Scaling Multinomial Logistic Regression via Hybrid Parallelism

Parameswaran Raman
University of California Santa Cruz

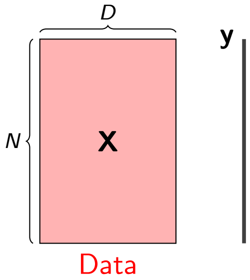
KDD 2019
Aug 4-8

Joint work with:
Sriram Srinivasan, Shin Matsushima, Xinhua Zhang, Hyokun Yun, S.V.N. Vishwanathan

Multinomial Logistic Regression (MLR)

Given:

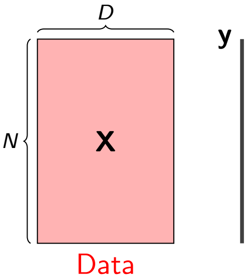
- Training data $(\mathbf{x}_i, y_i)_{i=1,\ldots,N}$, $\mathbf{x}_i \in \mathbb{R}^D$
- Labels $y_i \in \{1, 2, \ldots, K\}$



Multinomial Logistic Regression (MLR)

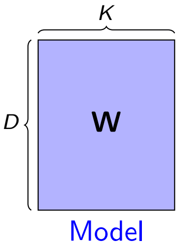
Given:

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



Goal:

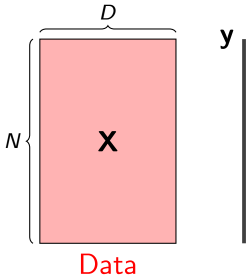
- Learn a model W
- Predict labels for the test data points using W



Multinomial Logistic Regression (MLR)

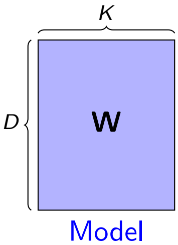
Given:

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



Goal:

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

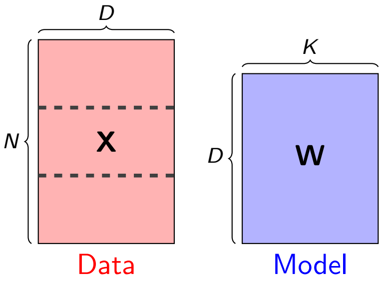


Assume: N, D and K are large ($N >>> D >> K$)

Motivation for Hybrid Parallelism

Popular ways to distribute MLR:

Data parallel (partition data, duplicate parameters)

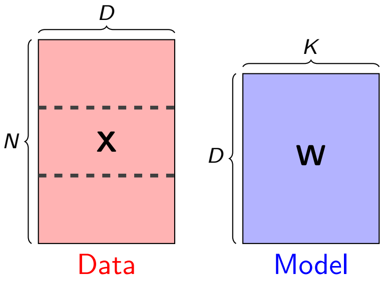


Storage Complexity:
$O(\frac{ND}{\mathbf{P}})$ data, $O(KD)$ model
e.g. L-BFGS

Motivation for Hybrid Parallelism

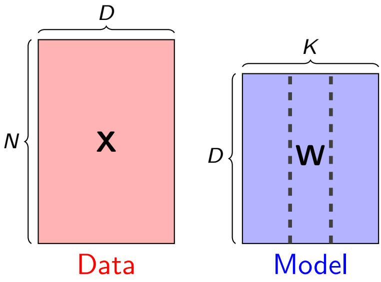
Popular ways to distribute MLR:

Data parallel (partition data, duplicate parameters)



Storage Complexity:
$O(\frac{ND}{\mathbf{P}})$ data, $O(KD)$ model
e.g. L-BFGS

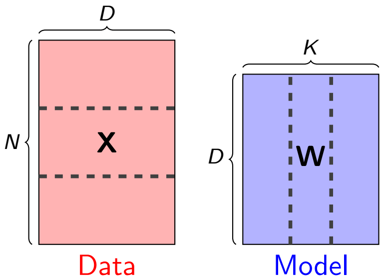
Model parallel (partition parameters, duplicate data)



Storage Complexity:
$O(ND)$ data, $O(\frac{KD}{\mathbf{P}})$ model
e.g. LC [Gopal et al 2013]

Can we get the best of both worlds?

