Distributed Machine Learning - Challenges and Approaches

Parameswaran Raman
(Advisor: S.V.N. Vishwanathan)

Department of Computer Science
University of California, Santa Cruz

May 15, 2018
Outline

1. Motivation
   - Distributed Machine Learning

2. Multinomial Logistic Regression
   - Background
   - DS-MLR
   - Experiments

3. Inference for Mixture Models
   - Background
   - Existing inference methods
   - Extreme Stochastic Variational Inference (ESVI)
Motivation
Multinomial Logistic Regression
Inference for Mixture Models
References

Distributed Machine Learning

MACHINE LEARNING

MACHINE LEARNING EVERYWHERE!
Motivation
Multinomial Logistic Regression
Inference for Mixture Models
References

Distributed Machine Learning
Applications

- **Video**: Netflix, Amazon Instant Video
- **Voice**: Alexa, Google Home
- **Images**: fb tagging, amazon visual product search
- **Text**: google translate, summarization, etc
- **Social Good**: medicine, farming, etc
"I think AI is akin to building a **rocket ship**. You need a **huge engine** and a **lot of fuel**. If you have a large engine and a tiny amount of fuel, you won’t make it to orbit. If you have a tiny engine and a ton of fuel, you can’t even lift off. The analogy to deep learning is that the rocket engine is the **deep learning models** and the fuel is the **huge amounts of data** we can feed to these algorithms."

- Andrew Ng
Why Distributed Machine Learning?

We want to

- Consume **massive amount of data**
- Learn **sophisticated models**

Traditionally, **only one** of these aspects has been tackled at a time. Later in this talk, we will see how to achieve both **simultaneously** (focus of my research)
Why Distributed Machine Learning?

We want to

- Consume **massive amount of data**
- Learn **sophisticated models**

Traditionally, **only one** of these aspects has been tackled at a time. Later in this talk, we will see how to achieve both **simultaneously** (focus of my research)
Why Distributed Machine Learning?

Some are easier than the others!

- Logistic Regression (binary, multi-class)
- Support Vector Machines
- k-Nearest Neighbors
- Decision Trees (GBDT, etc)
- Matrix Factorization
- Deep Neural Networks (ResNet, CNNs, RNNs, LSTMs)
- Latent Variable Models (GMM, LDA, SBM, etc)
- Hierarchical Bayesian Models
Why Distributed Machine Learning?

massive amount of data
Why Distributed Machine Learning?

massive amount of data

\[
\min_w \sum_{i=1}^N f_i(w, x_i)
\]

\[
w = w - \eta \sum_{i=1}^N \nabla_w f_i(w, x_i)
\]
massive amount of data

$$\min_w \sum_{i=1}^{N} f_i(w, x_i)$$

$$w = w - \eta \sum_{i=1}^{N} \nabla_w f_i(w, x_i)$$
Why Distributed Machine Learning?

massive amount of data

\[
\min_w \sum_{i=1}^N f_i(w, x_i)
\]

\[
w = w - \eta \sum_{i=1}^N \nabla_w f_i(w, x_i)
\]

- Both \( f_i \) and gradients \( \nabla_w f_i(\cdot) \) can be computed in parallel
- **How to distribute computation?** Data \( x_1...N \) is distributed among workers, Model \( w \) is replicated on each worker
Why Distributed Machine Learning?

massive amount of data

Data Parallelism!
Why Distributed Machine Learning?

massive amount of data

Data Parallelism!
Why Distributed Machine Learning?

massive amount of data

Popular Related Work in this direction:

Figure: Parameter Server

Figure: HogWild

Scaling Distributed Machine Learning with the Parameter Server

Mu Li∗, David G. Andersen∗, Jun Woo Park∗, Alexander J. Smola∗†, Amr Ahmed†, Vanja Josifovski†, James Long†, Eugene J. Shekita†, Bor-Yiing Su†

∗Carnegie Mellon University  †Baidu  ‡Google

{muli, dga, jurenwoop}@cs.cmu.edu, alex@smola.org, {amnn, vanjaj, jamlong, shekitja, boryiingsu}@google.com

