapReduce

- Framework proposed by Google [Dean and Ghemawat 2004]
- Hadoop, OSS implementation by Yahoo [White 2010]
- Central concept
  - **Mappers** process small units of data
  - **Reducers** aggregate / combine results of mappers into final result
  - **Drivers** Run a series of jobs to get the work done
  - Overall framework distributes intermediate results where they need to go
Outline

1. Topic Model Introduction
2. Inference
3. Extensions
Inference
Generative models tell a story of how your data came to be.

There are missing pieces to that story (e.g. the topics).

Statistical inference fills in the missing pieces.
Generative models tell a story of how your data came to be.

There are missing pieces to that story (e.g. the topics).

Statistical inference fills in the missing pieces.

Hard problem - requires looking at the entire dataset.

Why we need large scale solutions.
Generative models tell a story of how your data came to be.

There are missing pieces to that story (e.g., the topics).

Statistical inference fills in the missing pieces.

Hard problem - requires looking at the entire dataset.

Why we need large scale solutions.

Use MapReduce!
### Inference

<table>
<thead>
<tr>
<th>Variational</th>
<th>MCMC / Gibbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Few, expensive iterations</td>
<td>Many, cheap iterations</td>
</tr>
<tr>
<td>Deterministic</td>
<td>Random</td>
</tr>
<tr>
<td>Conjugate easier, tractable without</td>
<td>Effective for <strong>conjugate</strong> distributions</td>
</tr>
<tr>
<td>Easy convergence diagnosis</td>
<td>Tricky convergence diagnosis</td>
</tr>
</tbody>
</table>
Inference

Variational

- First LDA implementation
  \cite{Blei:2003}
- Master-Slave LDA
  \cite{Nallapati:2007}
- Apache Mahout

MCMC / Gibbs

- Popular
  \cite{Griffiths:2004}
- Sparsity helps \cite{Yao:2009}
- Assume shared memory?
  \cite{Asuncion:2008}
- YahooLDA
  \cite{Smola:2010}
Expectation Maximization Algorithm

- Input: $z$ (hidden variables), $\xi$ (parameters), $D$ (data)
- Start with initial guess of $z$, parameters $\xi$
- Repeat
  - Compute the expected value of latent variables $z$
  - Compute the parameters $\xi$ that maximize likelihood $L$ (use calculus)
- With each iteration, objective function $L$ goes up
Expectation Maximization Algorithm

- Input: \( z \) (hidden variables), \( \xi \) (parameters), \( D \) (data)
- Start with initial guess of \( z \), parameters \( \xi \)
- Repeat
  - **E-Step** Compute the expected value of latent variables \( z \)
  - Compute the parameters \( \xi \) that maximize likelihood \( L \) (use calculus)
- With each iteration, objective function \( L \) goes up
Expectation Maximization Algorithm

- Input: \( z \) (hidden variables), \( \xi \) (parameters), \( D \) (data)
- Start with initial guess of \( z \), parameters \( \xi \)
- Repeat
  - E-Step Compute the expected value of latent variables \( z \)
  - M-Step Compute the parameters \( \xi \) that maximize likelihood \( L \) (use calculus)
- With each iteration, objective function \( L \) goes up
Sometimes you can’t actually optimize $L$
So we instead optimize a lower bound based on a “variational” distribution $q$

$$L = \mathbb{E}_q [\log (p(D|Z)p(Z|\xi))] - \mathbb{E}_q [\log q(Z)]$$

This is called variational EM (normal EM is when $p = q$)

Makes the math possible to optimize $L$
Variational distribution

(a) LDA

(b) Variational
Variational distribution

(c) LDA

(d) Variational
Updates - Important Part

- $\phi$ How much the $n^{th}$ word in a document expressed topic $k$
- $\gamma_{d,k}$ How much the $k^{th}$ topic is expressed in a document $d$
- $\beta_{v,k}$ How much word $v$ is associated with topic $k$

\[
\phi_{d,n,k} \propto \beta_{w_{d,n},k} \cdot e^{\psi(\gamma_k)}
\]
\[
\gamma_{d,k} = \alpha_k + \sum_{n=1}^{N_d} \phi_{d,n,k},
\]
\[
\beta_{v,k} \propto \eta + \sum_{d=1}^{C} (w_{v}^{(d)} \phi_{d,v,k})
\]