Yes! Hybrid Parallelism

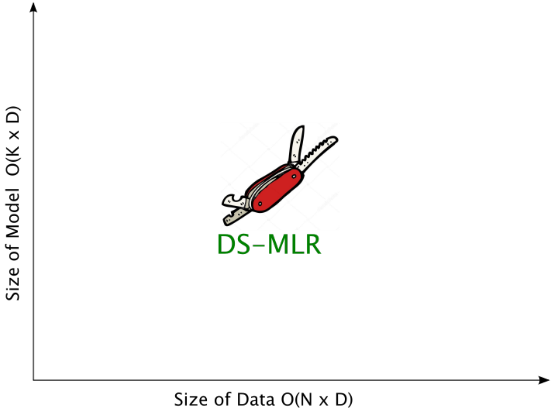


Storage Complexity:
$O(\frac{ND}{\mathbf{P}})$ data, $O(\frac{KD}{\mathbf{P}})$ model

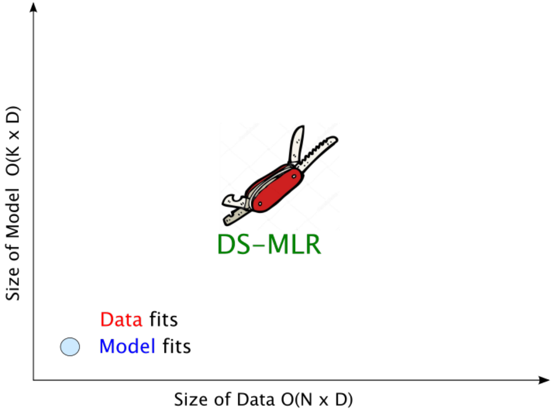
We propose a Hybrid Parallel method **DS-MLR**

Hybrid Parallelism is like a swiss-army knife

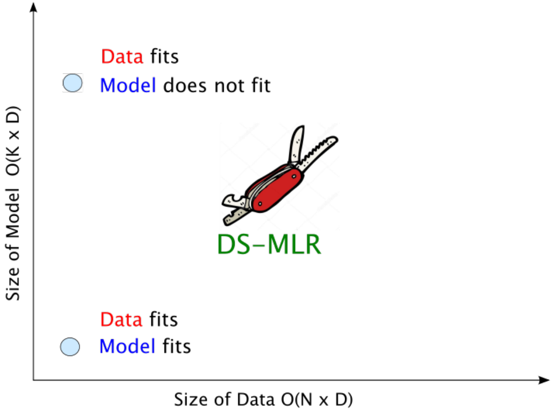
Hybrid Parallelism is like a swiss-army knife



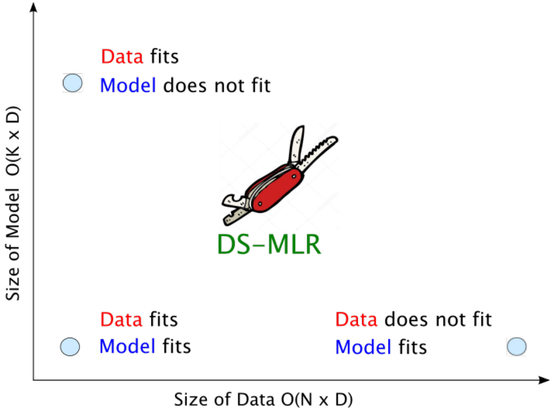
Hybrid Parallelism is like a swiss-army knife



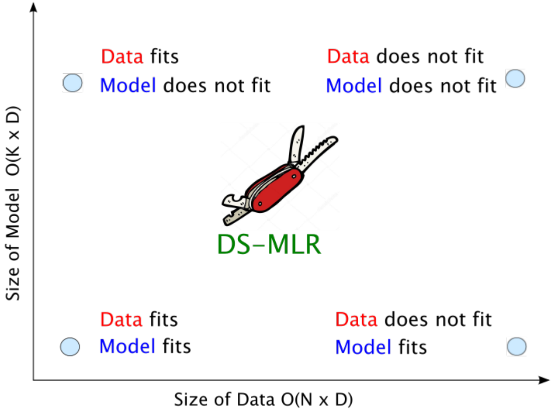
Hybrid Parallelism is like a swiss-army knife



