Distributed Training for Large-scale Logistic Models

Siddharth Gopal

Carnegie Mellon University

21 Aug 2013

\(^1\) Joint work with Yiming Yang presented at ICML’13
Outline of the Talk

- Logistic Models
- Maximum Likelihood Estimation
- Parallelization
- Experiments
Logistic Models model probability of an outcome $Y$ given a predictor $x$.

$$P(Y = y | x; \mathbf{w}) \propto \exp(\mathbf{w}^\top \phi(y, x))$$


For example, in Multinomial Logistic Regression

$$P(Y = k | x; \mathbf{w}) = \frac{\exp(w_k^\top x)}{\sum_j \exp(w_j^\top x)}$$
Train Logistic models on large-scale data.

What is Large-scale?
- Large number of Training Examples
- High dimensionality
- Large number of Outcomes
Focus of the Talk

Train Logistic models on large-scale data.

What is Large-scale?
- Large number of Training Examples
- **High dimensionality**
- Large number of Outcomes
### Motivation

Some commonly used data on the web,

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Instances</th>
<th>#Labels</th>
<th>#Features</th>
<th>#Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>ODP subset</td>
<td>93,805</td>
<td>12,294</td>
<td>347,256</td>
<td>4,269,165,264</td>
</tr>
<tr>
<td>Wikipedia subset</td>
<td>2,365,436</td>
<td>325,056</td>
<td>1,617,899</td>
<td>525,907,777,344</td>
</tr>
<tr>
<td>Image-net</td>
<td>14,197,122</td>
<td>21,841</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Motivation

Some commonly used data on the web,

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Instances</th>
<th>#Labels</th>
<th>#Features</th>
<th>#Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>ODP subset</td>
<td>93,805</td>
<td>12,294</td>
<td>347,256</td>
<td>4,269,165,264</td>
</tr>
<tr>
<td>Wikipedia subset</td>
<td>2,365,436</td>
<td>325,056</td>
<td>1,617,899</td>
<td>525,907,777,344</td>
</tr>
<tr>
<td>Image-net</td>
<td>14,197,122</td>
<td>21,841</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

- How can we parallelize the training of such models?
- How can we optimize different subsets of parameters simultaneously?
Maximum Likelihood Estimation (MLE)

Typical MLE estimation

- $N$ training examples, $K$ classes.
- $x_i$ denotes the $i^{th}$ training example.
- Indicator variable $y_{ik}$ denotes whether $x_i$ belongs to class $k$.
- Estimate parameters $\mathbf{w}$ by maximizing the log-likelihood,

$$\max_{\mathbf{w}} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \log P(y_{ik}|x_i; \mathbf{w}) - \frac{\lambda}{2} \| \mathbf{w} \|^2$$
Typical MLE estimation

- $N$ training examples, $K$ classes.
- $x_i$ denotes the $i^{th}$ training example.
- Indicator variable $y_{ik}$ denotes whether $x_i$ belongs to class $k$.
- Estimate parameters $\mathbf{w}$ by maximizing the log-likelihood,

$$
\max_{\mathbf{w}} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \log P(y_{ik}|x_i; \mathbf{w}) - \frac{\lambda}{2} \| \mathbf{w} \|^2
$$

[OPT1] $$\min_{\mathbf{w}} \frac{\lambda}{2} \| \mathbf{w} \|^2 - \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k^\top x_i + \sum_{i=1}^{N} \log \left( \sum_{k=1}^{K} \exp(\mathbf{w}_k^\top x_i) \right)$$
Parallelization

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

The log-sum-exp (LSE) function couples all the class-level parameter \(w_k\)'s together.

Replace LSE by a parallelizable function

This parallelizable function should be an upper-bound

It should not make the optimization harder - like introduce non-convexity, non-differentiability etc.
Parallelization

\[
\min_{\mathbf{w}} \frac{\lambda}{2} \| \mathbf{w} \|^2 - \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k^\top \mathbf{x}_i + \sum_{i=1}^{N} \log \left( \sum_{k=1}^{K} \exp(\mathbf{w}_k^\top \mathbf{x}_i) \right)
\]

- The log-sum-exp (LSE) function couples all the class-level parameter \( w_k \)'s together.

The log-sum-exp (LSE) function couples all the class-level parameter \( w_k \)'s together.
Parallelization

\[
\min_{\mathbf{w}} \frac{\lambda}{2} \| \mathbf{w} \|^2 - \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k^\top \mathbf{x}_i + \sum_{i=1}^{N} \log \left( \sum_{k=1}^{K} \exp(\mathbf{w}_k^\top \mathbf{x}_i) \right)
\]

- The log-sum-exp (LSE) function couples all the class-level parameter \( w_k \)'s together.
- Replace LSE by a parallelizable function
  - This parallelizable function should be an upper-bound
  - It should not make the optimization harder - like introduce non-convexity, non-differentiability etc.

Siddharth Gopal
Distributed Training for Large-scale Logistic Models
Properties used

- LSE is a convex-function
- Convex function can be approximated to any precision by piecewise linear functions.

\[
\max_j \{ a_j^\top \gamma + b_j \} \leq \log \left( \sum_{k=1}^K \exp(\gamma_k) \right) \leq \max_{j'} \{ c_{j'}^\top \gamma + d_{j'} \}
\]

\[a, c \in \mathbb{R}^K \quad b, d \in \mathbb{R}\]
Bound 1 - Piecewise Linear Bound (Hsiung et al)

\[
\max_j \{a_j^\top \gamma + b_j\} \leq \log \left( \sum_{k=1}^{K} \exp(\gamma_k) \right) \leq \max_{j'} \{c_{j'}^\top \gamma + d_{j'}\}
\]

\[a, c \in \mathcal{R}^K \quad b, d \in \mathcal{R}\]

Advantages
- The bound can be made arbitrarily accurate by increasing the number of pieces.