This is the algorithm!
Updates - Important Part

- \( \phi \) How much the \( n^{th} \) word in a document expressed topic \( k \) (Mapper)
- \( \gamma_{d,k} \) How much the \( k^{th} \) topic is expressed in a document \( d \) (Mapper)
- \( \beta_{v,k} \) How much word \( v \) is associated with topic \( k \)

\[
\phi_{d,n,k} \propto \beta_{w_{d,n,k}} \cdot e^{\psi(\gamma_k)}
\]
\[
\gamma_{d,k} = \alpha_k + \sum_{n=1}^{N_d} \phi_{d,n,k},
\]
\[
\beta_{v,k} \propto \eta + \sum_{d=1}^{C} (w_{v}^{(d)} \phi_{d,v,k})
\]

This is the algorithm!
Updates - Important Part

- $\phi$ How much the $n^{th}$ word in a document expressed topic $k$ (Mapper)
- $\gamma_{d,k}$ How much the $k^{th}$ topic is expressed in a document $d$ (Mapper)
- $\beta_{v,k}$ How much word $v$ is associated with topic $k$ (Reducer)

\[
\phi_{d,n,k} \propto \beta_{w_{d,n},k} \cdot e^{\psi(\gamma_k)}
\]
\[
\gamma_{d,k} = \alpha_k + \sum_{n=1}^{N_d} \phi_{d,n,k},
\]
\[
\beta_{v,k} \propto \eta + \sum_{d=1}^{C} (w_{v}^{(d)} \phi_{d,v,k})
\]

This is the algorithm!
Updates - Important Part

- $\phi$ How much the $n^{th}$ word in a document expressed topic $k$ (Mapper)
- $\gamma_{d,k}$ How much the $k^{th}$ topic is expressed in a document $d$ (Mapper)
- $\beta_{v,k}$ How much word $v$ is associated with topic $k$ (Reducer)

This is the algorithm!
Other considerations

- Thus far, no difference from Mahout or [Nallapati et al. 2007]
- Computing objective function $\mathcal{L}$ to assess convergence
- Updating hyperparameters
  - Many implementations don’t do this
  - Critical for topic quality and good likelihood
Expanding Equation 1 gives us $\mathcal{L}(\gamma, \phi; \alpha, \beta)$ for one document:

\[
\mathcal{L}(\gamma, \phi; \alpha, \beta) = \sum_{d=1}^{C} \mathcal{L}_d(\gamma, \phi; \alpha, \beta)
\]

\[
= \sum_{d=1}^{C} \mathcal{L}_d(\alpha) + \sum_{d=1}^{C} \left( \mathcal{L}_d(\gamma, \phi) + \mathcal{L}_d(\phi) + \mathcal{L}_d(\gamma) \right),
\]

\[
\text{Driver} \quad \text{computed in mapper} \quad \text{computed in Reducer}
\]
Updating hyperparameters

We use a Newton-Raphson method which requires the Hessian matrix and the gradient,

\[ \alpha_{\text{new}} = \alpha_{\text{old}} - \mathcal{H}^{-1}(\alpha_{\text{old}}) \cdot g(\alpha_{\text{old}}), \]
We use a Newton-Raphson method which requires the Hessian matrix and the gradient,
\[
\alpha_{\text{new}} = \alpha_{\text{old}} - \mathcal{H}^{-1}(\alpha_{\text{old}}) \cdot g(\alpha_{\text{old}}),
\]
where the Hessian matrix \( \mathcal{H} \) and gradient \( g(\alpha) \) are
\[
\mathcal{H}(k, l) = \delta(k, l) C \Psi'(\alpha_k) - C \Psi' \left( \sum_{l=1}^{K} \alpha_l \right),
\]
\[
g(k) = C \left( \Psi \left( \sum_{l=1}^{K} \alpha_l \right) - \Psi(\alpha_k) \right) + \sum_{d=1}^{C} \Psi(\gamma_{d,k}) - \Psi \left( \sum_{l=1}^{K} \gamma_{d,l} \right).
\]

**Complexity**

Removing document-dependence: update \( O(K^2) \) in the driver
Other implementation details

- Computing \( \psi \) function is expensive, so we cache approximate values.
- The number of intermediate values swamp the system, so we employ in-mapper combiners [Lin and Dyer 2010].
- Initialization.
Other implementation details

- Computing $\Psi$ function is expensive, so we cache / approximate values
  - Always helps
- The number of intermediate values swamp the system, so we employ in-mapper combiners [Lin and Dyer 2010]
  - Only helps with many topics
- Initialization
  - Helps in first iterations
Comparison with Mahout

Held-out likelihood vs. time (sec)
TREC (100 topics, 500k documents)
Outline

1. Topic Model Introduction
2. Inference
3. Extensions
How are psychological factors expressed in blogs?

- Linguistic Inquiry in Word Count [Pennebaker and Francis 1999]
- Example psychological processes:
  - Anger: hate, kill, annoyed
  - Negative Emotions: hurt, ugly, nasty
- What words cooccur with these words in a particular corpus?
How are psychological factors expressed in blogs?