Hybrid Parallelism is like a swiss-army knife



Hybrid Parallelism is like a swiss-army knife



Empirical Study - Multi Machine

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

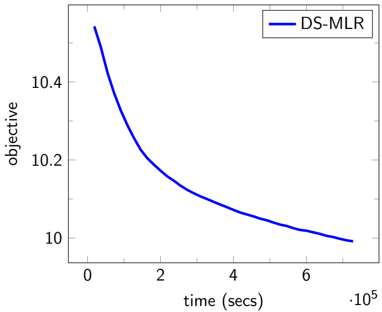


Figure: Data does not fit, Model does not fit

- 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

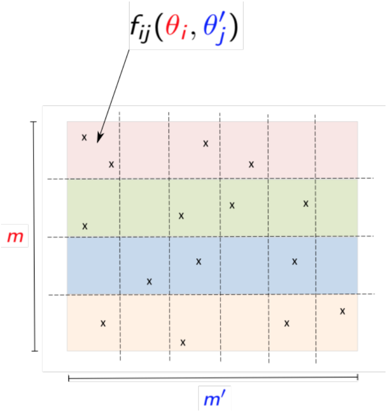
Definition

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

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

Double-Separability

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

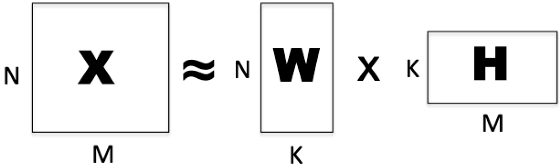


Each sub-function f_{ij} can be computed **independently** and in **parallel**

Are all machine learning models doubly-separable?

Sometimes . . .

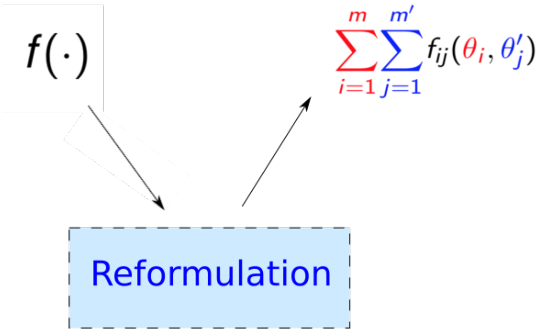
e.g. Matrix Factorization



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

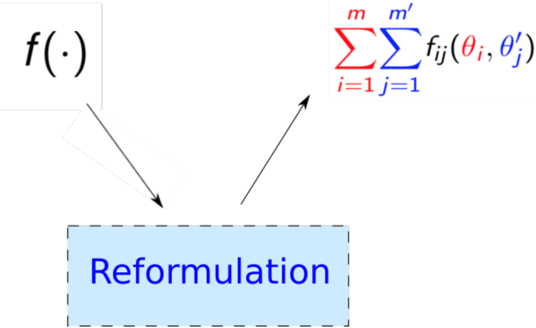
Objective function is trivially doubly-separable!

Others need algorithmic reformulations . . .



- **Binary Classification** ("DSO: Distributed Stochastic Optimization for the Regularized Risk", Matsushima et al 2014)
- **Ranking** ("RoBiRank: Ranking via Robust Binary Classification", Yun et al 2014)

Others need algorithmic reformulations . . .



- **Binary Classification** ("DSO: Distributed Stochastic Optimization for the Regularized Risk", Matsushima et al 2014)
- **Ranking** ("RoBiRank: Ranking via Robust Binary Classification", Yun et al 2014)

In this paper, we introduce DS-MLR

Doubly-**S**eparable reformulation for **M**ultinomial **L**ogistic **R**egression (DS-MLR)

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

Reformulation

Doubly-Separable form

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

DS-MLR

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

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

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 $(\mathbf{x}_i, y_i)_{i=1,\dots,N}$ where $\mathbf{x}_i \in \mathbb{R}^D$
- corresp. labels $y_i \in \{1, 2, \dots, K\}$
- N, D and K are large ($N >>> D >> K$)

Goal

The probability that $\mathbf{x}_i$ belongs to class k is given by:

$$p(y_i = k | \mathbf{x}_i, W) = \frac{\exp(\mathbf{w}_k{}^T \mathbf{x}_i)}{\sum_{j=1}^{K} \exp(\mathbf{w}_j{}^T \mathbf{x}_i)}$$

where $W = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_K\}$ denotes the parameter for the model.

Multinomial Logistic Regression (MLR)

Given

- Training data $(\mathbf{x}_i, y_i)_{i=1,\ldots,N}$ where $\mathbf{x}_i \in \mathbb{R}^D$
- corresp. labels $y_i \in \{1, 2, \ldots, K\}$
- N, D and K are large ($N >>> D >> K$)

Goal

The probability that $\mathbf{x}_i$ belongs to class k is given by:

$$p(y_i = k | \mathbf{x}_i, W) = \frac{\exp(\mathbf{w}_k{}^T \mathbf{x}_i)}{\sum_{j=1}^{K} \exp(\mathbf{w}_j{}^T \mathbf{x}_i)}$$

where $W = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K\}$ denotes the parameter for the model.

