Scaling Multinomial Logistic Regression via Hybrid Parallelism

Parameswaran Raman
University of California Santa Cruz

KDD 2019
Aug 4-8

Joint work with:
Sriram Srinivasan, Shin Matsushima, Xinhua Zhang, Hyokun Yun,
S.V.N. Vishwanathan
Multinomial Logistic Regression (MLR)

Given:

- Training data \((x_i, y_i)_{i=1,...,N}\), 
  \(x_i \in \mathbb{R}^D\)
- Labels \(y_i \in \{1, 2, \ldots, K\}\)

Goal:

- Learn a model \(W\)
- Predict labels for the test data points using \(W\)

Assume:

- \(N\), \(D\) and \(K\) are large (\(N \gg D > K\))

\[
\begin{align*}
  &N \times D \\
  &\text{Data} \\
  &\text{X} \\
  \end{align*}
\]
Given:

- Training data \((x_i, y_i)_{i=1,\ldots,N}\),
  \(x_i \in \mathbb{R}^D\)
- Labels \(y_i \in \{1, 2, \ldots, K\}\)

Goal:

- Learn a model \(W\)
- Predict labels for the test data points using \(W\)
Multinomial Logistic Regression (MLR)

Given:
- Training data \((x_i, y_i)_{i=1,...,N}\), \(x_i \in \mathbb{R}^D\)
- Labels \(y_i \in \{1, 2, \ldots, K\}\)

Goal:
- Learn a model \(W\)
- Predict labels for the test data points using \(W\)

Assume: \(N, D\) and \(K\) are large (\(N >>> D >> K\))
Motivation for Hybrid Parallelism

Popular ways to distribute MLR:

Data parallel (partition data, duplicate parameters)

Data Storage Complexity: $O\left(\frac{ND}{P}\right)$ data, $O(KD)$ model

e.g. L-BFGS
Motivation for Hybrid Parallelism

Popular ways to distribute MLR:

Data parallel (partition data, duplicate parameters)

Model parallel (partition parameters, duplicate data)

Storage Complexity:

$O\left(\frac{ND}{P}\right)$ data, $O(KD)$ model

e.g. L-BFGS

Storage Complexity:

$O(ND)$ data, $O\left(\frac{KD}{P}\right)$ model

e.g. LC [Gopal et al 2013]
Can we get the best of both worlds?
Yes! Hybrid Parallelism

Storage Complexity:
\[ O\left(\frac{ND}{P}\right) \text{ data, } O\left(\frac{KD}{P}\right) \text{ model} \]

We propose a Hybrid Parallel method DS-MLR
Hybrid Parallelism is like a swiss-army knife
Hybrid Parallelism is like a swiss-army knife
Hybrid Parallelism is like a swiss-army knife
Hybrid Parallelism is like a swiss-army knife

- **Data fits**
  - Model does not fit

- **Data fits**
  - Model fits

Size of Model $O(K \times D)$

Size of Data $O(N \times D)$
Hybrid Parallelism is like a swiss-army knife

![Diagram showing the relationship between data size and model size]
Hybrid Parallelism is like a swiss-army knife
Empirical Study - Multi Machine

Reddit-Full dataset (Data Size: 228 GB, Model Size: 358 GB)

Figure: Data does not fit, Model does not fit

- 211 million examples - $O(N)$
- 44 billion parameters - $O(K \times D)$
How do we achieve **Hybrid Parallelism** in machine learning models?
Double-Separability

↓

Hybrid Parallelism
Double-Separability

**Definition**

Let \( \{S_i\}_{i=1}^m \) and \( \{S'_j\}_{j=1}^{m'} \) be two families of sets of parameters. A function \( f : \prod_{i=1}^m S_i \times \prod_{j=1}^{m'} S'_j \rightarrow \mathbb{R} \) 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:

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

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

Each sub-function \( f_{ij} \) can be computed independently and in parallel.
Are all machine learning models doubly-separable?
Sometimes . . .

e.g. Matrix Factorization

\[ L(w_1, w_2, \ldots, w_N, h_1, h_2, \ldots, h_M) = \sum_{i=1}^{N} \sum_{j=1}^{M} f(w_i, h_j) \]

Objective function is trivially doubly-separable!
Others need algorithmic reformulations . . .