HOGWILD!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent

Feng Niu leonn@cs.wisc.edu
Benjamin Recht brecht@cs.wisc.edu
Christopher Ré chrisre@cs.wisc.edu

Stephen J. Wright
swright@cs.wisc.edu
Computer Sciences Department
University of Wisconsin-Madison
Madison, WI 53706
massive amount of data

Popular Related Work in this direction:

Communication-Efficient
Distributed Dual Coordinate Ascent

Adding vs. Averaging in Distributed Primal-Dual Optimization

Chenxin Ma*
Industrial and Systems Engineering, Lehigh University, USA

Virginia Smith*
University of California, Berkeley, USA

Martin Jaggi
ETH Zürich, Switzerland

Michael I. Jordan
University of California, Berkeley, USA

Peter Richtárik
School of Mathematics, University of Edinburgh, UK

*Authors contributed equally.
Why Distributed Machine Learning?

sophisticated models
Why Distributed Machine Learning?

sophisticated models (large number of parameters)
Why Distributed Machine Learning?

**sophisticated models** (large number of parameters)

e.g. Multinomial Logistic Regression

\[
\min_{w_1, \ldots, w_K} \frac{\lambda}{2} \sum_{k=1}^{K} \| w_k \|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i + \frac{1}{N} \sum_{i=1}^{N} \log \left( \sum_{k=1}^{K} \exp(w_k^T x_i) \right)
\]

- log-partition function - couples \( w_k \) parameters
- sum over \( K \) can be parallelized once the \( \log(\cdot) \) has been linearized

- Both function and gradients w.r.t \( w_k \) (\( k = 1, \ldots, K \)) can be computed in parallel

- **How to distribute computation?** Model \( w_k \) (\( k = 1, \ldots, K \)) is distributed across workers, Data \( x_1 \ldots N \) is replicated on each worker
**Why Distributed Machine Learning?**

**sophisticated models** (large number of parameters)

e.g. Multinomial Logistic Regression

\[
\min_{w_1, \ldots, w_K} \frac{\lambda}{2} \sum_{k=1}^{K} \|w_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i + \frac{1}{N} \sum_{i=1}^{N} \log \left( \sum_{k=1}^{K} \exp(w_k^T x_i) \right)
\]

- log-partition function - couples \(w_k\) parameters
- sum over \(K\) can be parallelized once the \(\log(\cdot)\) has been linearized
- Both function and gradients w.r.t \(w_k\) \((k = 1, \ldots, K)\) can be computed in parallel

**How to distribute computation?** Model \(w_k\) \((k = 1, \ldots, K)\) is distributed across workers, Data \(x_1 \ldots N\) is replicated on each worker
Why Distributed Machine Learning?

**sophisticated models** (large number of parameters)

- Model is partitioned and trained in parallel
Why Distributed Machine Learning?

**sophisticated models** (large number of parameters)

- Model is partitioned and trained in parallel

**Model Parallelism!**
Why Distributed Machine Learning?

sophisticated models

Popular Related Work in this direction:

Distributed training of Large-scale Logistic models

Siddharth Gopal  
5000 Forbes Ave, Carnegie Mellon University, Pittsburgh PA 15217 USA  
SGOPAL@CS.CMU.EDU

Yiming Yang  
5000 Forbes Ave, Carnegie Mellon University, Pittsburgh PA 15217 USA  
YIMING@CS.CMU.EDU

Figure: PMLR
However, in reality *data* and *parameters* can be too ↑ to fit on single-machine
Need to scale both vertically (model parallel) as well as horizontally (data parallel).

Need to distribute both the (parameters) as well as (data).
Big Picture

Need to scale both vertically (model parallel) as well as horizontally (data parallel).

Need to distribute both the (parameters) as well as (data).
Another aspect to consider in Distributed ML

Formulation of Model

Optimization

Computation

Communication

Network Topology
Big Picture

Need to scale both vertically (model parallel) as well as horizontally (data parallel).

Need to distribute both the (parameters) as well as (data).