Multinomial Logistic Regression (MLR)

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

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

where λ is the regularization hyper-parameter.

Multinomial Logistic Regression (MLR)

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

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

where λ is the regularization hyper-parameter.

Reformulation into Doubly-Separable form

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

$$\min_{W,A} \quad L_1(W, A) = \frac{\lambda}{2}\sum_{k=1}^{K}\|\mathbf{w}_k\|^2 - \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik}\mathbf{w}_k{}^T\mathbf{x}_i - \frac{1}{N}\sum_{i=1}^{N}\log a_i$$

$$\text{s.t.} \quad a_i = \frac{1}{\sum_{k=1}^{K}\exp(\mathbf{w}_k{}^T\mathbf{x}_i)}$$

Reformulation into Doubly-Separable form

Step 2: Turn the problem to unconstrained min-max problem by introducing Lagrange multipliers $\beta_i, \forall i = 1, \ldots, N$

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

Primal Updates for W, A and Dual Update for β (similar in spirit to dual-decomp. methods).

Reformulation into Doubly-Separable form

Step 2: Turn the problem to unconstrained min-max problem by introducing Lagrange multipliers $\beta_i, \forall i = 1, \ldots, N$

$$\min_{W,A} \max_{\beta} \quad L_2(W, A, \beta) = \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{w}_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k{}^T \mathbf{x}_i - \frac{1}{N} \sum_{i=1}^{N} \log a_i$$

$$+ \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \beta_i \, a_i \exp(\mathbf{w}_k{}^T \mathbf{x}_i) - \frac{1}{N} \sum_{i=1}^{N} \beta_i$$

Primal Updates for W, A and Dual Update for β (similar in spirit to dual-decomp. methods).

Reformulation into Doubly-Separable form

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 $a_i^* = \frac{1}{\beta_i \sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i)}$.
- Dual-ascent update for β_i is no longer needed, since the penalty is always zero if β_i is set to a constant equal to 1.

$$\min_{W,A} \quad L_3(W, A) = \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{w}_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k{}^T \mathbf{x}_i - \frac{1}{N} \sum_{i=1}^{N} \log a_i$$
$$+ \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} a_i \exp(\mathbf{w}_k{}^T \mathbf{x}_i) - \frac{1}{N}$$

Reformulation into Doubly-Separable form

Step 4: Simple re-write

Doubly-Separable form

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

Reformulation into Doubly-Separable form

Doubly-Separable form

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

Each worker samples a pair $(\mathbf{w}_k, a_i)$.

- Update $\mathbf{w}_k$ using stochastic gradient
- Update a_i using its exact closed-form solution $a_i = \frac{1}{\sum_{k=1}^{K}\exp({\mathbf{w}_k}^T\mathbf{x}_i)}$

Reformulation into Doubly-Separable form

Doubly-Separable form

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

Each worker samples a pair $(\mathbf{w}_k, a_i)$.

- Update $\mathbf{w}_k$ using stochastic gradient
- Update a_i using its exact closed-form solution $a_i = \frac{1}{\sum_{k=1}^{K}\exp({\mathbf{w}_k}^T\mathbf{x}_i)}$

Reformulation into Doubly-Separable form

Doubly-Separable form

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

Each worker samples a pair $(\mathbf{w}_k, a_i)$.

- Update $\mathbf{w}_k$ using stochastic gradient
- Update a_i using its exact closed-form solution $a_i = \frac{1}{\sum_{k=1}^{K}\exp(\mathbf{w}_k{}^T\mathbf{x}_i)}$

Reformulation into Doubly-Separable form

Doubly-Separable form

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

Each worker samples a pair $(\mathbf{w}_k, a_i)$.

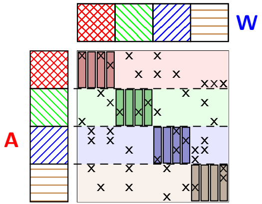
- Update $\mathbf{w}_k$ using stochastic gradient
- Update a_i using its exact closed-form solution $a_i = \frac{1}{\sum_{k=1}^{K}\exp(\mathbf{w}_k{}^T\mathbf{x}_i)}$

Delving deeper

- **Reformulation**
- **Parallelization**
- **Empirical Study**

Parallelization - Asynchronous

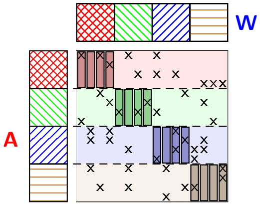
NOMAD [Yun et al 2014]