- **Ranking** ("RoBiRank: Ranking via Robust Binary Classification", Yun et al 2014)
Others need algorithmic reformulations . . .

- **Ranking** ("RoBiRank: Ranking via Robust Binary Classification", Yun et al 2014)
In this paper, we introduce DS-MLR

**Doubly-Separable reformulation for Multinomial Logistic Regression (DS-MLR)**

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

makes model parallelism hard

---

**Doubly-Separable form**

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

- **Fully de-centralized distributed algorithm** (data and model fully partitioned across workers)
- **Asynchronous** (communicate model in the background while computing parameter updates)
- Avoids expensive **Bulk Synchronization** steps
Fully de-centralized distributed algorithm (data and model fully partitioned across workers)

Asynchronous (communicate model in the background while computing parameter updates)

Avoids expensive Bulk Synchronization steps
DS-MLR CV

- **Fully de-centralized distributed** algorithm (data and model fully partitioned across workers)
- **Asynchronous** (communicate model in the background while computing parameter updates)
- Avoids expensive **Bulk Synchronization** steps
Delving deeper

DS-MLR
Delving deeper

- Reformulation
- Parallelization
- Empirical Study
Multinomial Logistic Regression (MLR)

Given
- Training data \((x_i, y_i)_{i=1,...,N}\) where \(x_i \in \mathbb{R}^D\)
- Corresponding labels \(y_i \in \{1, 2, \ldots, K\}\)
- \(N, D\) and \(K\) are large \((N >>> D >> K)\)

Goal
The probability that \(x_i\) belongs to class \(k\) is given by:

\[
p(y_i = k | x_i, W) = \frac{\exp(w_k^T x_i)}{\sum_{j=1}^{K} \exp(w_j^T x_i)}
\]

where \(W = \{w_1, w_2, \ldots, w_K\}\) denotes the parameter for the model.
Multinomial Logistic Regression (MLR)

Given
- Training data \((x_i, y_i)_{i=1,\ldots,N}\) where \(x_i \in \mathbb{R}^D\)
- Corresponding labels \(y_i \in \{1, 2, \ldots, K\}\)
- \(N, D\) and \(K\) are large \((N \gg D \gg K)\)

Goal
The probability that \(x_i\) belongs to class \(k\) is given by:

\[
p(y_i = k | x_i, W) = \frac{\exp(w_k^T x_i)}{\sum_{j=1}^{K} \exp(w_j^T x_i)}
\]

where \(W = \{w_1, w_2, \ldots, w_K\}\) denotes the parameter for the model.
Multinomial Logistic Regression (MLR)

The corresponding $l_2$ regularized negative log-likelihood loss:

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

where $\lambda$ is the regularization hyper-parameter.
Multinomial Logistic Regression (MLR)

The corresponding $l_2$ regularized negative log-likelihood loss:

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

where $\lambda$ is the regularization hyper-parameter.

makes model parallelism hard
Reformulation into Doubly-Separable form

**Step 1:** Introduce redundant constraints (new parameters $A$) into the original MLR problem

\[
\min_{W,A} L_1(W,A) = \frac{\lambda}{2} \sum_{k=1}^{K} \|w_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i - \frac{1}{N} \sum_{i=1}^{N} \log a_i \\
\text{s.t. } a_i = \frac{1}{\sum_{k=1}^{K} \exp(w_k^T x_i)}
\]
Step 2: Turn the problem to unconstrained min-max problem by introducing Lagrange multipliers $\beta_i, \forall i = 1, \ldots, N$

$$
\min_{W, A} \max_{\beta} L_2(W, A, \beta) = \frac{\lambda}{2} \sum_{k=1}^{K} \|w_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i - \frac{1}{N} \sum_{i=1}^{N} \log a_i 
+ \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \beta_i a_i \exp(w_k^T x_i) - \frac{1}{N} \sum_{i=1}^{N} \beta_i
$$