Disadvantages
- Max-function makes the objective non-differentiable.
- The number of variational parameters grows with the approximation level.
- Optimizing the variational parameter is hard.
The LSE is bound by,

$$\log \left( \sum_{k=1}^{K} \exp(w_k^T x_i) \right) \leq a_i + \sum_{k=1}^{K} \log(1 + \exp(w_k^T x_i - a_i)) \quad , \quad a_i \in \mathcal{R}$$

Advantages

- The bound is parallelizable.
- It is an upper bound.
- It is differentiable and \textit{convex}. 
Disadvantage

- The bound is not tight enough.

The gap between true objective and upper-bounded objective on the 20-newsgroup dataset.
A relatively famous bound using the concavity of the log-function

\[ \log(x) \leq ax - \log(a) - 1 \quad \forall \ x, \ a > 0 \]
Applying to the LSE function,

\[
\log \left( \sum_{k=1}^{K} \exp(w_k^\top x_i) \right) \leq a_i \sum_{k=1}^{K} \exp(w_k^\top x_i) - \log(a_i) - 1
\]

Advantages

- The bound is parallelizable.
- It is differentiable.
- Optimizing the variational parameter \( a_i \) is easy.
- The upper bound is exact at \( a_i = \frac{1}{\sum_{k=1}^{K} \exp(w_k^\top x_i)} \).

Disadvantage

- The combined objective is non-convex.
MLE estimation  \[ \min_w \frac{\lambda}{2} \|w\|^2 - \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i + \sum_{i=1}^{N} \log \left( \sum_{k=1}^{K} \exp(w_k^T x_i) \right) \]

Log-concavity Bound  \[ \log \left( \sum_{k=1}^{K} \exp(w_k^T x_i) \right) \leq a_i \sum_{k=1}^{K} \exp(w_k^T x_i) - \log(a_i) - 1 \]
Reaching Optimality

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

**Log-concavity Bound**
\[
\log \left( \sum_{k=1}^K \exp(w_k^T x_i) \right) \leq a_i \sum_{k=1}^K \exp(w_k^T x_i) - \log(a_i) - 1
\]

**Combined Objective**
\[
F(W, A) = \frac{\lambda}{2} \sum_{k=1}^K \|w_k\|^2 + \sum_{i=1}^N \left[ - \sum_{k=1}^K y_{ik} w_k^T x_i + a_i \sum_{k=1}^K \exp(w_k^T x_i) - \log(a_i) - 1 \right]
\]
Reaching Optimality

**MLE estimation** \( \min_w \frac{\lambda}{2} \|w\|^2 - \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i + \sum_{i=1}^{N} \log \left( \sum_{k=1}^{K} \exp(w_k^T x_i) \right) \)

**Log-concavity Bound** \( \log \left( \sum_{k=1}^{K} \exp(w_k^T x_i) \right) \leq a_i \sum_{k=1}^{K} \exp(w_k^T x_i) - \log(a_i) - 1 \)

**Combined Objective**

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

Despite the non-convexity, we can show that
- The combined objective has a unique minima.
- This minimum coincides with the optimal MLE solution.
An iterative and parallel block coordinate descent algorithm to converge to the unique minimum.

Algorithm 1 A parallel block coordinate descent

Initialize: \( t \leftarrow 0, A^0 \leftarrow \frac{1}{K}, W^0 \leftarrow 0. \)

While: Not converged

In parallel: \( W^{t+1} \leftarrow \arg \min_W F(W, A^t) \)
\( A^{t+1} \leftarrow \arg \min_A F(W^{t+1}, A) \)
\( t \leftarrow t + 1 \)
Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th># instances</th>
<th>#Leaf-labels</th>
<th>#Features</th>
<th>#Parameters</th>
<th>Parameter Size (approx)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEF</td>
<td>10,000</td>
<td>63</td>
<td>80</td>
<td>5,040</td>
<td>40KB</td>
</tr>
<tr>
<td>NEWS20</td>
<td>11,260</td>
<td>20</td>
<td>53,975</td>
<td>1,079,500</td>
<td>4MB</td>
</tr>
<tr>
<td>LSHTC-small</td>
<td>4,463</td>
<td>1,139</td>
<td>51,033</td>
<td>227,760,279</td>
<td>911MB</td>
</tr>
<tr>
<td>LSHTC-large</td>
<td>93,805</td>
<td>12,294</td>
<td>347,256</td>
<td>4,269,165,264</td>
<td>17GB</td>
</tr>
</tbody>
</table>

Optimization Methods

- Double Majorization Bound (DM)
- Log concavity Bound (LC)
- Limited Memory BFGS (LBFGS) - the most widely used method.
- Alternating Direction Method of Multipliers (ADMM)
Figure: The difference from the true optimum vs time
Discussed multiple ways to perform distributed training of large-scale Logistic Models.

The LC method seem to offer the best trade-off between accuracy and time.

Several open questions,

- Effect of the regularization parameter $\lambda$.
- Effect of the correlation between the parameters.
Binary vs Multiclass

Binary Log-reg
Multiclass Log-reg
Lambda (Regularization parameter)
Accuracy

Siddharth Gopal
Distributed Training for Large-scale Logistic Models