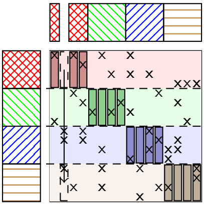
(a) Initial Assignment of W and A

Parallelization - Asynchronous

NOMAD [Yun et al 2014]



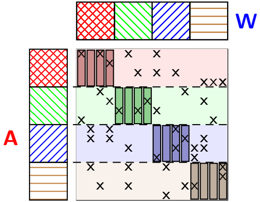
(a) Initial Assignment of W and A



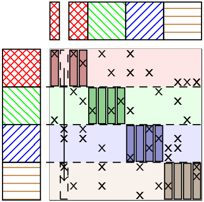
(b) worker 1 updates $\mathbf{w}_2$ and communicates it to worker 4

Parallelization - Asynchronous

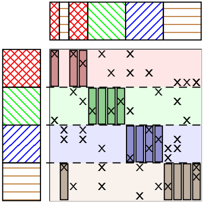
NOMAD [Yun et al 2014]



(a) Initial Assignment of W and A



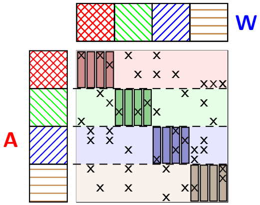
(b) worker 1 updates $\mathbf{w}_2$ and communicates it to worker 4



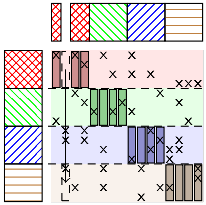
(c) worker 4 can now update $\mathbf{w}_2$

Parallelization - Asynchronous

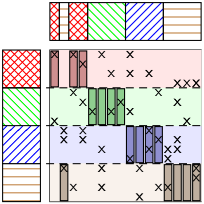
NOMAD [Yun et al 2014]



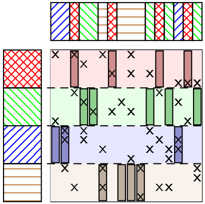
(a) Initial Assignment of W and A



(b) worker 1 updates $\mathbf{w}_2$ and communicates it to worker 4



(c) worker 4 can now update $\mathbf{w}_2$

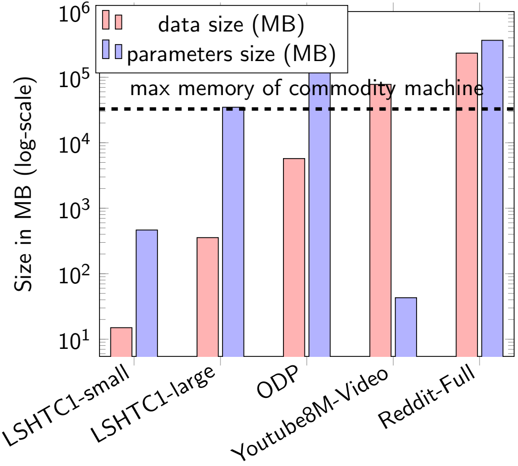


(d) As algorithm proceeds, ownership of $\mathbf{w}_k$ changes continuously.

Delving deeper

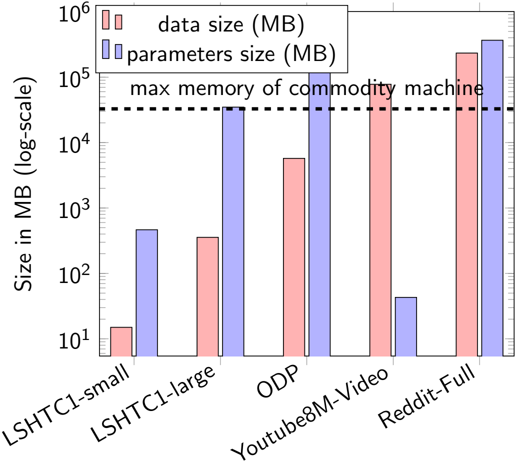
- **Reformulation**
- **Parallelization**
- **Empirical Study**