- Linguistic Inquiry in Word Count [Pennebaker and Francis 1999]
- Example psychological processes:
  - Anger: hate, kill, annoyed
  - Negative Emotions: hurt, ugly, nasty
- What words cooccur with these words in a particular corpus?
- Use LIWC categories as an informed prior to “seed” topics

\[
\beta_{v,k} \propto \eta_{v,k} + \sum_{d=1}^{C} (w_{v}^{(d)} \phi_{d,v,k})
\]
How are psychological factors expressed in blogs?

- Linguistic Inquiry in Word Count [Pennebaker and Francis 1999]
- Example psychological processes:
  - Anger: hate, kill, annoyed
  - Negative Emotions: hurt, ugly, nasty
- What words cooccur with these words in a particular corpus?
- Use LIWC categories as an informed prior to “seed” topics

\[
\beta_{v,k} \propto \eta_{v,k} + \sum_{d=1}^{C} (w_{v}^{(d)} \phi_{d,v,k})
\]

- Not possible in SparseLDA-based models
Workflow for Informed Prior

[Diagram showing the workflow with nodes labeled 'Parameters', 'Document', 'Map: Update γ, φ', 'Inf. Prior', 'Reducer', 'Driver: Update α', 'Test Likelihood Convergence', 'Distributed Cache', and 'Write λ', 'Write α', 'Write λ'].

- Parameters
- Document
- Map: Update γ, φ
- Inf. Prior
- Reducer
- Driver: Update α
- Test Likelihood Convergence
- Distributed Cache
- Write λ
- Write α
- Write λ

Steps:
4. Sufficient Statistics for λ Update
5. Hessian Terms
6. Test Likelihood Convergence
Psychologically-Informed Topics from Blogs

<table>
<thead>
<tr>
<th>Affective Processes</th>
<th>Negative Emotions</th>
<th>Positive Emotions</th>
<th>Anxiety</th>
<th>Anger</th>
<th>Sadness</th>
</tr>
</thead>
<tbody>
<tr>
<td>easili</td>
<td>sorri</td>
<td>lord</td>
<td>bird</td>
<td>iraq</td>
<td>level</td>
</tr>
<tr>
<td>dare</td>
<td>crappi</td>
<td>prayer</td>
<td>diseases</td>
<td>american</td>
<td>grief</td>
</tr>
<tr>
<td>truli</td>
<td>bullshit</td>
<td>pray</td>
<td>shi</td>
<td>countri</td>
<td>disord</td>
</tr>
<tr>
<td>lol</td>
<td>goddamn</td>
<td>merci</td>
<td>infect</td>
<td>militari</td>
<td>moder</td>
</tr>
<tr>
<td>needi</td>
<td>messi</td>
<td>etern</td>
<td>blood</td>
<td>nation</td>
<td>miseri</td>
</tr>
<tr>
<td>jealousi</td>
<td>shitti</td>
<td>truli</td>
<td>snake</td>
<td>unit</td>
<td>lbs</td>
</tr>
<tr>
<td>friendship</td>
<td>bitchi</td>
<td>humbl</td>
<td>anxieti</td>
<td>america</td>
<td>loneli</td>
</tr>
<tr>
<td>betray</td>
<td>angri</td>
<td>god</td>
<td>creatur</td>
<td>force</td>
<td>pain</td>
</tr>
</tbody>
</table>

Using 50 topics on Blog Authorship corpus [Koppel et al. 2006]
Polylingual LDA

- Assumes documents have multiple “faces” [Mimno et al. 2009]
- Topics also assumed to have per-language distribution
- As long as documents talk about the same thing, learns consistent topics across languages
- First variational inference algorithm
Workflow for Polylingual LDA