Primal Updates for $W, A$ and Dual Update for $\beta$ (similar in spirit to dual-decomp. methods).
Step 2: Turn the problem to unconstrained min-max problem by introducing Lagrange multipliers $\beta_i, \forall i = 1, \ldots, N$

\[
\min_{W, A} \max_{\beta} L_2(W, A, \beta) = \frac{\lambda}{2} \sum_{k=1}^{K} \|w_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i - \frac{1}{N} \sum_{i=1}^{N} \log a_i \\
+ \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \beta_i a_i \exp(w_k^T x_i) - \frac{1}{N} \sum_{i=1}^{N} \beta_i
\]

Primal Updates for $W, A$ and Dual Update for $\beta$ (similar in spirit to dual-decomp. methods).
Step 3: Stare at the updates long-enough

- When $a_i^{t+1}$ is solved to optimality, it admits an exact closed-form solution given by $a_i^* = \frac{1}{\beta_i \sum_{k=1}^{K} \exp(w_k^T x_i)}$.

- Dual-ascent update for $\beta_i$ is no longer needed, since the penalty is always zero if $\beta_i$ is set to a constant equal to 1.

$$\min_{W, A} L_3(W, A) = \frac{\lambda}{2} \sum_{k=1}^{K} \|w_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i - \frac{1}{N} \sum_{i=1}^{N} \log a_i$$

$$+ \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} a_i \exp(w_k^T x_i) - \frac{1}{N}$$
Reformulation into Doubly-Separable form

**Step 4:** Simple re-write

Doubly-Separable form

$$\min_{W,A} \quad \sum_{i=1}^{N} \sum_{k=1}^{K} \left( \frac{\lambda \|w_k\|^2}{2N} - \frac{y_{ik} w_k^T x_i}{N} - \frac{\log a_i}{NK} + \frac{\exp(w_k^T x_i + \log a_i)}{N} - \frac{1}{NK} \right)$$
Reformulation into Doubly-Separable form

Doubly-Separable form

$$\min_{W,A} \sum_{i=1}^{N} \sum_{k=1}^{K} \left( \frac{\lambda \|w_k\|^2}{2N} - \frac{y_{ik}w_k^Tx_i}{N} - \frac{\log a_i}{NK} + \frac{\exp(w_k^Tx_i + \log a_i)}{N} - \frac{1}{NK} \right)$$

Each worker samples a pair $$(w_k, a_i)$$.  

- Update $$w_k$$ using stochastic gradient
- Update $$a_i$$ using its exact closed-form solution
  $$a_i = \frac{1}{\sum_{k=1}^{K} \exp(w_k^Tx_i)}$$
Reformulation into Doubly-Separable form

Doubly-Separable form

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

Each worker samples a pair \((w_k, a_i)\).

- Update \(w_k\) using stochastic gradient
- Update \(a_i\) using its exact closed-form solution \(a_i = \frac{1}{\sum_{k=1}^{K} \exp(w_k^T x_i)}\)
Reformulation into Doubly-Separable form

Doubly-Separable form

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

Each worker samples a pair \((w_k, a_i)\).

- Update \(w_k\) using stochastic gradient
- Update \(a_i\) using its exact closed-form solution \(a_i = \frac{1}{\sum_{k=1}^{K} \exp(w_k^T x_i)}\)
Reformulation into Doubly-Separable form

Doubly-Separable form

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

Each worker samples a pair \((w_k, a_i)\).

- Update \(w_k\) using stochastic gradient
- Update \(a_i\) using its exact closed-form solution \(a_i = \frac{1}{\sum_{k=1}^{K} \exp(w_k^T x_i)}\)
Delving deeper

- Reformulation
- Parallelization
- Empirical Study
Parallelization - Asynchronous

(a) Initial Assignment of \( W \) and \( A \)

NOMAD [Yun et al 2014]
Parallelization - Asynchronous

(a) Initial Assignment of $W$ and $A$

(b) worker 1 updates $w_2$ and communicates it to worker 4
Parallelization - Asynchronous

(a) Initial Assignment of $W$ and $A$

(b) worker 1 updates $w_2$ and communicates it to worker 4

(c) worker 4 can now update $w_2$

NOMAD [Yun et al 2014]
Parallelization - Asynchronous

(a) Initial Assignment of $W$ and $A$

(b) worker 1 updates $w_2$ and communicates it to worker 4

(c) worker 4 can now update $w_2$

(d) As algorithm proceeds, ownership of $w_k$ changes continuously.