Motivation for Hybrid Parallelism



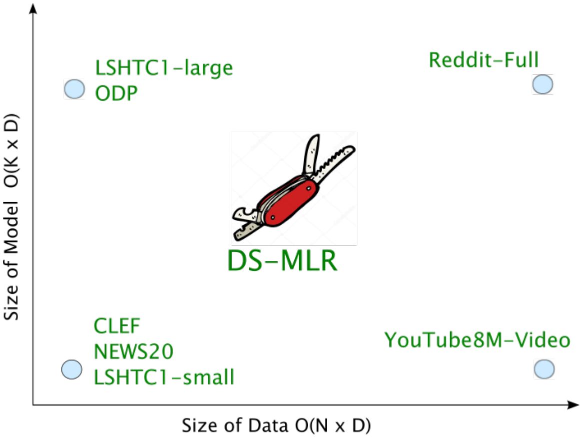
Reddit-Full dataset: Data **228 GB** and Model: **358 GB**

Motivation for Hybrid Parallelism



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

Datasets


LSHTC1-large
ODP
Reddit-Full
DS-MLR
CLEF
NEWS20
LSHTC1-small
YouTube8M-Video
Size of Model O(K x D)
Size of Data O(N x D)

Empirical Study - Single Machine

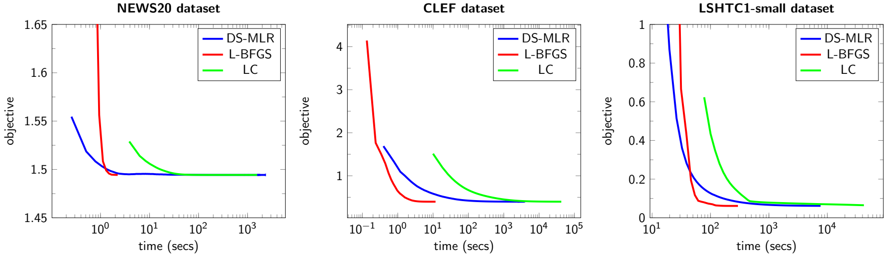


Figure: Data fits, Model fits

Empirical Study - Multi Machine

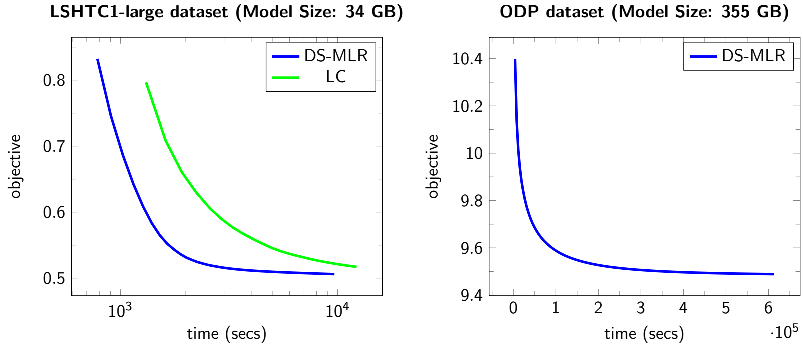


Figure: Data fits, Model does not fit

Empirical Study - Multi Machine

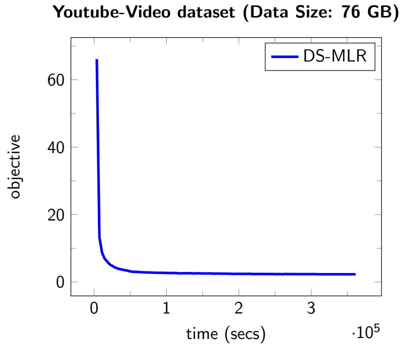


Figure: Data does not fit, Model fits

Empirical Study - Multi Machine

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

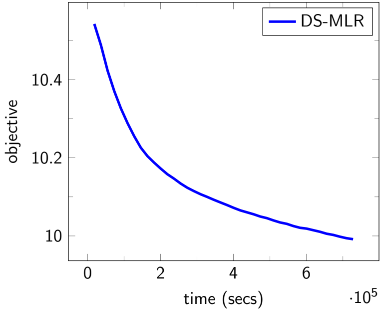


Figure: Data does not fit, Model does not fit

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

Conclusion

We proposed **DS-MLR**

- **Hybrid Parallel** reformulation for MLR $\rightarrow \frac{\text{O(Data)}}{P}$ and $\frac{\text{O(Parameters)}}{P}$
- **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 Illinios, 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:
Parameswaran Raman, Sriram Srinivasan, Shin Matsushima, Xinhua Zhang, Hyokun Yun, and S.V.N. Vishwanathan. 2019. Scaling Multinomial Logistic Regression via Hybrid Parallelism. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3292500.3330837

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 $\mathbf{x}_i \in \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ 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

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

Acknowledgements

Thanks to all my collaborators!