Need to perform communication in the background while performing computation (asynchronous!)
Some are easier than the others

- Optimization problem directly exhibits separability
e.g. Matrix Factorization

\[ \mathcal{L}(W, H) = \sum_{i=1}^{N} \sum_{j=1}^{M} (A_{ij} - \langle w_i, h_j \rangle) \]
Definition

Let \( \{S_i\}_{i=1}^m \) and \( \{S'_j\}_{j=1}^{m'} \) be two families of sets of parameters. A function \( f : \prod_{i=1}^m S_i \times \prod_{j=1}^{m'} S'_j \rightarrow \mathbb{R} \) is doubly separable if \( \exists \) \( f_{ij} : S_i \times S'_j \rightarrow \mathbb{R} \) for each \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, m' \) such that:

\[
f(\theta_1, \theta_2, \ldots, \theta_m, \theta'_1, \theta'_2, \ldots, \theta'_{m'}) = \sum_{i=1}^{m} \sum_{j=1}^{m'} f_{ij}(\theta_i, \theta'_j)
\]

In the context of Matrix Factorization,

\[
\mathcal{L}(w_1, w_2, \ldots, w_N, h_1, h_2, \ldots, h_M) = \sum_{i=1}^{N} \sum_{j=1}^{M} f(w_i, h_j)
\]
Some are easier than the others

- Optimization problem directly exhibits separability
e.g. Matrix Factorization

- Optimization problem may require reformulation
e.g. Multinomial Logistic Regression, Learning to Rank, Variational Inference for Mix Models
Focus of my research

Data and Model Parallelism

How can we apply this to more diverse models where the structure to exploit is not immediately obvious?

1. Multinomial Logistic Regression
2. Inference for Mixture Models
3. Ranking and Latent Collaborative Retrieval
Outline

1. Motivation
   - Distributed Machine Learning

2. Multinomial Logistic Regression
   - Background
   - DS-MLR
   - Experiments

3. Inference for Mixture Models
   - Background
   - Existing inference methods
   - Extreme Stochastic Variational Inference (ESVI)
Formal Definition

- $N$ data points $(x_i, y_i)$ where $x_i \in \mathbb{R}^d$, $y_i \in \{1, 2, \ldots, K\}$
- $y_{ik} = I(y_i = k)$ denotes the membership of $x_i$ to class $k$
- Probability that $x$ belongs to class $k$ is given by:

$$\mathbb{P}(y = k|x) = \frac{\exp(w_k^T x)}{\sum_{j=1}^{K} \exp(w_j^T x)}$$

where $W = \{w_1, w_2, \ldots, w_K\}$ denotes the parameter vector for each of the $K$ classes.
Objective function of MLR

\[ \mathcal{L}_1 = \frac{\lambda}{2} \sum_{k=1}^{K} \| w_k \|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i + \frac{1}{N} \sum_{i=1}^{N} \log \left( \sum_{k=1}^{K} \exp(w_k^T x_i) \right) \]

Scaling Challenges:
- log-partition function - couples \( w_k \) parameters
- need to sum over large number of classes
Reformulating the objective of MLR

Log-concavity bound

\[ \log(\gamma) \leq a \cdot \gamma - \log(a) - 1, \quad \forall \gamma, a > 0, \]

where \( a \) is a variational parameter. This bound is tight when \( a = \frac{1}{\gamma} \).

[1] use this to linearize the objective function and make it model parallel
Reformulating the objective of MLR

\[ L_2 = \frac{\lambda}{2} \sum_{k=1}^{K} \| w_k \|^2 + \frac{1}{N} \sum_{i=1}^{N} \left( - \sum_{k=1}^{K} y_{ik} w_k^T x_i + a_i \sum_{k=1}^{K} \exp(w_k^T x_i) - \log(a_i) - 1 \right) \]

where \( a_i \) can be computed in closed form as:

\[ a_i = \frac{1}{\sum_{k=1}^{K} \exp(w_k^T x_i)} \]
Reformulating the objective of MLR

Solve $K$ sub-problems

$$
\arg\min_{w_k} \frac{\lambda}{2} \|w_k\|^2 - \frac{1}{N} \left( \sum_{i=1}^{N} y_{ik} w_k^T x_i + \sum_{i=1}^{N} a_i \exp(w_k^T x_i) \right).
$$

Issues:
- Each sub-problem involves $\sum_{i=1}^{N}$
- Often as data grows, $N$ and $K$ grow hand in hand
- $a_i$ is updated only after all sub-problems are solved
We go one step further..