NOMAD [Yun et al 2014]
Delving deeper

- Reformulation
- Parallelization
- Empirical Study
Motivation for Hybrid Parallelism

Reddit-Full dataset: Data 228 GB and Model: 358 GB
Motivation for Hybrid Parallelism

Reddit-Full dataset: Data 228 GB and Model: 358 GB
Datasets

Size of Model $O(K \times D)$ vs Size of Data $O(N \times D)$

- LSHTC1-large
- ODP
- Reddit-Full
- CLEF
- NEWS20
- LSHTC1-small
- YouTube8M-Video
- DS-MLR
Empirical Study - Single Machine

Figure: Data fits, Model fits
Empirical Study - Multi Machine

![Graphs showing data fits and model does not fit]

**Figure**: Data fits, Model does not fit
Empirical Study - Multi Machine

Figure: Data does not fit, Model fits
Empirical Study - Multi Machine

Reddit-Full dataset (Data Size: 228 GB, Model Size: 358 GB)

- 211 million examples - $O(N)$
- 44 billion parameters - $O(K \times D)$

Figure: Data does not fit, Model does not fit
Conclusion

We proposed **DS-MLR**

- **Hybrid Parallel** reformulation for MLR $\rightarrow \frac{O(\text{Data})}{P}$ and $\frac{O(\text{Parameters})}{P}$
- **Fully De-centralized** and **Asynchronous** algorithm
- **Avoids** Bulk-synchronization
- Empirical results suggest **wide applicability** and **good predictive performance**
Future Extensions

Design **Doubly-Separable** losses for other machine learning models:

- Extreme multi-label classification
- Log-linear parameterization for undirected graphical models
- Deep Learning

Thank You!
More details

Please check out our paper / poster

Scaling Multinomial Logistic Regression via Hybrid Parallelism

Parameswaran Raman  
University of California, Santa Cruz  
params@ucsc.edu

Sriram Srinivasan  
University of California, Santa Cruz  
ssriniv9@ucsc.edu

Shin Matsushima  
University of Tokyo, Japan  
shin_matsushima@mist.i.u-tokyo.ac.jp

Xinhua Zhang  
University of Illinois, Chicago  
zhangx@uic.edu

Hyokun Yun  
Amazon  
yunhyoku@amazon.com

S.V.N. Vishwanathan  
Amazon  
vishy@amazon.com

ABSTRACT
We study the problem of scaling Multinomial Logistic Regression (MLR) to datasets with very large number of data points in the presence of large number of classes. At a scale where neither data nor the parameters are able to fit on a single machine, we argue that simultaneous data and model parallelism (Hybrid Parallelism) is inevitable. The key challenge in achieving such a form of parallelism in MLR is the log-partition function which needs to be computed across all K classes per data point, thus making model parallelism non-trivial.

To overcome this problem, we propose a reformulation of the original objective that exploits double-separability, an attractive property that naturally leads to hybrid parallelism. Our algorithm (DS-MLR) is asynchronous and completely de-centralized, requiring minimal communication across workers while keeping both data

ACM Reference Format:

1 INTRODUCTION
In this paper, we focus on multinomial logistic regression (MLR), also known as softmax regression which computes the probability of a D-dimensional data point \(x_i \in \{x_1, x_2, \ldots, x_N\}\) belonging to a class \(k \in \{1, 2, \ldots, K\}\). The model is parameterized by a parameter matrix \(W \in \mathbb{R}^{D \times K}\). MLR is a method of choice for several real-world tasks such as Image Classification [20] and Video Recommendation [21].

Code: https://bitbucket.org/params/dsmlr
Acknowledgements

Thanks to all my collaborators!