[Diagram visualizing the workflow for Polylingual LDA with nodes and arrows indicating the process steps.]
<table>
<thead>
<tr>
<th>English</th>
<th>German</th>
</tr>
</thead>
<tbody>
<tr>
<td>game</td>
<td>spiel</td>
</tr>
<tr>
<td>games</td>
<td>spieler</td>
</tr>
<tr>
<td>player</td>
<td>serie</td>
</tr>
<tr>
<td>players</td>
<td>erschien</td>
</tr>
<tr>
<td>released</td>
<td>gibt</td>
</tr>
<tr>
<td>comics</td>
<td>commics</td>
</tr>
<tr>
<td>characters</td>
<td>veröffentlicht</td>
</tr>
<tr>
<td>character</td>
<td>version</td>
</tr>
<tr>
<td>version</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>opera</th>
<th>musical</th>
<th>greek</th>
<th>turkish</th>
<th>cup</th>
<th>league</th>
<th>said</th>
<th>italian</th>
<th>soviet</th>
</tr>
</thead>
<tbody>
<tr>
<td>composer</td>
<td>region</td>
<td>club</td>
<td>could</td>
<td>pope</td>
<td>military</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>orchestra</td>
<td>hungarian</td>
<td>played</td>
<td>children</td>
<td>nitaly</td>
<td>union</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>piano</td>
<td>wine</td>
<td>football</td>
<td>death</td>
<td>catholic</td>
<td>russian</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>works</td>
<td>hungary</td>
<td>games</td>
<td>father</td>
<td>bishop</td>
<td>power</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>symphony</td>
<td>greece</td>
<td>career</td>
<td>wrote</td>
<td>roman</td>
<td>israel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>instruments</td>
<td>turkey</td>
<td>game</td>
<td>mother</td>
<td>rome</td>
<td>empire</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>composers</td>
<td>ottoman</td>
<td>championship</td>
<td>never</td>
<td>st</td>
<td>republic</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>gespielt</th>
<th>musik</th>
<th>ungarn</th>
<th>saison</th>
<th>frau</th>
<th>papst</th>
</tr>
</thead>
<tbody>
<tr>
<td>komponiert</td>
<td>turkei</td>
<td>gewann</td>
<td>the</td>
<td>rom</td>
<td>regierung</td>
</tr>
<tr>
<td>oper</td>
<td>turkischen</td>
<td>spielte</td>
<td>Familie</td>
<td>ii</td>
<td>republik</td>
</tr>
<tr>
<td>komponisten</td>
<td>griechenland</td>
<td>karriere</td>
<td>mutter</td>
<td>kirche</td>
<td>sowjetunion</td>
</tr>
<tr>
<td>werke</td>
<td>rumanien</td>
<td>fc</td>
<td>vater</td>
<td>di</td>
<td>kam</td>
</tr>
<tr>
<td>orchester</td>
<td>ungarischen</td>
<td>spielen</td>
<td>leben</td>
<td>bishof</td>
<td>krieg</td>
</tr>
<tr>
<td>wiener</td>
<td>griechischen</td>
<td>wechselte</td>
<td>starb</td>
<td>italien</td>
<td>land</td>
</tr>
<tr>
<td>komposition</td>
<td>istanbul</td>
<td>mannschaft</td>
<td>tod</td>
<td>italienisch</td>
<td>bevolkerung</td>
</tr>
<tr>
<td>klavier</td>
<td>serbien</td>
<td>olympischen</td>
<td>kinder</td>
<td>konig</td>
<td>ende</td>
</tr>
<tr>
<td>olympische</td>
<td>land</td>
<td>ende</td>
<td>serbien</td>
<td>olympischen</td>
<td>konig</td>
</tr>
<tr>
<td>2</td>
<td>land</td>
<td>ende</td>
<td>serbien</td>
<td>olympischen</td>
<td>konig</td>
</tr>
<tr>
<td>klavier</td>
<td>land</td>
<td>ende</td>
<td>serbien</td>
<td>olympischen</td>
<td>konig</td>
</tr>
<tr>
<td>olympische</td>
<td>land</td>
<td>ende</td>
<td>serbien</td>
<td>olympischen</td>
<td>konig</td>
</tr>
<tr>
<td>2</td>
<td>land</td>
<td>ende</td>
<td>serbien</td>
<td>olympischen</td>
<td>konig</td>
</tr>
</tbody>
</table>
Which large-scale implementation is right for me?