Modify the objective to exploit data parallelism as well.
Outline

1. Motivation
   - Distributed Machine Learning

2. Multinomial Logistic Regression
   - Background
   - DS-MLR
   - Experiments

3. Inference for Mixture Models
   - Background
   - Existing inference methods
   - Extreme Stochastic Variational Inference (ESVI)
Double-Separability

**Definition**

Let \( \{S_i\}_{i=1}^m \) and \( \{S'_j\}_{j=1}^{m'} \) be two families of sets of parameters. A function \( f : \prod_{i=1}^m S_i \times \prod_{j=1}^{m'} S'_j \to \mathbb{R} \) is doubly separable if \( \exists f_{ij} : S_i \times S'_j \to \mathbb{R} \) for each \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, m' \) such that:

\[
f(\theta_1, \theta_2, \ldots, \theta_m, \theta'_1, \theta'_2, \ldots, \theta'_{m'}) = \sum_{i=1}^m \sum_{j=1}^{m'} f_{ij}(\theta_i, \theta'_j)
\]

In the context of MLR,

\[
\mathcal{L}(a_1, a_2, \ldots, a_N, w_1, w_2, \ldots, w_K) = \sum_{i=1}^N \sum_{k=1}^K f(a_i, w_k)
\]
Double-Separability

\[ \mathcal{L}(a_1, a_2, \ldots, a_N, w_1, w_2, \ldots, w_K) = \sum_{i=1}^{N} \sum_{k=1}^{K} f(a_i, w_k) \]
Our DS-MLR formulation

\[ L(a, W) = \sum_{i=1}^{N} \sum_{k=1}^{K} \left( \frac{\lambda}{2N} \| w_k \|^2 - \frac{y_{ik} w_k^T x_i}{N} + \frac{\exp(w_k^T x_i + \log a_i)}{N} - \frac{\log a_i}{KN} - \frac{1}{KN} \right) \]
Stochastic Optimization

\[
\begin{align*}
    w_k &\leftarrow w_k - \eta_1 K \left( \lambda w_k - y_{ik} x_i + \exp(w_k^T x_i + \log a_i) x_i \right) \\
    \log a_i &\leftarrow \log a_i - \eta_2 K \left( \exp(w_k^T x_i + \log a_i) - \frac{1}{K} \right)
\end{align*}
\]

- Each term in the updates depends on only one \( x_i \) and \( w_k \)
- \( a_i \) and \( w_k \) can now be updated simultaneously
Each iteration consists of $P$ inner-epochs

During each inner-epoch each processor makes updates on its local block
Algorithm 1 DS-MLR Synchronous

1. $K$: # classes, $P$: # workers, $T$: total outer iterations, $t$: outer iteration index, $s$: inner epoch index
2. $W^{(p)}$: weights per worker, $a^{(p)}$: variational parameters per worker
3. Initialize $W^{(p)} = 0$, $a^{(p)} = \frac{1}{K}$
4. for all $t = 1, 2, \ldots, T$ do
   5. for all $s = 1, 2, \ldots, P$ do
      6. Send $W^{(p)}$ to worker on the right
      7. Receive $W^{(p)}$ from worker on the left
      8. Update $W^{(p)}$ stochastically
      9. Update $a^{(p)}$ stochastically
   10. end for
11. end for
Algorithm - DS-MLR Asynchronous

(a) Initial assignment of $W$ and $X$. Each worker works only on the diagonal active area in the beginning.

(b) After a worker finishes processing column $k$, it sends the corresponding item parameter $w_k$ to another worker. Here, $w_2$ is sent from worker 1 to 4.

(c) Upon receipt, the column is processed by the new worker. Here, worker 4 can now process column 2 since it owns the column.