- **Yahoo LDA** [Smola and Narayanamurthy 2010]
  - Fastest
  - Sparse Gibbs sampling
  - Great when you can use memcached

- **Mahout**
  - Variational
  - Simplest

- **Mr LDA**
  - Designed for extensibility
  - Multilingual
  - Hyperparameter updating [Wallach et al. 2009]
  - Likelihood monitoring
Conclusion

- **Mr. LDA**: A scalable implementation for topic modeling
- Extensible variational inference
- Next steps
  - Supporting more modeling assumptions (including non-conjugacy)
  - Nonparametrics (over topics and vocabulary)
  - Multiple starts

**Download the Code**

http://mrllda.cc
Ke Zhai

- First author
- Immigration issues prevented presentation

MR LDA

- MR = MapReduce
- LDA = latent Dirichlet allocation
Ke Zhai

- First author
- Immigration issues prevented presentation

**MR LDA**

- MR = MapReduce
- LDA = latent Dirichlet allocation
- MR LDA = Ke
Merci!

- Jimmy Lin
- NSF #1018625


Map($d, \vec{w}$)
1: repeat
2: for all $v \in [1, V]$ do
3:   for all $k \in [1, K]$ do
4:     Update $\phi_{v,k} = \beta_{v,k} \times \exp(\Psi(\gamma_{d,k}))$.
5:   end for
6:   Normalize row $\phi_{v,*}$, such that $\sum_{k=1}^{K} \phi_{v,k} = 1$.
7:   Update $\sigma = \sigma + \vec{w}_v \phi_v$, where $\phi_v$ is a $K$-dimensional vector, and $\vec{w}_v$ is the count of $v$ in this document.
8: end for
9: Update row vector $\gamma_{d,*} = \alpha + \sigma$.
10: until convergence
11: for all $k \in [1, K]$ do
12:   for all $v \in [1, V]$ do
13:     Emit key-value pair $\langle k, \Delta \rangle : \vec{w}_v \phi_v$.
14:     Emit key-value pair $\langle k, \nu \rangle : \vec{w}_v \phi_v$. {order inversion}
15:   end for
16: Emit key-value pair $\langle \Delta, k \rangle : (\Psi(\gamma_{d,k}) - \Psi\left(\sum_{l=1}^{K} \gamma_{d,l}\right))$.
17: {emit the $\gamma$-tokens for $\alpha$ update}
18: Output key-value pair $\langle k, d \rangle - \gamma_{d,k}$ to file.
19: end for
20: Emit key-value pair $\langle \Delta, \Delta \rangle - \mathcal{L}$, where $\mathcal{L}$ is log-likelihood of this document.
Map($d, \tilde{w}$)
1: repeat
2: for all $v \in [1, V]$ do
3: for all $k \in [1, K]$ do
4: Update $\phi_{v,k} = \beta_{v,k} \times \exp(\Psi(\gamma_{d,k}))$.
5: end for
6: Normalize row $\phi_{v,*}$, such that $\sum_{k=1}^{K} \phi_{v,k} = 1$.
7: Update $\sigma = \sigma + \tilde{w}_v \phi_v$, where $\phi_v$ is a $K$-dimensional vector, and $\tilde{w}_v$ is the count of $v$ in this document.
8: end for
9: Update row vector $\gamma_{d,*} = \alpha + \sigma$.
10: until convergence
11: for all $k \in [1, K]$ do
12: for all $v \in [1, V]$ do
13: Emit key-value pair $\langle k, \triangle \rangle : \tilde{w}_v \phi_v$.
14: Emit key-value pair $\langle k, v \rangle : \tilde{w}_v \phi_v$. {order inversion}
15: end for
16: Emit key-value pair $\langle \triangle, k \rangle : (\Psi(\gamma_{d,k}) - \Psi(\sum_{l=1}^{K} \gamma_{d,l}))$.
   {emit the $\gamma$-tokens for $\alpha$ update}
17: Output key-value pair $\langle k, d \rangle - \gamma_{d,k}$ to file.
18: end for
19: Emit key-value pair $\langle \triangle, \triangle \rangle - L$, where $L$ is log-likelihood of this document.
Map($d, \vec{w}$)
1: repeat
2: for all $v \in [1, V]$ do
3: for all $k \in [1, K]$ do
4: Update $\phi_{v,k} = \beta_{v,k} \times \exp(\Psi(\gamma_{d,k}))$.
5: end for
6: Normalize row $\phi_{v,*}$, such that $\sum_{k=1}^{K} \phi_{v,k} = 1$.
7: Update $\sigma = \sigma + \vec{w}_v \phi_v$, where $\phi_v$ is a $K$-dimensional vector, and $\vec{w}_v$ is the count of $v$ in this document.
8: end for
9: Update row vector $\gamma_{d,*} = \alpha + \sigma$.
10: until convergence
11: for all $k \in [1, K]$ do
12: for all $v \in [1, V]$ do
13: Emit key-value pair $\langle k, \triangle \rangle : \vec{w}_v \phi_v$.
14: Emit key-value pair $\langle k, v \rangle : \vec{w}_v \phi_v$. {order inversion}
15: end for
16: Emit key-value pair $\langle \triangle, k \rangle : (\Psi(\gamma_{d,k}) - \Psi(\sum_{l=1}^{K} \gamma_{d,l}))$.
{emit the $\gamma$-tokens for $\alpha$ update}
17: Output key-value pair $\langle k, d \rangle - \gamma_{d,k}$ to file.
18: end for
19: Emit key-value pair $\langle \triangle, \triangle \rangle - \mathcal{L}$, where $\mathcal{L}$ is log-likelihood of this document.
Input:
KEY - key pair \( \langle p_{\text{left}}, p_{\text{right}} \rangle \).
VALUE - an iterator \( I \) over sequence of values.