(d) During the execution of the algorithm, the ownership of the global parameters (weight vectors) $w_k$ changes.
Outline

1. Motivation
   - Distributed Machine Learning

2. Multinomial Logistic Regression
   - Background
   - DS-MLR
   - Experiments

3. Inference for Mixture Models
   - Background
   - Existing inference methods
   - Extreme Stochastic Variational Inference (ESVI)
### Datasets

**Table:** Description of Datasets used

<table>
<thead>
<tr>
<th>Dataset</th>
<th># instances</th>
<th># features</th>
<th>#classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEF</td>
<td>10,000</td>
<td>80</td>
<td>63</td>
</tr>
<tr>
<td>NEWS20</td>
<td>11,260</td>
<td>53,975</td>
<td>20</td>
</tr>
<tr>
<td>LSHTC1-small</td>
<td>4,463</td>
<td>51,033</td>
<td>1,139</td>
</tr>
<tr>
<td>LSHTC1-large</td>
<td>93,805</td>
<td>347,256</td>
<td>12,294</td>
</tr>
<tr>
<td>WikipediaLarge</td>
<td>2,365,436</td>
<td>20,000</td>
<td>325,056</td>
</tr>
<tr>
<td>ODP</td>
<td>1,084,404</td>
<td>422,712</td>
<td>105,034</td>
</tr>
<tr>
<td>Youtube8M-Video</td>
<td>4,902,565</td>
<td>1,152</td>
<td>4,716</td>
</tr>
<tr>
<td>Reddit-Small</td>
<td>52,883,089</td>
<td>1,348,182</td>
<td>33,225</td>
</tr>
<tr>
<td>Reddit-Full</td>
<td>211,532,359</td>
<td>1,348,182</td>
<td>33,225</td>
</tr>
</tbody>
</table>
Data and Model Fit in Memory

- NEWS20 dataset: $P=1\times1\times1$, $\lambda = 8.881e-05$, $\eta = 1e4$
- CLEF dataset: $P=1\times1\times1$, $\lambda = 0.001$, $\eta = 1e2$
- LSHTC1-small dataset: $P=1\times1\times1$, $\lambda = 2.2406e-07$, $\eta = 1e4$
Data Fits and Model does not Fit
Motivation
Multinomial Logistic Regression
Inference for Mixture Models
References

Data does not Fit and Model Fits

Youtube-Video dataset: \( P=40 \times 1 \times 250, \lambda = 1e^{-19}, \eta = 4e10 \)

[Graph showing the objective function over time for DS-MLR]
Data does not fit and model does not fit.
Summary

Figure: Applicability of the popular algorithms to common scenarios in distributed machine learning

- Distributed (both multi-core and multi-machine)
- Asynchronous and Lock-free
For details, please refer the arXiv version [2].
Here, we have the following parameters:

- $\theta_k \sim$ Exp Family (global)
- $z_i \sim$ Multinomial (local)
- $\pi \sim$ Dirichlet (global)
Here, we have the following parameters:
- $\theta_k \sim \text{Exp Family (global)}$
- $z_i \sim \text{Multinomial (local)}$
- $\pi \sim \text{Dirichlet (global)}$
Outline

1. Motivation
   - Distributed Machine Learning

2. Multinomial Logistic Regression
   - Background
   - DS-MLR
   - Experiments

3. Inference for Mixture Models
   - Background
   - Existing inference methods
   - Extreme Stochastic Variational Inference (ESVI)
Recap - Bayesian Inference

Given data $x = \{x_1, x_2, \ldots, x_n\}$,

$$p(\theta|x) = \frac{\underbrace{p(x|\theta) \cdot p(\theta)}_{\text{likelihood prior}}}{\underbrace{\int p(x, \theta) d\theta}_{\text{marginal likelihood (model evidence)}}}$$

Modeling Assumptions:
- $p(x|\theta) \sim \exp(\langle \phi(x), \theta \rangle) - g(\theta))$
- $x$ are iid

Most inference problems will be one of:
- **Marginalization** $p(x) = \int p(x, \theta) d\theta$
- **Expectation** $\mathbb{E}[f(x|z)] = \int f(x)p(x|z)dz$
- **Prediction** $p(y|x) = \int p(y|\theta, x)p(\theta|x)d\theta$
Variational Inference

The joint distribution of the data and latent variables can be written as:

\[
p(x, \pi, z, \theta | \alpha, n, \nu) = p(\pi | \alpha) \cdot \prod_{k=1}^{K} p(\theta_k | n_k, \nu_k) \cdot \prod_{i=1}^{N} p(z_i | \pi) \cdot p(x_i | z_i, \theta)
\]  

(1)

Variational inference approximates this distribution with a fully, factorized distribution of the following form:

\[
q\left(\pi, z, \theta | \tilde{\pi}, \tilde{z}, \tilde{\theta}\right) = q(\pi | \tilde{\pi}) \cdot \prod_{i=1}^{N} q(z_i | \tilde{z}_i) \cdot \prod_{k=1}^{K} q(\theta_k | \tilde{\theta}_k).
\]  

(2)
Variational Inference

- Introduce $q$ (factorizable) that approximates $p$
- Maximize Evidence Lower Bound (ELBO):

$$
\mathcal{L} \left( \tilde{\pi}, \tilde{Z}, \tilde{\theta} \right) = \mathbb{E}_q \left[ \log p \left( x, \pi, z, \theta \right) \right] - \mathbb{E}_q \left[ \log q \left( \pi, z, \theta | \tilde{\pi}, \tilde{Z}, \tilde{\theta} \right) \right]
$$
Neither **Data parallel** nor **Model parallel**

![Diagram](image)

**Figure**: Access pattern of variables during Variational Inference (VI) updates. Green indicates that the variable or data point is being read, while red indicates that the variable is being updated.
Stochastic Variational Inference (SVI)

Data parallel but not Model parallel

Figure: Access pattern of variables during Variational Inference (SVI) updates. Green indicates that the variable or data point is being read, while red indicates that the variable is being updated.
How to go further?

Can we achieve **both data and model parallelism** in this problem?
Outline

1. Motivation
   - Distributed Machine Learning

2. Multinomial Logistic Regression
   - Background
   - DS-MLR
   - Experiments

3. Inference for Mixture Models
   - Background
   - Existing inference methods
   - Extreme Stochastic Variational Inference (ESVI)
**Extreme Stochastic Variational Inference (ESVI)**

(a) \( \tilde{\pi} \) update

(b) \( \tilde{\theta} \) update

(c) \( \tilde{z} \) update

**Figure:** Access pattern of variables during ESVI updates. Green indicates that the variable or data point is being read, while red indicates that the variable is being updated.
Both **Data parallel** and **Model parallel**

**Figure**: Access pattern of variables during ESVI updates. Green indicates that the variable or data point is being read, while red indicates that the variable is being updated.
Experiments

Datasets:

<table>
<thead>
<tr>
<th>Dataset</th>
<th># documents</th>
<th># vocabulary</th>
<th># words</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIPS</td>
<td>1,312</td>
<td>12,149</td>
<td>1,658,309</td>
</tr>
<tr>
<td>Enron</td>
<td>37,861</td>
<td>28,102</td>
<td>6,238,796</td>
</tr>
<tr>
<td>Ny Times</td>
<td>298,000</td>
<td>102,660</td>
<td>98,793,316</td>
</tr>
<tr>
<td>PubMed</td>
<td>8,200,000</td>
<td>141,043</td>
<td>737,869,083</td>
</tr>
<tr>
<td>UMBC-3B</td>
<td>40,599,164</td>
<td>3,431,260</td>
<td>3,013,004,127</td>
</tr>
</tbody>
</table>

Table: Data Characteristics

We apply ESVI to GMM and LDA in our experiments.
Experiments

ESVI-GMM (Single-Machine)
Experiments

ESVI-GMM (Multi-Machine)
Experiments

ESVI-LDA (Single-Machine) - Varying K, Fixed P
Experiments

ESVI-LDA (Single-Machine) - Varying P, Fixed K
Experiments

ESVI-LDA (Multi-Machine) - PubMed, UMBC

- PubMed dataset: $P=32 \times 16 \ K=128$
- UMBC-3B dataset: $P=32 \times 16 \ K=128$
Experiments

ESVI-LDA Predictive Performance

![Enron dataset graph](image1)

![NY Times dataset graph](image2)
Summary

Figure: Applicability of the three algorithms to common scenarios in distributed machine learning

- Distributed (both multi-core and multi-machine)
- Asynchronous and Lock-free
For details, please refer the arXiv version [3].