Reduce
1: Compute the sum \( \sigma \) over all values in the sequence \( I \).
2: if \( p_{\text{left}} = \triangle \) then
3: \hspace{1em} if \( p_{\text{right}} = \triangle \) then
4: \hspace{2em} Output key-value pair \( \langle \triangle, \triangle \rangle \) \( - \sigma \) to file.
5: \hspace{2em} \{output the model likelihood \( L \) for convergence checking\}
6: \hspace{1em} else
7: \hspace{2em} Output key-value pair \( \langle \triangle, p_{\text{right}} \rangle \) \( - \sigma \) to file.
8: \hspace{2em} \{output the \( \gamma \)-tokens to update \( \alpha \)-vectors, Section ??\}
9: \hspace{1em} end if
10: else
11: \hspace{2em} if \( p_{\text{right}} = \triangle \) then
12: \hspace{3em} Update the normalization factor \( n = \sigma \). \{order inversion\}
13: \hspace{2em} else
14: \hspace{3em} Output key-value pair \( \langle k, v \rangle : \frac{\sigma}{n} \). \{output normalized \( \beta \) value\}
15: \hspace{2em} end if
16: end if
Input:
KEY - key pair \( \langle p_{\text{left}}, p_{\text{right}} \rangle \).
VALUE - an iterator \( \mathcal{I} \) over sequence of values.

Reduce
1: Compute the sum \( \sigma \) over all values in the sequence \( \mathcal{I} \).
2: if \( p_{\text{left}} = \triangle \) then
3:   if \( p_{\text{right}} = \triangle \) then
4:     Output key-value pair \( \langle \triangle, \triangle \rangle - \sigma \) to file.
        \{output the model likelihood \( \mathcal{L} \) for convergence checking\}
5:   else
6:     Output key-value pair \( \langle \triangle, p_{\text{right}} \rangle - \sigma \) to file.
        \{output the \( \gamma \)-tokens to update \( \alpha \)-vectors, Section ??\}
7: end if
8: else
9:   if \( p_{\text{right}} = \triangle \) then
10:      Update the normalization factor \( n = \sigma \). \{order inversion\}
11:   else
12:      Output key-value pair \( \langle k, v \rangle : \frac{\sigma}{n} \). \{output normalized \( \beta \) value\}
13: end if
14: end if
**Input:**

**KEY** - key pair \(<p_{\text{left}}, p_{\text{right}}>\).

**VALUE** - an iterator \(\mathcal{I}\) over sequence of values.

**Reduce**

1. Compute the sum \(\sigma\) over all values in the sequence \(\mathcal{I}\).
2.  
   if \(p_{\text{left}} = \triangle\) then
3.      
   if \(p_{\text{right}} = \triangle\) then
4.         Output key-value pair \(<\triangle, \triangle> - \sigma\) to file.
   
   \{output the model likelihood \(\mathcal{L}\) for convergence checking\}
5.      else
6.         Output key-value pair \(<\triangle, p_{\text{right}}> - \sigma\) to file.
   
   \{output the \(\gamma\)-tokens to update \(\alpha\)-vectors, Section ??\}
7.    end if
8.  else
9.      if \(p_{\text{right}} = \triangle\) then
10.         Update the normalization factor \(n = \sigma\). \{order inversion\}
11.    else
12.      Output key-value pair \(<k, v> : \frac{\sigma}{n}\). \{output normalized \(\beta\) value\}
13.    end if
14. end if